Electronic Research Archive

http://www.aimspress.com/journal/era

ERA, 31(8): 5104–5123.
DOI: 10.3934/era.2023261
Received: 13 April 2023
Revised: 27 June 2023
Accepted: 04 July 2023
Published: 17 July 2023

Research article

Unconditionally stable monte carlo simulation for solving the multi-dimensional Allen–Cahn equation

Youngjin Hwang, Ildoo Kim, Soobin Kwak, Seokjun Ham, Sangkwon Kim and Junseok Kim*

Department of Mathematics, Korea University, Seoul 02841, Republic of Korea

* **Correspondence:** Email: cfdkim@korea.ac.kr.

Abstract: In this study, we present an efficient and novel unconditionally stable Monte Carlo simulation (MCS) for solving the multi-dimensional Allen–Cahn (AC) equation, which can model the motion by mean curvature flow of a hypersurface. We use an operator splitting method, where the diffusion and nonlinear terms are solved separately. The diffusion term is calculated using MCS for the stochastic differential equation, while the nonlinear term is locally computed for each particle in a virtual grid. Several numerical experiments are presented to demonstrate the performance of the proposed algorithm. The computational results confirm that the proposed algorithm can solve the AC equation more efficiently as the dimension of space increases.

Keywords: Monte Carlo simulation; operator splitting method; unconditionally stable scheme; multi-dimensional Allen–Cahn equation

1. Introduction

In this paper, we propose an explicit hybrid numerical method for solving the multi-dimensional Allen–Cahn (AC) equation. The AC equation has emerged as a widely studied mathematical model, playing a pivotal role in the analysis of phase transitions [1]. This equation serves as a powerful tool for understanding the intricate dynamics that govern the boundary between ordered and disordered states in various physical systems [2]. With its ability to capture the evolution and behavior of interfaces, the AC equation has proven instrumental in fields, such as materials science, condensed matter physics and pattern formation [3, 4]. Its innovative formulation has sparked numerous advancements in our comprehension of complex phenomena, shaping our understanding of phase transition dynamics and paving the way for further exploration and discovery in diverse scientific disciplines [5, 6]. The multi-dimensional AC equation is given by

$$\frac{\partial \phi(\mathbf{x},t)}{\partial t} = -\frac{F'(\phi(\mathbf{x},t))}{\epsilon^2} + \Delta\phi(\mathbf{x},t), \quad \mathbf{x} \in \Omega, \ t > 0, \tag{1.1}$$

$$\mathbf{n} \cdot \nabla\phi(\mathbf{x}, t) = 0 \text{ on } \partial\Omega,$$

where $\mathbf{n}$ is outer unit normal vector and $\Omega \subset \mathbb{R}^d$ is a domain. Here, d is the dimension of space, $\phi(\mathbf{x}, t)$ is the concentration, $F(\phi) = 0.25\phi^2(\phi - 1)^2$ and ϵ is an interface layer parameter. The AC equation is the L^2-gradient flow of the following total free energy functional:

$$\mathcal{E}(\phi) = \int_\Omega \left(\frac{F(\phi)}{\epsilon^2} + \frac{1}{2}|\nabla\phi|^2 \right) d\mathbf{x}.$$

In the AC Eq (1.1), we have the following diffusion equation:

$$\frac{\partial\phi(\mathbf{x}, t)}{\partial t} = \Delta\phi(\mathbf{x}, t), \tag{1.2}$$

which causes numerical difficulties when the space dimension is high, i.e., the curse of dimensionality which makes it computationally intractable because of lack of computer memory and excessive computational cost. Classical discretization schemes encounter a significant challenge known as the curse of dimensionality, which was initially introduced by Bellman [7] in the realm of optimal control problems for classical multidimensional partial differential equation (PDE) models [8]. The curse of dimensionality presents a pervasive challenge across diverse domains, such as mathematics, physics, data science and any context that deals with the arrangement, aggregation and examination of spaces with numerous dimensions [9–11]. This predicament becomes particularly conspicuous when endeavoring to tackle extensively utilized and extensively studied second-order semi-linear parabolic PDEs within high-dimensional spaces. There have been many studies to solve Eq (1.2) numerically. Shawn Koohy et al. [11] developed a method to mitigate the curse of dimensionality based on backward stochastic differential equations and deep neural networks to solve multi-dimensional AC equations. In [12], the parabolic sine-Gordon equation was solved using an operator splitting method that splits the linear and nonlinear terms, where the diffusion term is solved using the Fourier spectral method. Ayub et al. [13] solved the diffusion term in the AC equation using the Fourier spectral method. Lee [14] proposed the second-order operator splitting Fourier spectral method for the factional-in-space reaction-diffusion equation and solved the diffusion equation using the discrete cosine transform. Jeong et al. [15] proposed an explicit hybrid finite difference method using the explicit Euler method for the diffusion equation and a closed-form analytical solution for the nonlinear term by splitting the AC equation into a diffusion term and a nonlinear term based on the operator splitting method.

The stochastic heat equation is one of the representative equations among the stochastic partial differential equations. In [16], the authors discussed numerical positivity and almost surely exponential stability of the stochastic heat equation. Sun and Kumar [17] developed a numerical method for high-dimensional stationary Fokker–Planck equations using a tensor decomposition approach, focusing on the curse of dimension. Shrestha [18] solved numerically one-dimensional diffusion equation using the Monte Carlo simulation (MCS). The Fokker–Planck and Langevin equations with no drift term and constant diffusion coefficient are the diffusion equation. Medved et al. [19] compared and analyzed the Fokker–Planck and Langevin equations. Naeimi and Farshad [20] developed the finite volume Monte Carlo (FVMC) method for solving three-dimensional steady-state heat equations. In [21], Naeimi considered the application of MCS to a general form of heat equations

and also investigated the floating random walk procedure to solve multi-dimensional problems with various boundary conditions. Nakagawa [22] verified a proposed method by presenting the results of numerical experiments applying the MCS to the stochastic heat equation with various boundary conditions. Venkiteswaran and Junk [23] developed a Quasi-Monte Carlo (QMC) method for diffusion equations in high dimensions and the numerical results showed faster convergence than the standard MCS. In [24], Novikov et al. explained Laplacian as a random for Brownian particle location and analyzed and presented the particle-based numerical method, MCS of Brownian diffusion effects, for transport equations for spheres (or circles).

Our proposed method is based on the operator splitting scheme. Various methods have been developed for solving the AC type equations based on the operator splitting method. In [25], Lee developed a numerical solution us ing the operator splitting scheme for the energy-dissipative and mass-conservative AC equation. The operator splitting method for the AC equation proposed in [26] is second-order accurate and stable. Cheng et al. [27] presented fast and stable operator splitting methods for phase-field models. Chertock et al. [28] proposed a fast explicit operator splitting method for the convection-diffusion equation. In addition, various studies on the AC equation have been performed by researchers. Poochinapan and Wongsaijai [29] developed a fourth-order compact structure-preserving difference method and verified that the numerical method has fourth-order and second-order accuracy in space and time, respectively. The primary purpose of this paper is to reduce the curse of dimensionality of the numerical algorithms for the AC equation. To mitigate the curse of dimensionality, our proposed method uses the MCS for the diffusion term and a closed-form analytic solution for the nonlinear term. Kai and Wei [30] developed high-order energy-stable methods for the multi-length-scale incommensurate phase-field crystal equation, which permits studying the phase behavior of aperiodic structures. The Cahn–Hilliard (CH) equation is one of the famous phase field models, along with the AC equation. Liupeng and Yunqing [31] developed a second-order scalar auxiliary variable approach in time and employed the linear finite element method in space to solve the CH type equation of the phase field crystal model.

The advantages of the proposed method are as follows:

- It can overcome the dimensional curse in solving the multi-dimensional AC equation.
- As the dimensionality increases, the convergence speed is faster compared to that of the finite difference method.

The contents of this paper are as follows. A detailed numerical solution algorithm is described in Section 2 and then the proposed method is verified through several numerical experiments in Section 3. Finally, a conclusion is presented in Section 4.

2. Numerical algorithm

Using the operator splitting method, the proposed unconditionally stable MCS of the multi-dimensional AC equation consists of two steps. First, we solve the following diffusion equation using MCS.

$$\frac{\partial \phi(\mathbf{x}, t)}{\partial t} = \Delta \phi(\mathbf{x}, t). \tag{2.1}$$

Second, we solve the following ordinary differential equation (ODE) using a closed-form analytic solution [4].

$$\frac{\partial\phi(\mathbf{x},t)}{\partial t} = -\frac{F'(\phi(\mathbf{x},t))}{\epsilon^2} = -\frac{\phi^3(\mathbf{x},t) - 1.5\phi^2(\mathbf{x},t) + 0.5\phi(\mathbf{x},t)}{\epsilon^2}. \tag{2.2}$$

Instead of solving the PDE (2.1), we use the following stochastic differential equation (SDE):

$$d\mathbf{X}(t) = \sqrt{2}d\mathbf{W}(t), \tag{2.3}$$

where $\mathbf{X}(t)$ is the position of a particle at time t, d is the dimension of the space and $d\mathbf{W}(t) = \sqrt{dt}\mathbf{Z}$ is the Brownian motion. Here, $\mathbf{Z}$ is a random vector variable drawn from a standardized multivariate normal distribution. Then, Eq (2.1) is the Fokker–Planck equation of Eq (2.3) [18].

