Doctoral Dissertation

Various numerical schemes and analysis for the conservative Allen–Cahn equation

Soobin Kwak

Department of Mathematics

Graduate School

Korea University

August 2024

Various numerical schemes and analysis for the conservative Allen–Cahn equation

by
Soobin Kwak

---

under the supervision of Professor Junseok Kim

A dissertation submitted in partial fulfillment of
the requirements for the degree of
Doctor of Philosophy

Department of Mathematics

Graduate School
Korea University

April 2024

The dissertation of Soobin Kwak has been approved by the dissertation committee in partial fulfillment of the requirements for the degree of Doctor of Philosophy

June 2024

Committee Chair: Junseok Kim

Committee Member: Seunggyu Lee

Committee Member: Hyun Geun Lee

Committee Member: Darae Jeong

Committee Member: Yongho Choi

Various numerical schemes and analysis for the conservative Allen–Cahn equation

by Soobin Kwak

Department of Mathematics

under the supervision of Professor Junseok Kim

Abstract

In this dissertation, we introduce the Allen–Cahn (AC) equation and examine its various properties. We introduce the mass conservative Allen–Cahn (CAC) equation to compensate for the property of the AC equation not conserving mass. We briefly review two well-known CAC equations and propose a new CAC equation that employs curvature-dependent Lagrange multipliers. Furthermore, we provide solutions for the three CAC equations, including the newly proposed CAC equation, using various numerical methods based on the operator splitting method. The presented novel CAC equation exhibits an enhanced capability to preserve features. In contrast to traditional CAC equations, which typically involve motion by mean curvature under area or volume constraints, the proposed model minimizes the dynamics of such curvature motion and focuses solely on smoothing the transition layers of interfaces. As a result, it is well-suited as a foundational equation for modeling conservative phase-field scenarios, including two-phase fluid flows. Various computational tests have validated the superior feature-preservation characteristics of the proposed CAC equation.

Keywords: Conservative Allen–Cahn equation, Feature preserving property, Spectral

method, Alternating direction explicit method, Runge–Kutta method

보존적 알렌–칸 방정식을 위한 다양한 수치 기법및 분석

곽 수 빈

수 학 과

지 도 교 수: 김 준 석

국문 초록

본 논문에서는 알렌–칸 방정식에 대하여 소개하고 다양한 속성을 살펴봅니다. 질량을 보존하지 않는 알렌–칸 방정식의 특성을 보완하기 위해 질량 보존적 알렌–칸 방정식을 도입합니다. 우리는 잘 알려진 두 가지 보존적 알렌–칸 방정식을 간략하게 살펴보고 곡률 종속 라그랑주 승수를 사용하는 새로운 보존적 알렌–칸 방정식을 제안합니다. 또한, 새롭게 제안된 보존적 알렌–칸 방정식을 포함한 세 가지 보존적 알렌–칸 방정식에 대해 연산자 분할 방법을 기반으로 다양한 수치 방법을 사용하여 해를 제공합니다. 제안된 보존적 알렌–칸 방정식은 우수한 구조 보존 특성을 가지고 있습니다. 면적이나 부피 제약이 있는 평균 곡률에 의한 운동을 갖는 기존 보존적 알렌–칸 방정식과 달리 제안 모델은 평균 곡률에 의한 운동의 동역학을 최소화하고 인터페이스 전이층의 평활화 특성만 갖습니다. 따라서 이는 2상 유체 흐름과 같은 보존적인 상태장 응용을 모델링하기 위한 빌딩 블록 방정식으로 활용될 수 있습니다. 구조 보존 특성 측면에서 제안된 보존적 알렌–칸 방정식의 우수한 성능을 확인하기 위해 여러 수치 계산 테스트가 수행되었습니다.

중심어: 보존적 알렌–칸 방정식, 구조 보존 속성, 스펙트럴 방법, 교대 방향 명시적 방법, 룽게–쿠타 방법

You always said you wanted to see me go to college, but you left us too soon.