For simplicity of exposition, let us consider the solution algorithm in two-dimensional space, $\Omega = (L_x, R_x) \times (L_y, R_y)$. Let the spatial steps be h_x, h_y, the temporal step be Δt, the total number of particles be M and $N_x = (R_x - L_x)/h_x$, $x_i = L_x + (i-0.5)h_x$ for $i = 1, \ldots, N_x$, $N_y = (R_y - L_y)/h_y$, $y_j = L_y + (j-0.5)h_y$ for $j = 1, \ldots, N_y$. We denote the numerical approximations of $\phi(x_i, y_j, n\Delta t)$ by ϕ_{ij}^n, the position of k-th particle at time $n\Delta t$ by $\mathbf{X}_k^n$ and the value of the k-th particle at time $n\Delta t$ by ψ_k^n. The initial conditions for the particle positions $\mathbf{X}_k^0$, $k = 1, 2, \ldots, M$ and the corresponding particle values ψ_k^0, $k = 1, 2, \ldots, M$, are given as follows. If $\phi(x_i, y_j, 0) > 0.05$ for $i = 1, 2, \ldots, N_x$, $j = 1, 2, \ldots, N_y$, it is given as

$$\begin{aligned} \mathbf{X}_k^0 &= (x_i, y_j), \qquad \text{for } k = qm,\ qm+1,\ \ldots,\ (q+1)m-1, \\ \psi_k^0 &= \frac{\phi(\mathbf{X}_k^0, 0)}{m}, \quad \text{for } k = qm,\ qm+1,\ \ldots,\ (q+1)m-1, \end{aligned}$$

where q is some integer and m is the number of particles given at per point. The particle position X_k^n and particle value ψ_k^n form pairs for $k = 1, 2, \ldots, M$. For a better understanding, a schematic diagram showing the positions of some initial particles $\mathbf{X}_k^0$ for $k = 1, \ldots, m$ in one-dimensional space is presented in Figure 1.

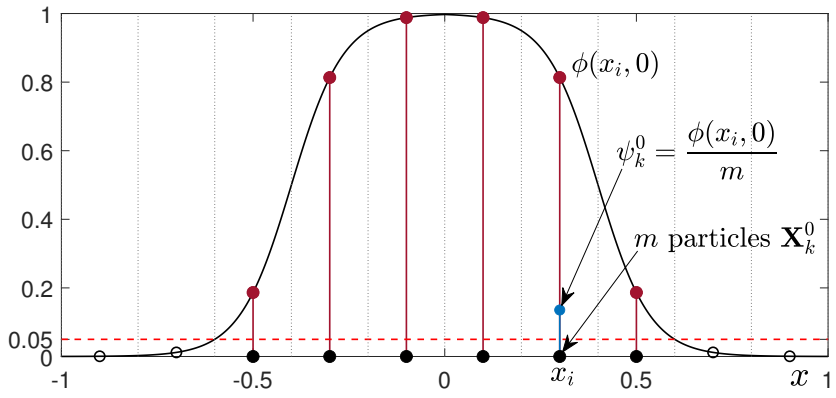


Figure 1. Schematic diagram of the initial particles position in one-dimensional space.

First, we solve the discretized diffusion equation. We update the particle positions $\mathbf{X}_k^n$ for $k = 1, \ldots, M$ according to Eq (2.3) as follows:

$$\mathbf{X}_k^{n+1} = \mathbf{X}_k^n + \sqrt{2\Delta t}\mathbf{Z}, \text{ for } k = 1, \ldots, M. \tag{2.4}$$

Second, solve for the discretized nonlinear Eq (2.2). We define the local sum of particle values as follows:

$$\phi_{ij}^{*} = \sum_{\mathbf{X}_k^{n+1} \in \Omega_{ij}} \psi_k^n, \tag{2.5}$$

where $\Omega_{ij} = (L_x + h_x(i-1), L_x + h_x i) \times (L_y + h_y(j-1), L_y + h_y j)$ is a cell for some i and j. Figure 2(d), (e) and (g) schematically show this procedure. If $\phi_{ij}^{*} > tol$, $i = 1, 2, \ldots, N_x$, $j = 1, 2, \ldots, N_y$, then we analytically solve the following ODE (2.2):

$$\frac{d\Phi(t)}{dt} = -\frac{F'(\Phi(t))}{\epsilon^2} = -\frac{\Phi^3(t) - 1.5\Phi^2(t) + 0.5\Phi(t)}{\epsilon^2} \tag{2.6}$$

with an initial condition $\Phi(0) = \phi_{ij}^{*}$, where tol is a given tolerance and is used to avoid division by an extremely small number. The analytic solution of Eq (2.6) with the initial condition ϕ_{ij}^{*} after time Δt is given as

$$\phi_{ij}^{**} = \Phi(\Delta t) = \frac{1}{2} + \frac{\phi_{ij}^{*} - 0.5}{\sqrt{e^{-\frac{\Delta t}{2\epsilon^2}} + (2\phi_{ij}^{*} - 1)^2 \left(1 - e^{-\frac{\Delta t}{2\epsilon^2}}\right)}}. \tag{2.7}$$

Then, we update the particle value ψ_k^n for the k-th particle position $\mathbf{X}_k^n$ for $k = 1, 2, \ldots, M$ as follows:

$$\psi_k^{n+1} = \frac{\phi_{ij}^{**}}{\phi_{ij}^{*}} \psi_k^n, \quad \text{for } \mathbf{X}_k^{n+1} \in \Omega_{ij}. \tag{2.8}$$

This means that we have analytically solved Eq (2.6) with the initial condition $\Phi(0) = \phi_{ij}^{*}$ for each cell Ω_{ij} as follows:

$$\sum_{\mathbf{X}_k^{n+1} \in \Omega_{ij}} \psi_k^{n+1} = \frac{\phi_{ij}^{**}}{\phi_{ij}^{*}} \sum_{\mathbf{X}_k^{n+1} \in \Omega_{ij}} \psi_k^n = \phi_{ij}^{**}.$$

Figure 2(f),(h) show the results of the final procedure when the value of ϕ_{ij}^{*} is less than 0.5 or greater than 0.5, respectively.

We note that once $\psi_k^{n+1} > 0.5$, then $\psi_k^{\hat{n}} > 0.5$ is satisfied for all time, $\hat{n} > n+1$. To avoid this situation, if $\psi_k^{n+1} > 0.5$, then we add one additional particle $\mathbf{X}_{M+1}^{n+1}$ at the same position $\mathbf{X}_k^{n+1}$, i.e., $\mathbf{X}_{M+1}^{n+1} = \mathbf{X}_k^{n+1}$. In addition, we divide the value of particle ψ_k^{n+1} by 2, i.e., $\psi_k^{n+1} = \psi_k^{n+1}/2$ and set $\psi_{M+1}^{n+1} = \psi_k^{n+1}$.

3. Numerical experiments

For an appropriate interfacial parameter value, we consider an equilibrium solution of the AC equation, which is given by $\phi(x) = 0.5(1 + \tanh(x/(2\sqrt{2}\epsilon)))$. Based on this equilibrium solution, we define the interfacial layer parameter ϵ_l as follows:

$$\epsilon_l = lh/[4\sqrt{2}\tanh^{-1}(0.9)],$$

which implies that we have approximately lh width across the interfacial transition layer [32]. The schematic diagram for the interfacial layer is presented in Figure 3.

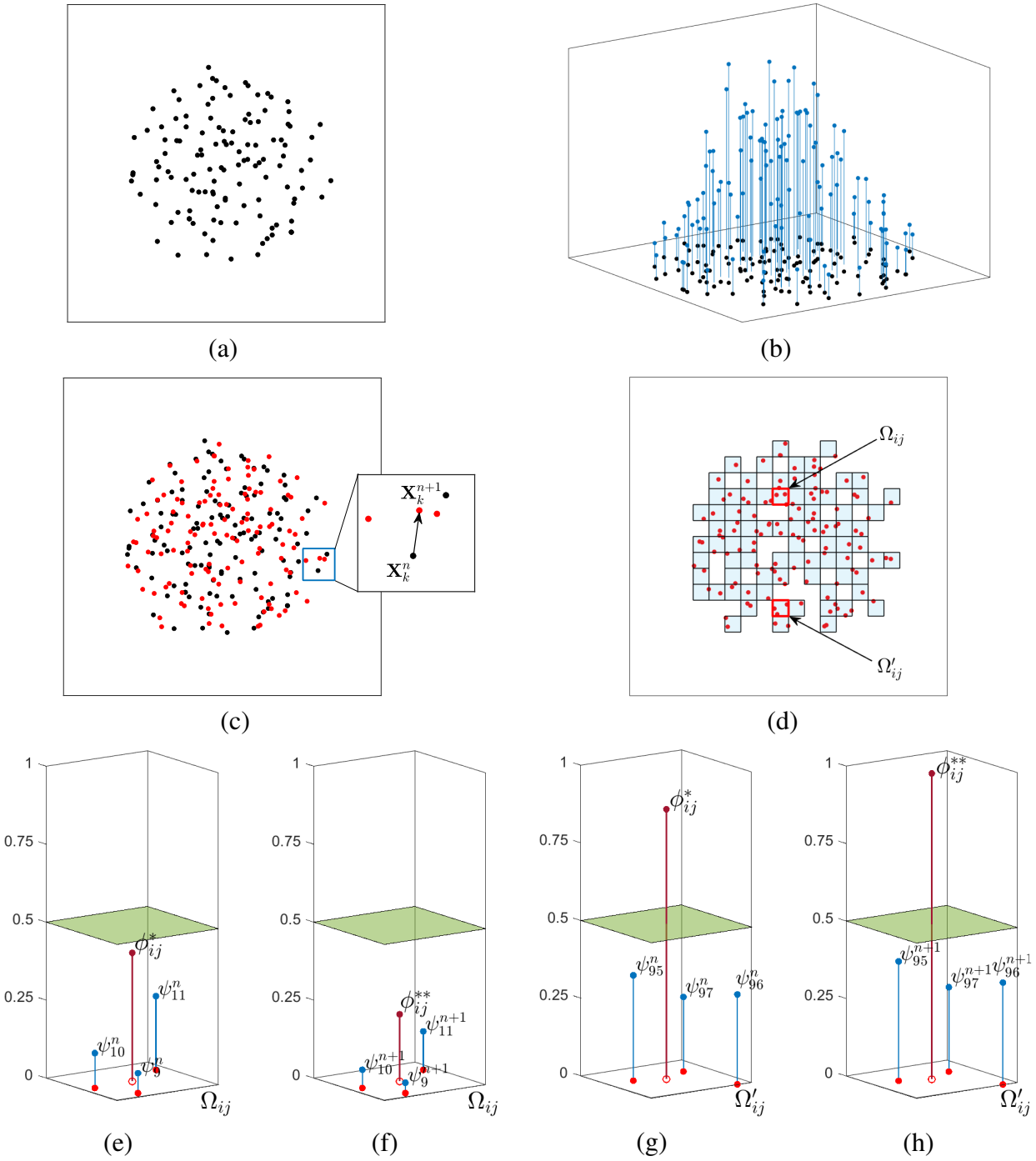


Figure 2. (a) Position of $\mathbf{X}_k^n$, (b) ψ_k^n values at $\mathbf{X}_k^n$, (c) new position $\mathbf{X}_k^{n+1}$ according to Eq (2.3), (d) $\mathbf{X}_k^{n+1}$ and Ω_{ij}, (e) ϕ_{ij}^* in Ω_{ij} and $\phi_{ij}^* < 0.5$, (f) ϕ_{ij}^{**} in Ω_{ij} and $\phi_{ij}^{**} < 0.5$, (g) ϕ_{ij}^* in Ω'_{ij} and $\phi_{ij}^* > 0.5$ and (h) ϕ_{ij}^{**} in Ω'_{ij} and $\phi_{ij}^{**} > 0.5$.

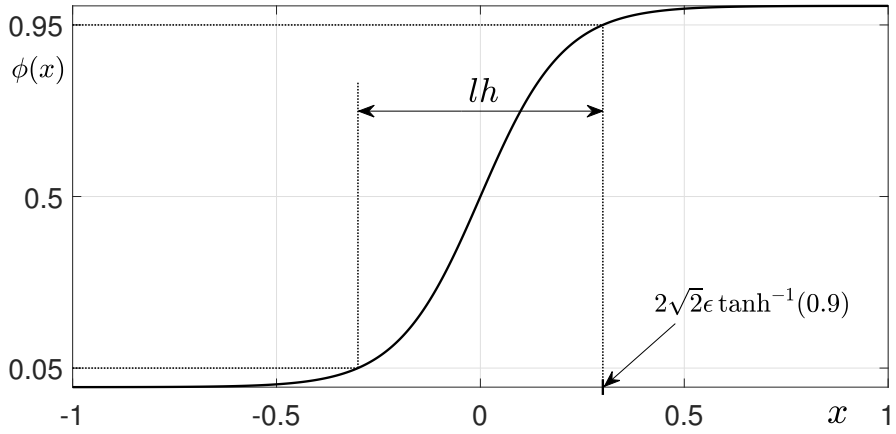


Figure 3. Schematic diagram of the interfacial layer.

3.1. Stability tests

In this section, we demonstrate the stability of the proposed algorithm. For simplicity of proof, we consider the proposed algorithm in two-dimensional space as in Section 2. Similarly, we can prove this in another multi-dimensional space. We verify the unconditional stability and energy decreasing for the proposed algorithm through numerical experiments. We have used the zero Neumann boundary condition as

$$\begin{aligned} \phi^n_{0,j} &= \phi^n_{1,j}, \ \phi^n_{N_x+1,j} = \phi^n_{N_x,j}, \ \text{for } j = 1, 2, \ldots, N_y, \\ \phi^n_{i,0} &= \phi^n_{i,1}, \ \phi^n_{i,N_y+1} = \phi^n_{i,N_y}, \ \text{for } i = 1, 2, \ldots, N_x. \end{aligned}$$

The discrete total energy $\mathcal{E}^d_h(\phi^n)$, discrete total mass $\mathcal{M}^d_h(\phi^n)$ and discrete maximum value $Max^d(\phi^n)$ are defined as:

$$\begin{aligned} \mathcal{E}^d_h(\phi^n) &= \sum_{i=1}^{N_x}\sum_{j=1}^{N_y}\left[\frac{F(\phi^n_{ij})}{\epsilon^2} + \frac{1}{2}\left(\frac{\phi^n_{i+1,j}-\phi^n_{ij}}{h}\right)^2 + \frac{1}{2}\left(\frac{\phi^n_{i,j+1}-\phi^n_{ij}}{h}\right)^2\right]h^2, \\ \mathcal{M}^d_h(\phi^n) &= \sum_{i=1}^{N_x}\sum_{j=1}^{N_y}\phi^n_{ij}h^2, \ \ Max^d(\phi^n) = \max_{1\le i\le N_x, 1\le j\le N_y}(\phi^n_{ij}). \end{aligned}$$

The first test is a stability test of the proposed method for the spatial steps. In this section, we use the following initial condition on a computational domain $\Omega = (-3, 3) \times (-3, 3)$

$$\phi(x, y, 0) = \frac{1}{2}\left(1 + \tanh\frac{1 - \sqrt{x^2 + y^2}}{2\sqrt{2}\epsilon}\right), \ \ (x, y) \in \Omega.$$

Here, the parameters used are $\Delta t = 0.0025$, $T = 0.5$, $\epsilon = 0.2$ and different spatial steps $h = 0.2,\ 0.1,\ 0.05$ with corresponding $m = 1000,\ 2000,\ 4000$, respectively. Numerical experiments were performed to assess the stability for different spatial steps $h = 0.2,\ 0.1,\ 0.05$ and the results of the stability test are depicted in Figure 4.