I believe you would be the proudest of me in the world.

I dedicate this dissertation to my beloved grandmother.

Preface

This dissertation is based on the work published in *Applied Mathematics Letters*, which has played a pivotal role in shaping the scope and direction of this dissertation.

- S. Kwak, J. Yang, and J. Kim, A conservative Allen–Cahn equation with a curvature-dependent Lagrange multiplier, Applied Mathematics Letters, 126 (2022) 107838.

This dissertation stands as a testament to the combined efforts and unwavering support of many individuals, to whom I am deeply thankful.

Acknowledgment

This dissertation owes its completion to the support and encouragement of numerous individuals. I would like to take this opportunity to express my deepest gratitude to all those who have contributed to the completion of this work.

First and foremost, I would like to thank my advisor, Professor Junseok Kim, for their invaluable guidance, continuous support, and patience throughout my research.

I extend my sincere thanks to my dissertation committee members, Hyun Geun Lee, Darae Jeong, Yongho Choi, and Seunggyu Lee, for their time, effort, and constructive feedback. Their expertise and suggestions have greatly improved the quality of my work.

I am also grateful to my colleagues and friends in the CFD Lab at Korea University.

A heartfelt thank you goes to my family for their unwavering love and support. To my parents, Hyun Sook Cho and Dong Gon Kwak, who have always believed in me and encouraged me to dream, and to my best friend for their understanding, patience, and constant encouragement, thank you. Your love and support have been my greatest strength. I also want to thank my one and only sibling, Soomin Kwak. I will always strive to be a reliable sister you can depend on.

Lastly, I dedicate this dissertation to my late grandmother, who always wished to see this day. I believe you would be the proudest of me in the world.

Thank you all for your unwavering support and belief in me.

Table of Contents

List of Tables

List of Figures

Nomenclature

ϕ	phase
t	time
M	mobility coefficient
F	Helmholtz free energy
κ	curvature
$\mathscr{E}$	Ginzburg–Lindau free energy
Ω	domain
$\partial\Omega$	boundary of Ω

Abbreviation

AC	Allen–Cahn
PDE	Partial differential equaiton
CAC	Conservative Allen–Cahn

CH	Cahn–Hilliard
OSM	Operator splitting method
ADE	Alternating direction explicit
RKF	Rung–Kutta–Fehlberg

Chapter 1. Introduction

The Allen–Cahn (AC) equation [1] is a model used to represent the phase-field. This partial differential equation (PDE) describes the phase separation phenomenon in a binary alloy system. It was introduced by S.M. Allen and J.W. Cahn. The AC equation continues to be actively researched in various fields such as data classification [2, 3, 4, 5], image segmentation [6, 7, 8, 9], image inpainting [10, 11, 12, 13, 14], shape transformation [15, 16, 17], crystal growth [18, 19, 20, 21, 22], ice melting [23], etc. [24, 25, 26].

The classical AC equation is given by:

$$\frac{\partial \phi(\mathbf{x},t)}{\partial t} = -M\left(\frac{F'(\phi(\mathbf{x},t))}{\varepsilon^2} - \Delta\phi(\mathbf{x},t)\right), \qquad \mathbf{x}\in\Omega,\ t\geq 0, \tag{1.1}$$

$$\mathbf{n}\cdot\nabla\phi(\mathbf{x},t) = 0, \qquad \mathbf{x}\in\partial\Omega. \tag{1.2}$$

Here, $\Omega\subset\mathbb{R}^d$ ($d=1,\ 2,\ldots$) is a bounded domain, t is time, $M(\phi)$ is the mobility coefficient [53] (in this dissertation, we take $M=1$ for simplicity), $\mathbf{n}$ is the unit normal vector on $\partial\Omega$, and $\Delta\phi=\nabla\cdot\nabla\phi$ is the Laplacian of ϕ.