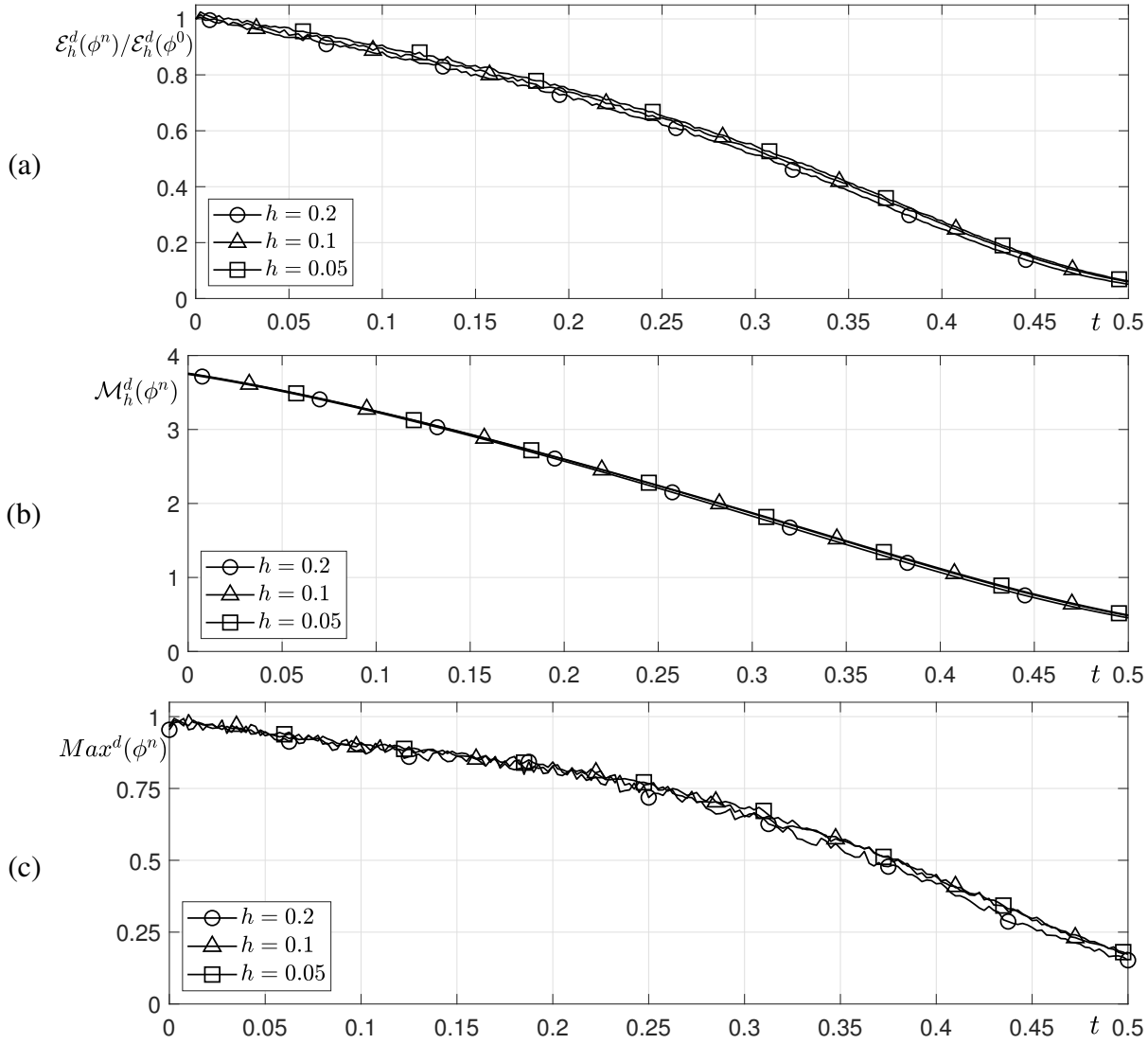


Figure 4. The results of stability tests with different spatial steps $h = 0.2,\ 0.1,\ 0.05$. (a) The normalized discrete total energy, (b) discrete total mass and (c) discrete maximum value.

As a second test, we consider the stability of the temporal step. We used parameters $h = 0.05$, $T = 0.5$, $m = 1400$, $\epsilon = \epsilon_{10}$ and different temporal steps $\Delta t = 2h^2,\ h^2,\ 0.5h^2$. Figure 5 (a)–(c) show the results of stability tests as discrete total energy, discrete total mass and discrete maximum value, respectively.

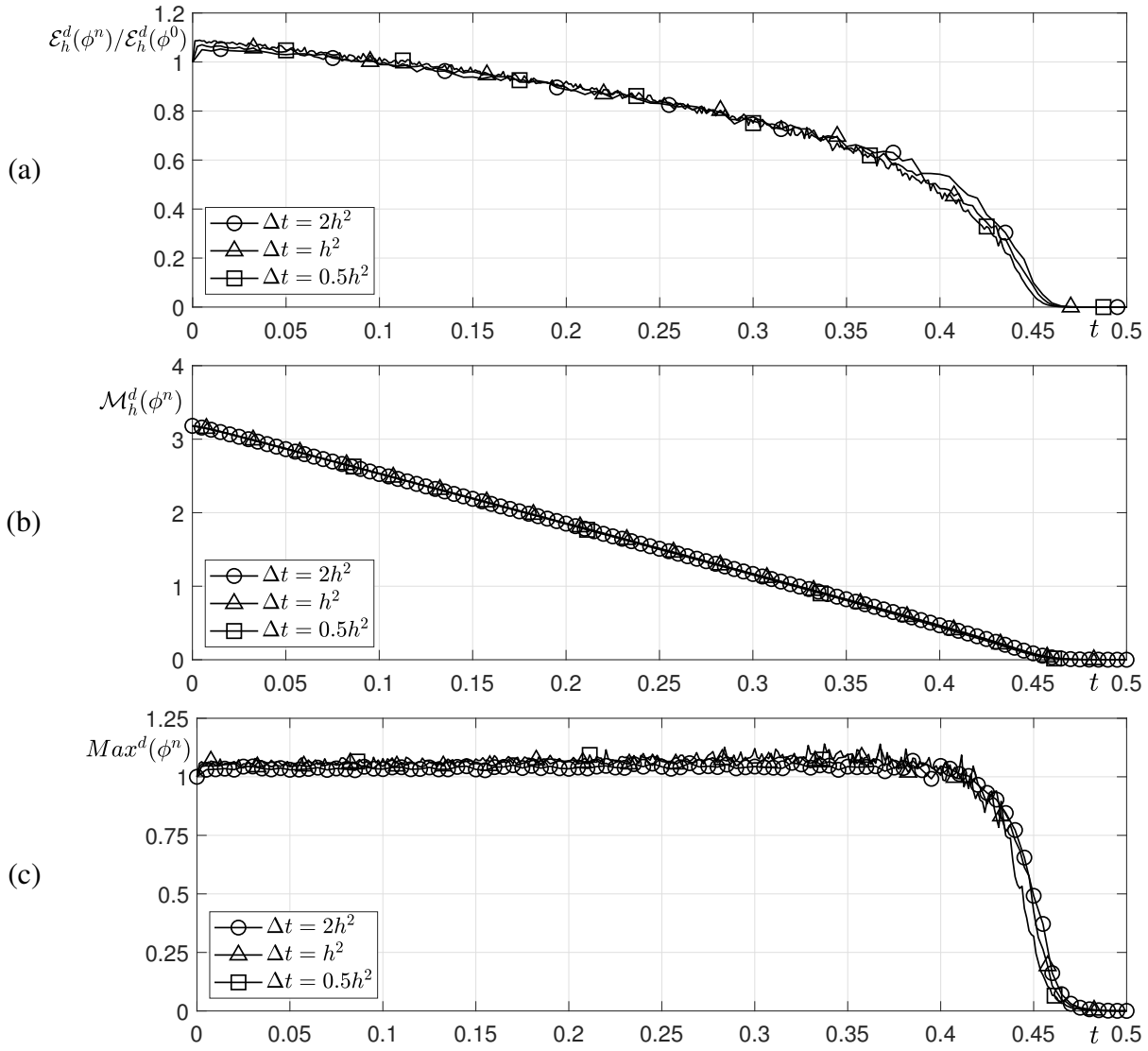


Figure 5. The results of stability tests with different temporal steps $\Delta t = 2h^2, h^2, 0.5h^2$. (a) The normalized discrete total energy, (b) discrete total mass and (c) discrete maximum value.

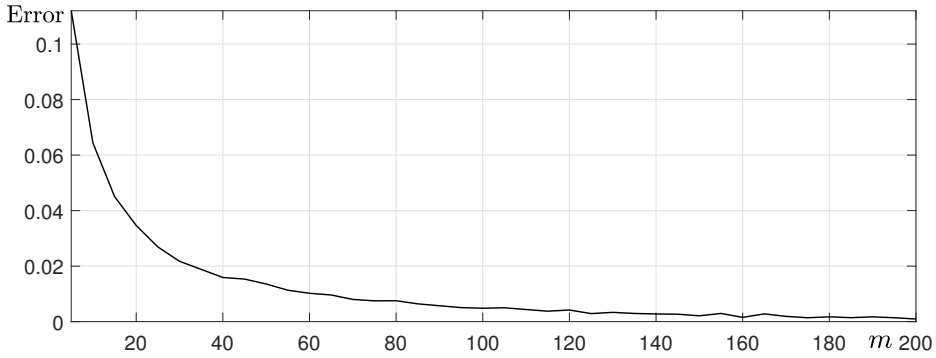
We numerically performed spatial and temporal stability tests and observed that the numerical solution does not explode over various spatial and temporal steps.

3.2. Convergence test

The proposed method is affected by the initial total number of particles M. However, the M required to obtain a numerical solution up to the total time T is theoretically impossible or difficult to obtain. Therefore, we performed a numerical test for the initial number of particles per point m required to obtain the enough M in two-and three-dimensional spaces. Note that this numerical test is a test to show the approximate m required for the proposed method. The numerical test repeats the numerical simulation by adding 5 to m until the absolute error between the analytical solution and the numerical solution of the proposed method is less than tolerance 0.001. The parameters used are $r = 0.8$, spatial

step $h = 0.05$, total time $T = 0.01$, $\epsilon = \epsilon_8$ and $x_i = -2.025 + 0.05\,j$ for $i = 1, \ldots, 80$, $y_j = -2.025 + 0.05\,j$ for $j = 1, \ldots, 80$. Here, m particles are given to the (x_i, y_j) when $\phi(x_i, y_j, 0)$ is greater than 0.05 at the (x_i, y_j), therefore the initial total number of particles M becomes 1264 m and 33,552 m in two- and three-dimensional space, respectively. Figure 6(a),(b) show the absolute errors of numerical and analytic solutions in two-and three-dimensional space for the initial number of particles per point m, respectively. As a result of the numerical test, the m required for the absolute error to be smaller than the absolute error is 200 in two-dimensional space and 105 in three-dimensional spaces. We observe that the absolute error decreases sharply as m increases and the required m decreases as the dimension of space increases. However, because the initial total number of particles M used is $252,800$ in the two-dimensional space and $3,522,960$ in the three-dimensional space, it increases as the dimension of space increases.

(a) 2D



(b) 3D

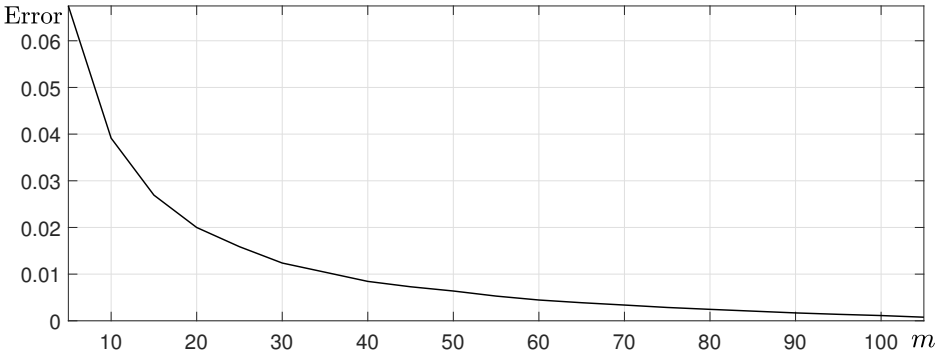


Figure 6. Absolute error of numerical and analytical solutions for initial number of particles per point m in two-and three-dimensional spaces.

3.3. Motion by mean curvature and ratio of CPU times

As ϵ goes to zero, the zero level set of the solution of the AC equation converges to the mean curvature flow [33, 34]. Thus, we consider the mean curvature flow of $(d-1)$-dimensional spheres in d-dimensional space. Let us define the gamma function as

$$\Gamma(s) = \int_0^\infty e^{-t} t^{s-1} dt, \quad \text{for } s > 0,$$

thus $\Gamma(k) = (k-1)!$, $\Gamma(k+1/2) = (k-1/2)(k-3/2)\cdots(1/2)\sqrt{\pi}$, for a positive integer k. Then, the area of sphere with radius $R = 1$ at the origin is $A(R) = 2\pi^{\frac{d}{2}}/\Gamma\left(\frac{d}{2}\right)$. We define a d-ball as a region

comprising of all points within a radius R of a given point in d-dimensional space. The volume of d-ball with radius $R = 1$ at the origin is

$$V(R) = A(R)\int_0^R r^{d-1}dr = \frac{2\pi^{\frac{d}{2}}}{d\,\Gamma\left(\frac{d}{2}\right)}.$$

Therefore, we obtain volume of d-ball with any radius $R > 0$ at the point $\mathbf{x} \in \mathbb{R}^d$ as $V(R) = 2\pi^{\frac{d}{2}}R^d/[d\,\Gamma(\frac{d}{2})]$. More details can be found in [35]. Then, we can express the radius R in terms of V:

$$R(V) = \frac{\left(d\,\Gamma\left(\frac{d}{2}\right)\right)^{\frac{1}{d}}}{2^{\frac{1}{d}}\sqrt{\pi}}V^{\frac{1}{d}}. \tag{3.1}$$

In d-dimensional space, the radius of d-ball can be calculated using the volume of d-ball and the calculated radius becomes the radius of the $(d-1)$-dimensional sphere. Then, mean curvature of sphere of radius R is $H = (d-1)/R$. The mean curvature flow equation reduces to the following ordinary differential equation, for a sphere of initial radius R_0:

$$\frac{d}{dt}R(t) = -\frac{d-1}{R(t)}, \quad R(0) = R_0. \tag{3.2}$$

The solution of Eq (3.2) is $R(t) = \sqrt{R_0^2 - 2(d-1)t}$. The volume of the numerical solution is obtained as follows: $V(n\Delta t) = \sum_{k=1}^{M}\psi_k^n h^d$. Therefore, by substituting the obtained volume $V(n\Delta t)$ into Eq (3.1), the radius $R(n\Delta t)$ is obtain. Because the proposed method is used in high-dimensional, instead of calculating the radius using the distance from the center to the zero-contour, we use the volume of the solution to calculate the radius. In numerical experiments, the initial radius R_0 is calculated numerically using the volume of the initial condition. Consequently, we assumed that for the analytic solution of the zero level set of the AC equation for the motion by mean curvature, ϵ becomes zero.

3.3.1. Two-dimensional space

The initial condition is given as

$$\phi(x, y, 0) = \frac{1}{2}\left(1 + \tanh\frac{r - \sqrt{x^2 + y^2}}{2\sqrt{2}\epsilon}\right)$$

on the computational domain $\Omega = (-2, 2)\times(-2, 2)$, where r is radius. The analytic solution of the zero level set of the AC equation, as ϵ converges to zero, is given by:

$$R(t) = \sqrt{R_0^2 - 2t},$$

which represents the analytic solution for the motion by mean curvature of a 1-dimensional sphere in 2-dimensional space. The parameters used are $r = 0.8$, spatial step $h = 0.05$, total time $T = 0.1$, $\epsilon = \epsilon_8$, $m = 600$ and $x_i = -2.025 + 0.05\,j$ for $i = 1, \ldots, 80$, $y_j = -2.025 + 0.05\,j$ for $j = 1, \ldots, 80$. Because the initial radius is 0.8, the analytic solution to the motion by mean curvature of the AC equation is $R(t) = \sqrt{0.64 - 2t}$. To verify the temporal convergence of the proposed method and compare the

explicit Euler method, we used different temporal steps $\Delta t = 10h^2,\ 5h^2,\ 2.5h^2$ for the proposed method and $\Delta t = 0.1h^2$ for the explicit Euler method. We used the initial number of particles per point $m = 600$, thus the initial particles are given 600 particles at position (x_i, y_j) if $\phi(x_i, y_j, 0)$ is greater than 0.05 at the cell center (x_i, y_j). Therefore, the initial number of particles $M = 758,400$ for numerical experiments in two-dimensional space. In addition, for explicit Euler method, a value was given only when $\phi(x_i, y_j, 0)$ at the cell center (x_i, y_j) was greater than 0.05 to make the initial radius R_0 the same. We consider the effects of temporal steps on the dynamics of the AC equation. Figure 7 shows temporal evolutions of the radius $R(t)$ using the proposed method with different temporal steps $\Delta t = 10h^2,\ 5h^2,\ 2.5h^2$ and the explicit Euler method with temporal step $\Delta t = 0.1h^2$ in two-dimensional space. As the temporal step Δt decreases, we can observe that the numerical solutions converge to the analytic solution $R(t)$. In addition, it can be seen that the proposed method is unconditional stable with respect to time when a enough initial total number of particles M is given.