The order parameter $\phi(\mathbf{x},t)\in[-1,1]$, represents the concentration of one of the two components in a binary mixture and is defined as follows:

$$\phi = \frac{m_1 - m_2}{m_1 + m_2}, \tag{1.3}$$

where m_1 and m_2 are the masses of the components 1 and 2. In other words, $\phi = -1$ and $\phi = 1$ represent different phases. ε is positive parameter related to interface thickness. Figure 1.1 visually illustrates the interface thickness. For further information on the

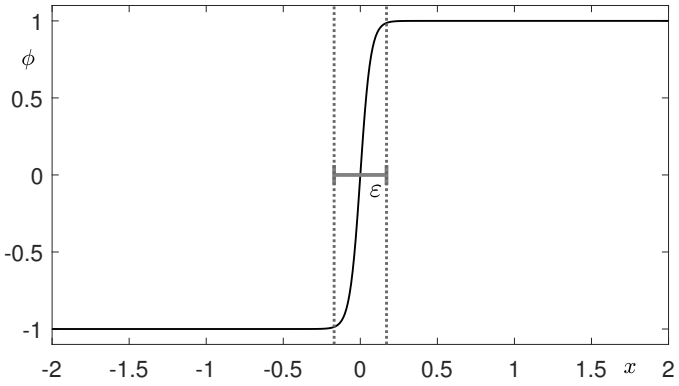


Figure 1.1: Schematic illustration of interface thickness.

relationship between the value of ε and the width of the transition layer, refer to [27].

The commonly used double well potential energies are logarithmic potential and polynomial potential. A logarithmic potential F_l can be described as:

$$F_l(\phi) = \frac{\theta}{2}\left[(1+\phi)\ln\left(\frac{1+\phi}{2}\right) + (1-\phi)\ln\left(\frac{1-\phi}{2}\right)\right] + \frac{\tilde{\theta}}{2}(1-\phi^2).$$

Here, θ refers to the absolute temperature, while $\tilde{\theta}$ corresponds to the critical temperature. However, as shown in Fig. 1.2, using logarithmic potential energy leads to singularities at $\phi = -1$ and $\phi = 1$. To avoid this issue, we use polynomial potential energy. The Helmholtz free energy, given by $F(\phi) = 0.25(\phi^2 - 1)^2$, represents the chemical potential energy (refer to Fig. 1.3).

The AC equation is derived by the L^2-gradient flow of the Ginzburg–Landau free

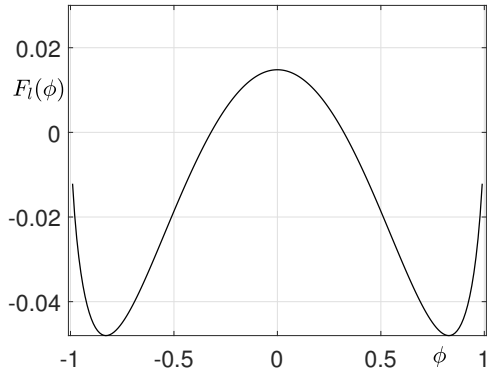


Figure 1.2: The logarithmic potential energy function $F_l(\phi)$.

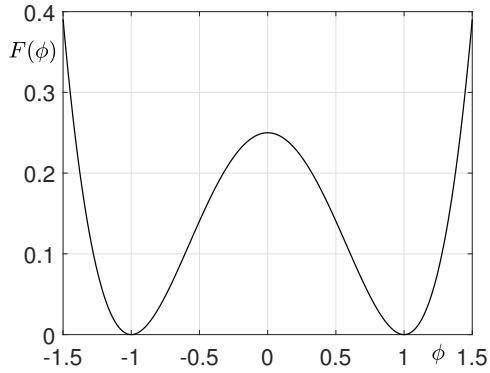


Figure 1.3: Double-well potential energy function $F(\phi)$.