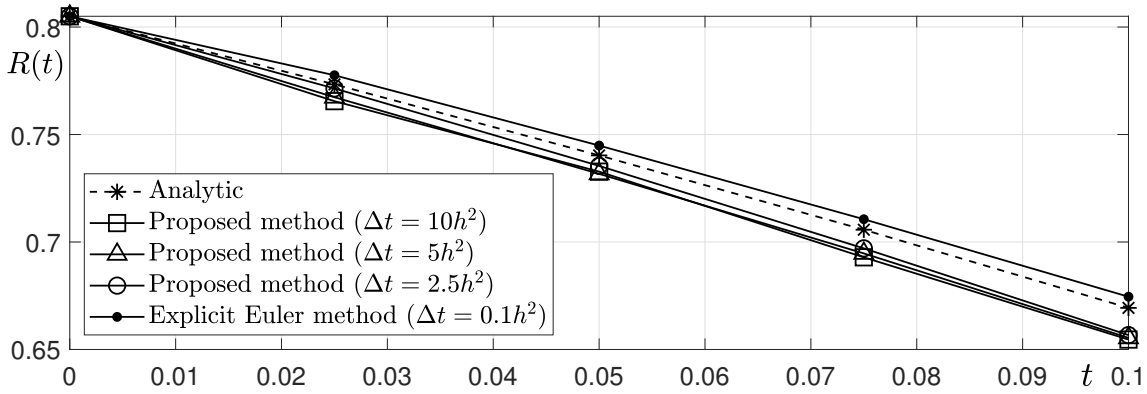


Figure 7. Temporal evolution of radius for the proposed method with different temporal steps and for the explicit Euler method in two-dimensional space.

To verify the computational speed of the numerical algorithm, we defined CPU time as the elapsed time of the main algorithm and defined the CPU time$_{Ratio}$ as the value obtained by dividing the CPU time of the numerical method by the CPU time of the explicit Euler method. Table 1 lists the CPU time and CPU time$_{Ratio}$ to solve the AC equation up to total time $T = 0.1$ for the proposed method with different temporal steps and the explicit Euler method in two-dimensional space. In low dimensional space, such as two-dimension space, the explicit Euler method computes faster than the proposed method. However, we perform numerical experiments in three- and four-dimensional space to show that our proposed method is more efficient in solving the AC equation as the dimension of space increases.

Table 1. CPU time to solve the AC equation in two-dimensional space up to total time $T = 0.1$ and the CPU time$_{Ratio}$ accordingly.

	Proposed method			Explicit Euler method
temporal step	$\Delta t = 10h^2$	$\Delta t = 5h^2$	$\Delta t = 2.5h^2$	$\Delta t = 0.1h^2$
CPU time (s)	2.029	2.334	2.987	0.258
CPU time$_{Ratio}$	7.858	9.040	11.569	1.000

Unless otherwise specified, parameters used in two-dimensional space expand equally in three- and four-dimensional spaces.

3.3.2. Three-dimensional space

Numerical experiments similar to those in two-dimensional space are performed in three-dimensional space as well. We consider the initial conditions as

$$\phi(x, y, z, 0) = \frac{1}{2}\left(1 + \tanh\frac{r - \sqrt{x^2 + y^2 + z^2}}{2\sqrt{2}\epsilon}\right), \quad (x, y, z) \in \Omega$$

on computational domain $\Omega = (-2, 2) \times (-2, 2) \times (-2, 2)$. The analytic solution of the zero level set of the AC equation, as ϵ converges to zero, is given by:

$$R(t) = \sqrt{R_0^2 - 4t},$$

which represents the analytic solution for the motion by mean curvature of a 2-dimensional sphere in 3-dimensional space. The parameters used in the three-dimensional space are extensions of the parameters used in two-dimensional except for the initial number of particles per point $m = 300$ and therefore initial number of particles $M = 10,065,600$. Because the initial radius is 0.8, the analytic solution to the motion by mean curvature of the AC equation is $R(t) = \sqrt{0.64 - 4t}$.

Figure 8 shows the temporal evolution of radius of the proposed method with difference temporal steps $\Delta t = 10h^2$, $5h^2$, $2.5h^2$ and explicit Euler method with temporal step $\Delta t = 0.1h^2$ in three-dimensional space. We can observe that, given an enough initial total number of particles M in three-dimensional space, the proposed method generates good results despite of the large temporal steps Δt.

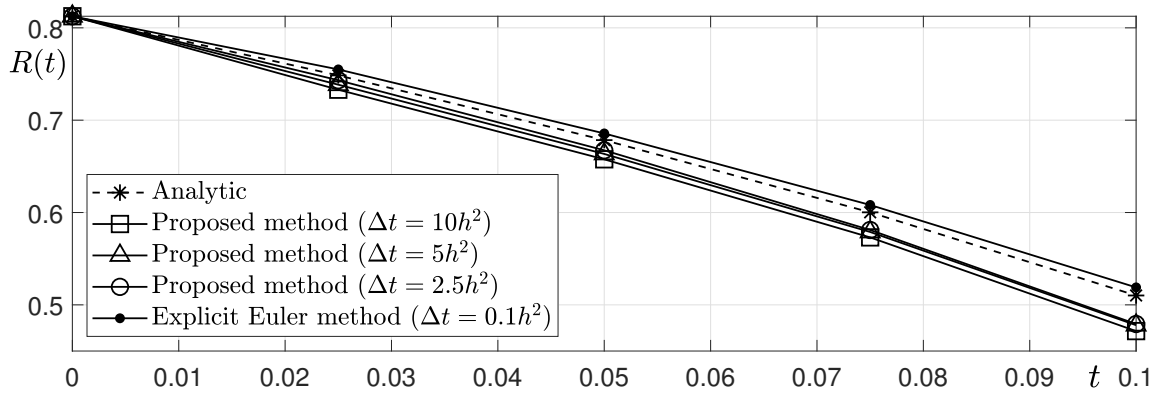


Figure 8. Temporal evolution of radius in three-dimensional space for the proposed method with different temporal steps and for the explicit Euler method.

Table 2 lists the CPU time and CPU time$_{Ratio}$ to solve the AC equation up to total time $T = 0.1$ for the proposed method with different temporal steps and the explicit Euler method in three-dimensional space. Compared with Table 1, we can observe that the overall CPU time$_{Ratio}$ is reduced, which means that the efficiency of the proposed method in three-dimensional space is better than that in two-dimensional space.

Table 2. CPU time to solve the AC equation in three-dimensional space up to total time $T = 0.1$ and the CPU time$_{Ratio}$ accordingly.

	Proposed method			Explicit Euler method
temporal step	$\Delta t = 10h^2$	$\Delta t = 5h^2$	$\Delta t = 2.5h^2$	$\Delta t = 0.1h^2$
CPU time (s)	21.330	26.535	38.671	18.182
CPU time$_{Ratio}$	1.173	1.459	2.127	1.000

3.3.3. Four-dimensional space

We also perform the same numerical experiments on the four-dimensional space. The initial condition is given as

$$\phi(x, y, z, u, 0) = \frac{1}{2}\left(1 + \tanh \frac{r - \sqrt{x^2 + y^2 + z^2 + u^2}}{2\sqrt{2}\epsilon}\right)$$

in computational domain $\Omega = (-2, 2) \times (-2, 2) \times (-2, 2) \times (-2, 2)$. The analytic solution of the zero level set of the AC equation, as ϵ converges to zero, is given by:

$$R(t) = \sqrt{R_0^2 - 6t},$$

which represents the analytic solution for the motion by mean curvature of a 3-dimensional sphere in 4-dimensional space. The parameters used in four-dimensional space are the same to two-and three-dimensional spaces except for the initial number of particles per point $m = 150$ and therefore initial number of particles $M = 118,560,000$. Because the initial radius is 0.8, the analytic solution to the motion by mean curvature of the AC equation is $R(t) = \sqrt{0.64 - 6t}$.

In Figure 9, we can observe that the numerical solution of the proposed method converges to the analytical solution when the temporal step Δt is decreased.

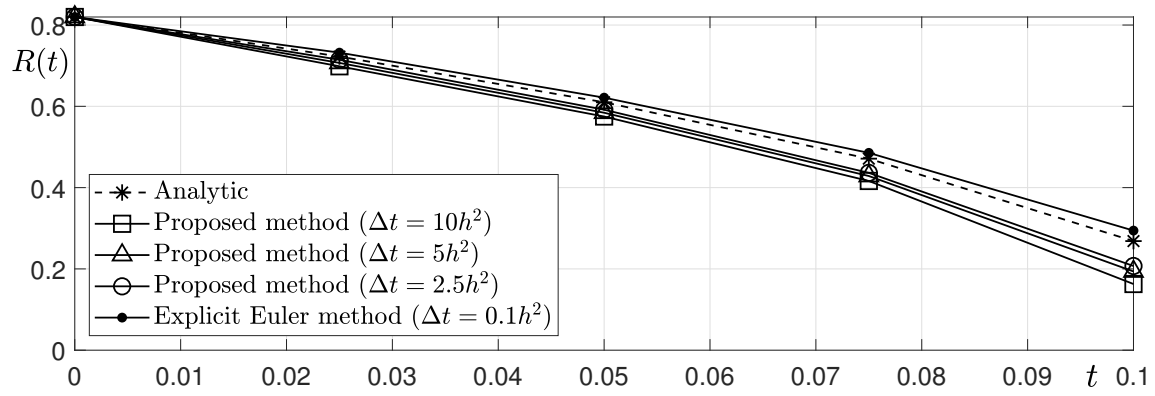


Figure 9. Temporal evolution of radius in four-dimensional space for the proposed method with different time steps and for the explicit Euler method.