energy functional, which is given by:

$$\mathscr{E}(\phi) := \int_\Omega \left(\frac{F(\phi)}{\varepsilon^2} + \frac{|\nabla\phi|^2}{2} \right) d\mathbf{x}. \tag{1.4}$$

To derive the AC equation, we take the variational derivative on $\mathscr{E}(\phi)$,

$$\begin{aligned}\frac{\delta\mathscr{E}(\phi)}{\delta\phi} &= \lim_{\zeta\to 0}\frac{\mathscr{E}(\phi+\zeta\psi)-\mathscr{E}(\phi)}{\zeta}\\ &= \int_\Omega \frac{\delta\mathscr{E}}{\delta\phi}\psi\, d\mathbf{x}\\ &= \int_\Omega \left(\frac{F'(\phi)}{\varepsilon^2}-\Delta\phi\right)\psi\, d\mathbf{x}.\end{aligned}$$

We can obtain AC equation as following:

$$\phi_t = -\frac{\delta\mathscr{E}}{\delta\phi} = -\frac{F'(\phi)}{\varepsilon^2}+\Delta\phi.$$

We briefly examine some of the key properties of the AC equation. First, an interesting physical and mathematical problem is determining the motion of this antiphase boundary. In [28], it is demonstrated that the interface moves with a normal velocity proportional to its mean curvature, as discussed in [1, 29]. Let $r = r(x,\ y,\ z,\ t)$ represent the signed distance from $\phi = 0$ for the point $(x,\ y,\ z)$. Here, $r < 0$ when $\phi(x,\ y,\ z,\ t) > 0$ and

$r > 0$ when $\phi(x,\ y,\ z,\ t) < 0$. Now, the term $\Delta\phi$ can be rewritten in the following:

$$\begin{aligned}
\Delta\phi &= \nabla\cdot\nabla\phi \\
&= \nabla\cdot(|\nabla\phi|\mathbf{n}) \\
&= \nabla\cdot((\nabla\phi\cdot\mathbf{n})\mathbf{n}) \\
&= \nabla\cdot(-\phi_r\mathbf{n}) \\
&= -\nabla\phi_r\cdot\mathbf{n} - \phi_r\nabla\cdot\mathbf{n} \\
&= -(\nabla\phi)_r\cdot\mathbf{n} - \phi_r\nabla\cdot\mathbf{n} \\
&= (\phi_r\mathbf{n})_r\cdot\mathbf{n} - \phi_r\nabla\cdot\mathbf{n} \\
&= (\phi_{rr}\mathbf{n} + \phi_r\mathbf{n}_r)\cdot\mathbf{n} - \phi_r\nabla\cdot\mathbf{n} \\
&= \phi_{rr} + (\kappa_1 + \kappa_2),
\end{aligned}$$

where κ_1 and κ_2 represent the principal curvatures of the surface, and considering that the divergence of the unit normal vector to a surface equals the negative of the mean curvature $(\kappa_1 + \kappa_2)$, the previous equality is valid. Consequently, this can be applied to the kinetic equation:

$$\phi_t = -\frac{F'(\phi)}{\varepsilon^2} + \phi_{rr} + (\kappa_1 + \kappa_2)\phi_r. \tag{1.5}$$

For a planar interface at equilibrium, the equation below is applicable:

$$-\frac{F'(\phi)}{\varepsilon^2} + \phi_{rr} \approx 0. \tag{1.6}$$

Thus, Eq. (1.5) can be reformulated as:

$$\phi_t = (\kappa_1 + \kappa_2)\phi_r. \tag{1.7}$$

At any given time, the zero level-set is represented as $\Gamma_t = \{(x,\ y,\ z) | \phi(x,\ y,\ z,\ t) = 0\}$. The velocity of this zero level-set Γ_t can then be described as follows:

$$\begin{aligned} 0 &= \left.\frac{d\phi(r,\ t)}{dt}\right|_{\Gamma_t} = \phi_r r_t + \phi_t \\ \Rightarrow r_t &= -\frac{\phi_t}{\phi_r} = -(\kappa_1 + \kappa_2). \end{aligned}$$

Thus, all interfaces between two phases move at a velocity denoted by V, which is defined as follows:

$$V = -(\kappa_1 + \kappa_2) = -\left(\frac{1}{R_1} + \frac{1}{R_2}\right),$$

where R_1 and R_2 represent the principal radii of curvature at a point on the surface.

In Fig. 1.4, the dashed line represents the initial condition, while the solid line shows the state after time evolution. As shown in Fig. 1.4, under time evolution, the AC equation causes the interface to move in the direction of the mean curvature flow.

Another notable characteristic of the AC equation is energy dissipation [30]. Differ-

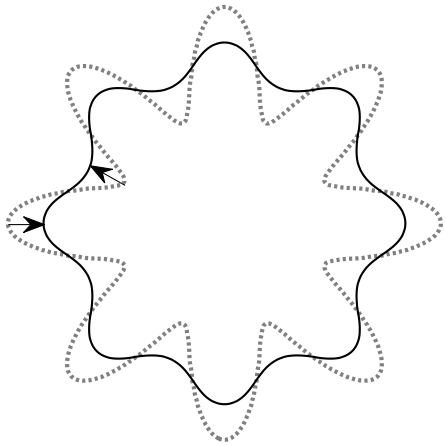


Figure 1.4: Schematic illustration of motion by mean curvature.

entiating the total energy $\mathscr{E}$ with respect to time t obtains the following result.

$$\begin{aligned}
\frac{d}{dt}\mathscr{E}(\phi) &= \int_\Omega \left(\frac{\partial \phi}{\partial t} \frac{F'(\phi)}{\varepsilon^2} + \nabla \frac{\partial \phi}{\partial t} \cdot \nabla \phi \right) d\mathbf{x} \\
&= \int_\Omega \phi_t \left(\frac{F'(\phi)}{\varepsilon^2} - \Delta \phi \right) d\mathbf{x} \\
&= -\int_\Omega (\phi_t)^2 \, d\mathbf{x} \\
&\leq 0.
\end{aligned}$$

Therefore, it is noted that the energy does not increase over time.

Finally, the AC equation does not conserve mass. An intrinsic characteristic of the AC equation (4.6) is its non-conservation of mass as time evolves, a property that can be

verified through analytical examination.

$$\begin{aligned}
\frac{d}{dt}\int_\Omega \phi(\mathbf{x},t)\,d\mathbf{x} &= \int_\Omega \phi_t\,d\mathbf{x} \\
&= \int_\Omega \left(-\frac{F'(\phi(\mathbf{x},t))}{\varepsilon^2}+\Delta\phi(\mathbf{x},t)\right)\,d\mathbf{x} \\
&= -\int_\Omega \frac{F'(\phi(\mathbf{x},t))}{\varepsilon^2}\,d\mathbf{x}+\int_{\partial\Omega}\mathbf{n}\cdot\nabla\phi(\mathbf{x},t)\,ds \\
&= -\int_\Omega \frac{F'(\phi(\mathbf{x},t))}{\varepsilon^2}\,d\mathbf{x} \\
&\neq 0.
\end{aligned}$$

Therefore, since the derivative of mass with respect to time t is not always zero, mass is not conserved. Intuitively, as shown in Fig. 1.5, we can observe that the time evolution of an arbitrary initial condition in the AC equation does not conserve mass.

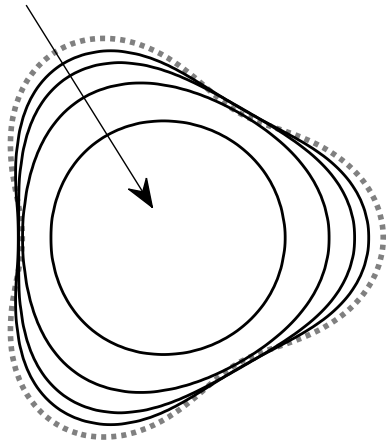


Figure 1.5: Schematic illustration of the non-conservation of mass in the AC equation. the initial condition is represented by a dotted line.