Table 3 lists the CPU time to solve the AC equation up to total time $T = 0.1$ using the proposed method with different time steps and for the explicit Euler method in four-dimensional space and the

CPU time$_{Ratio}$ accordingly. Contrary to the numerical experiment on the computational speed in two-dimensional space, we can observe that the CPU time of the proposed method is faster than the CPU time of the explicit Euler method.

Table 3. CPU time to solve the AC equation in four-dimensional space up to total time $T = 0.1$ and the CPU time$_{Ratio}$ accordingly.

	Proposed method			Explicit Euler method
time step	$\Delta t = 10h^2$	$\Delta t = 5h^2$	$\Delta t = 2.5h^2$	$\Delta t = 0.1h^2$
CPU time (s)	760.154	865.778	1109.417	1962.491
CPU time$_{Ratio}$	0.387	0.441	0.565	1.000

Next, we perform numerical experiments to demonstrate that the proposed method can solve the AC equation even for a long time, provided that the total number of particles is sufficient. The initial condition is given as

$$\phi(x, y, z, u, 0) = \frac{1}{2}\left(1 + \tanh\frac{r - \sqrt{x^2 + y^2 + z^2 + u^2}}{2\sqrt{2}\epsilon}\right)$$

in computational domain $\Omega = (-3, 3) \times (-3, 3) \times (-3, 3) \times (-3, 3)$. The parameters used are $r = 2$, $T = 0.6$, $h = 0.1$, $\Delta t = 5h^2$, $\epsilon = \epsilon_{10}$ and $m = 70$. Consequently, the total initial number of particles was $M = 134,482,880$ according to the initial number of particles per point (m). Figure 10 shows the temporal evolution of the radius using the proposed method and the explicit Euler method for a long time.

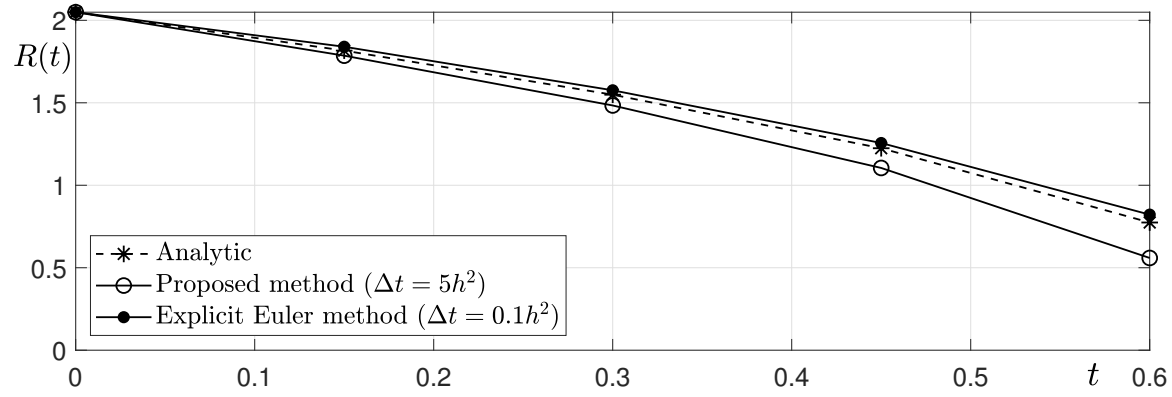


Figure 10. Temporal evolution of radius in four-dimensional space for a long time.

We observed that the proposed method solves the AC equation well when an adequate initial number of particles is provided.

3.4. Other nonlinear term

We consider the following Fisher–Kolmogorov–Petrovsky–Piscounov (Fisher–KPP) equation in two-dimensional space with $\phi(1-\phi)$ as an other nonlinear term [36, 37].

$$\frac{\partial \phi(x,y,t)}{\partial t} = \phi(x,y,t)(1-\phi(x,y,t)) + \Delta\phi(x,y,t). \tag{3.3}$$

We modify Eq (2.6) of the numerical algorithm to solve the nonlinear term of the Fisher–KPP equation using the proposed method.

$$\frac{d\Phi(t)}{dt} = \Phi(t)\,(1-\Phi(t)) \tag{3.4}$$

with an initial condition $\Phi(0) = \phi^*_{ij}$. Equation (3.4) is a logistic growth model and an analytic solution can be obtained [38]. The analytic solution of Eq (3.4) with the initial condition ϕ^*_{ij} after time Δt is given as

$$\phi^{**}_{ij} = \Phi(\Delta t) = \frac{\phi^*_{ij}}{\phi^*_{ij} + \left(1-\phi^*_{ij}\right)e^{-\Delta t}}.$$

To solve Eq (3.3) in the domain $(-20, 20)\times(-20, 20)$, we set the computational domain to $(-25, 25)\times(-25, 25)$ because the boundary must be considered for the initial condition.

$$\phi(x,y,0) = \frac{1}{4}\left[1 - \tanh\left(\frac{x}{2\sqrt{6}}\right)\right]^2.$$

The exact solution is given by

$$\phi(x,y,t) = \frac{1}{4}\left[1 - \tanh\left(\frac{x}{2\sqrt{6}} - \frac{5t}{12}\right)\right]^2.$$

For detailed information on the exact solution, please refer to [39]. The parameters used are $h = 2$, $T = 4$, $\Delta t = 0.01h^2$ and $m = 10,000$. We conduct numerical experiments by adjusting the standard value of 0.05, which provides particles at the beginning of the proposed algorithm, to 0.001 while considering smooth initial conditions. Figure 11 shows the temporal evolution of the numerical solution of the Fisher–KPP equation.

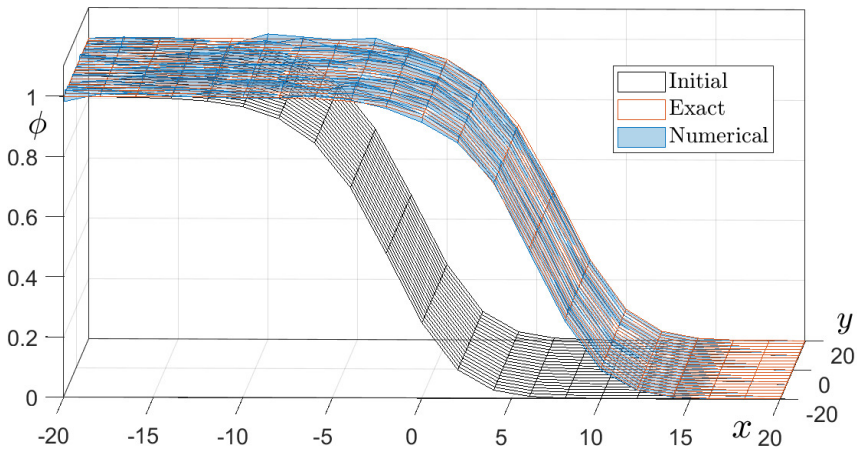


Figure 11. Numerical solution of the Fisher-KPP equation by the proposed method.

We observe that the proposed method can be applied to an other nonlinear equation with the nonlinear term which is different from $-F'(\phi)/\epsilon^2$.

4. Conclusions

In this paper, we presented an efficient and novel unconditionally stable MCS for solving the multi-dimensional AC equation that can model the motion by mean curvature flow of a hypersurface. We used an operator splitting method in which the diffusion and nonlinear terms are separately solved. The diffusion term is calculated using MCS for the stochastic differential equation and the nonlinear term is locally computed for each particle in the virtual grid. Therefore, the proposed method is unconditionally stable with respect to time, given an enough number of particles to solve the AC equation up to the total time. We performed numerical experiments in two-, three- and four-dimensional spaces to verify the performance of the proposed algorithm. The proposed algorithm is more efficient in high-dimensional space because the algorithm can be easily scaled despite the increase in the space dimension. Through numerical experiments, we observed that the proposed algorithm solves the AC equation more efficiently as the dimension of space increases.

Use of AI tools declaration

The authors have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

The corresponding author (J. S. Kim) expresses thanks for the support from the BK21 FOUR program. The authors express their gratitude to the reviewers for their valuable and insightful feedback on the revised version of this article.

Conflict of interest

The authors declare there are no conflicts of interest.

References

1. S. M. Allen, J. W. Cahn, A microscopic theory for antiphase boundary motion and its application to antiphase domain coarsening, *Acta Metall.*, **27** (1979), 1085–1095. https://doi.org/10.1016/0001-6160(79)90196-2
2. M. Olshanskii, X. Xu, V. Yushutin, A finite element method for Allen–Cahn equation on deforming surface, *Comput. Math. Appl.*, **90** (2021), 148–158. https://doi.org/10.1016/j.camwa.2021.03.018
3. J. W. Choi, H. G. Lee, D. Jeong, J. Kim, An unconditionally gradient stable numerical method for solving the Allen–Cahn equation, *Physica A*, **388** (2009), 1791–1803. https://doi.org/10.1016/j.physa.2009.01.026

4. Y. Li, H. G. Lee, D. Jeong, J. Kim, An unconditionally stable hybrid numerical method for solving the Allen–Cahn equation, *Comput. Math. Appl.*, **60** (2010), 1591–1606. https://doi.org/10.1016/j.camwa.2010.06.041
5. H. D. Vuijk, J. M. Brader, A. Sharma, Effect of anisotropic diffusion on spinodal decomposition, *Soft Matter*, **15** (2019), 1319–1326. https://doi.org/10.1039/C8SM02017E
6. T. H. Fan, J. Q. Li, B. Minatovicz, E. Soha, L. Sun, S. Patel, et al., Phase-field modeling of freeze concentration of protein solutions, *Polymers*, **11** (2018), 10. https://doi.org/10.3390/polym11010010
7. R. B. Marimont, M. B. Shapiro, Nearest neighbour searches and the curse of dimensionality, *IMA J. Appl. Math.*, **24** (1979), 59–70. https://doi.org/10.1093/imamat/24.1.59
8. X. Fang, L. Qiao, F. Zhang, F. Sun, Explore deep network for a class of fractional partial differential equations, *Chaos Solitons Fractals*, **172** (2023), 113528. https://doi.org/10.1016/j.chaos.2023.113528
9. V. Charles, J. Aparicio, J. Zhu, The curse of dimensionality of decision-making units: A simple approach to increase the discriminatory power of data envelopment analysis, *Eur. J. Oper. Res.*, **279** (2019), 929–940. https://doi.org/10.1016/j.ejor.2019.06.025
10. V. Berisha, C. Krantsevich, P. R. Hahn, S. Hahn, G. Dasarathy, P. Turaga, et al., Digital medicine and the curse of dimensionality, *npj Digit. Med.*, **4** (2021), 153. https://doi.org/10.1038/s41746-021-00521-5
11. S. Koohy, G. Yao, K. Rubasinghe, Numerical solutions to low and high-dimensional Allen–Cahn equations using stochastic differential equations and neural networks, *Partial Differ. Equations Appl. Math*, **7** (2023), 100499. https://doi.org/10.1016/j.padiff.2023.100499
12. S. Ham, Y. Hwang, S. Kwak, J. Kim, Unconditionally stable second-order accurate scheme for a parabolic sine–Gordon equation, *AIP Adv.*, **12** (2022), 025203. https://doi.org/10.1063/5.0081229
13. S. Ayub, H. Affan, A. Shah, Comparison of operator splitting schemes for the numerical solution of the Allen–Cahn equation, *AIP Adv.*, **9** (2019), 125202. https://doi.org/10.1063/1.5126651
14. H. G. Lee, A second-order operator splitting Fourier spectral method for fractional-in-space reaction-diffusion equations, *J. Comput. Appl. Math.*, **333** (2018), 395–403. https://doi.org/10.1016/j.cam.2017.09.007
15. D. Jeong, J. Kim, An explicit hybrid finite difference scheme for the Allen–Cahn equation, *J. Comput. Appl. Math.*, **340** (2018), 247–255. https://doi.org/10.1016/j.cam.2018.02.026
16. X. Yang, Z. Yang, C. Zhang, Stochastic heat equation: Numerical positivity and almost surely exponential stability, *Comput. Math. Appl.*, **119** (2022), 312–318. https://doi.org/10.1016/j.camwa.2022.05.031
17. Y. Sun, M. Kumar, Numerical solution of high dimensional stationary Fokker–Planck equations via tensor decomposition and Chebyshev spectral differentiation, *Comput. Math. Appl.*, **67** (2014), 1960–1977. https://doi.org/10.1016/j.camwa.2014.04.017
18. S. Shrestha, Monte carlo method to solve diffusion equation and error analysis, *J. Nepal Math. Soc.*, **4** (2021), 54–60. https://doi.org/10.3126/jnms.v4i1.37113

19. A. Medved, R. Davis, P. A. Vasquez, Understanding fluid dynamics from Langevin and Fokker–Planck equations, *Fluids*, **5** (2020), 40. https://doi.org/10.3390/fluids5010040

20. H. Naeimi, F. Kowsary, Finite Volume Monte Carlo (FVMC) method for the analysis of conduction heat transfer, *J. Braz. Soc. Mech. Sci. Eng.*, **41** (2019), 1–10. https://doi.org/10.1007/s40430-019-1762-3

21. H. Naeimi, Monte carlo methods for heat transfer, *Int. J. Math. Game Theory Algebra*, **29** (2020), 113–170.

22. T. Nakagawa, A. Tanaka, On a Monte Carlo scheme for some linear stochastic partial differential equations, *Monte Carlo Methods Appl.*, 27 (2021), 169–193. https://doi.org/10.1515/mcma-2021-2088

23. G. Venkiteswaran, M. Junk, Quasi-Monte Carlo algorithms for diffusion equations in high dimensions, *Monte Carlo Methods Appl.*, **68** (2005), 23–41. https://doi.org/10.1016/j.matcom.2004.09.003

24. A. Novikov, D. Kuzmin, O. Ahmadi, Random walk methods for Monte Carlo simulations of Brownian diffusion on a sphere, *Appl. Math. Comput.*, **364**, (2020), 124670. https://doi.org/10.1016/j.amc.2019.124670

25. D. Lee, The numerical solutions for the energy-dissipative and mass-conservative Allen–Cahn equation, *Comput. Math. Appl.*, **80** (2020), 263–284. https://doi.org/10.1016/j.camwa.2020.04.007

26. H. G. Lee, J. Y. Lee, A second order operator splitting method for Allen–Cahn type equations with nonlinear source terms, *Physica A*, **432** (2015), 24–34. https://doi.org/10.1016/j.physa.2015.03.012

27. Y. Cheng, A. Kurganov, Z. Qu, T. Tang, Fast and stable explicit operator splitting methods for phase-field models, *J. Comput. Phys.*, **303** (2015), 45–65. https://doi.org/10.1016/j.jcp.2015.09.005

28. A. Chertock, C. R. Doering, E. Kashdan, A. Kurganov, A fast explicit operator splitting method for passive scalar advection, *J. Sci. Comput.*, **45** (2010), 200–214. https://doi.org/10.1007/s10915-010-9381-2

29. K. Poochinapan, B. Wongsaijai, Numerical analysis for solving Allen–Cahn equation in 1D and 2D based on higher-order compact structure-preserving difference scheme, *Appl. Math. Comput.*, **434** (2022), 127374. https://doi.org/10.1016/j.amc.2022.127374

30. J. Kai, S. Wei, High-order energy stable schemes of incommensurate phase-field crystal model, *Electron. Res. Arch.*, **28** (2020), 1077–1093. https://doi.org/10.3934/era.2020059

31. W. Liupeng, H. Yunqing, Error estimates for second-order SAV finite element method to phase field crystal model, *Electron. Res. Arch.*, **29** (2021), 1735–1752. https://doi.org/10.3934/era.2020089

32. J. W. Choi, H. G. Lee, D. Jeong, J. Kim, An unconditionally gradient stable numerical method for solving the Allen–Cahn equation, *Physica A*, **388** (2009), 1791–1803. https://doi.org/10.1016/j.physa.2009.01.026

33. M. Franken, M. Rumpf, B. Wirth, A phase field based PDE constraint optimization approach to time discrete Willmore flow, *Int. J. Numer. Anal. Model.*, **2011** (2011).

34. X. Chen, C. M. Elliott, A. Gardiner, J. JING ZHAO, Convergence of numerical solutions to the Allen–Cahn equation, *Appl. Anal.*, **69** (1998), 47–56.

35. G. B. Folland, *Introduction to Partial Differential Equations*, Princeton university press, 1976. https://doi.org/10.1515/9780691213033

36. R. A. Fisher, The wave of advance of advantageous genes, *Ann. Eugen.*, **7** (1937), 355–369. https://doi.org/10.1111/j.1469-1809.1937.tb02153.x

37. A. Kolmogorov, Étude de l'équation de la diffusion avec croissance de la quantité de matière et son application à un problème biologique, *Moscow Univ. Bull. Math.*, **1** (1937), 1–25.

38. P. Román-Román, F. Torres-Ruiz, Modelling logistic growth by a new diffusion process: application to biological systems, *Biosystems*, **110** (2012), 9–21. https://doi.org/10.1016/j.biosystems.2012.06.004

39. X. Y. Wang, Exact and explicit solitary wave solutions for the generalised fisher equation, *Phys. Lett. A*, **131** (1988), 277–279. https://doi.org/10.1016/0375-9601(88)90027-8

AIMS Press