Consequently, the study of conservative Allen–Cahn (CAC) equation, which preserves mass, is popular among many researchers. The Cahn–Hilliard (CH) equation, which is derived from the Ginzburg–Landau free energy functional, also represents phase

separation and conserves the total mass. Nevertheless, the increasing adoption of the CAC equation over the CH equation can be attributed to its simplicity, enhanced computational efficiency, and superior accuracy [31]. A significant difference between these equations lies in their order: the CAC equation is a second-order PDE, whereas the CH equation is a more complex fourth-order PDE. This distinction underscores the advantage of solving lower-order PDEs, which generally leads to more efficient computational implementations and facilitates practical applications across a wide range of scientific and engineering fields.

This doctoral dissertation presents the concepts and outcomes featured in the following publication, reflecting the author's dedication throughout both the master and doctoral courses.

- **S. Kwak**, J. Yang, and J. Kim, A conservative Allen–Cahn equation with a curvature-dependent Lagrange multiplier, Applied Mathematics Letters, 126 (2022) 107838.

This dissertation is organized as follows. In Chapter 2, we examine two widely used CAC equations and introduce a novel CAC equation that incorporates a curvature-dependent Lagrange multiplier. Chapter 3 presents numerical solutions for these three CAC equations, utilizing various numerical methods and discrete curvature calculation techniques. Chapter 4 details the numerical results obtained from the three CAC equations, providing a comparative analysis with each other and with the results from the CH equation. Finally, Chapter 5 summarizes our conclusions and insights.

Chapter 2. Conservative Allen–Cahn equation

In this chapter, we present the two most well-known CAC equations and propose a novel CAC equation [32]. Each of these equations is a modification of the AC equation, enhanced by the addition of distinct Lagrange multipliers.

2.1 The time-dependent Lagrange multiplier

The first of the two widely known CAC equations that were introduced is the AC equation with a time-dependent Lagrange multiplier. The formulation of the time-dependent CAC equation is presented as follows [38, 39]:

$$\frac{\partial \phi(\mathbf{x},t)}{\partial t} = -\frac{F'(\phi(\mathbf{x},t))}{\varepsilon^2} + \Delta\phi(\mathbf{x},t) + \mathscr{L}(t), \tag{2.1}$$

where $\mathscr{L}(t)$ represents the time-dependent Lagrange multiplier defined as

$$\mathscr{L}(t) = \frac{\int_\Omega F'(\phi(\mathbf{x},t))\ d\mathbf{x}}{\varepsilon^2 \int_\Omega d\mathbf{x}}.$$

Note that if $\mathscr{L}(t)$ is not included, then Eq. (2.1) simplifies to the classical AC equation [1], widely employed for modeling phase transformations in binary mixtures and the dynamics of antiphase boundaries [40, 41, 42, 43]. We rigorously analyzed and verified the inherent mass conservation property of Eq. (2.1), ensuring its validity across various conditions and scenarios.

$$\begin{aligned}
\frac{d}{dt}\int_\Omega \phi(\mathbf{x},t)\,d\mathbf{x} &= \int_\Omega \frac{\partial \phi}{\partial t}\,d\mathbf{x} \\
&= \int_\Omega \left(-\frac{F'(\phi(\mathbf{x},t))}{\varepsilon^2} + \Delta\phi(\mathbf{x},t) + \mathscr{L}(t)\right) d\mathbf{x} \\
&= -\int_\Omega \frac{F'(\phi(\mathbf{x},t))}{\varepsilon^2}\,d\mathbf{x} + \int_{\partial\Omega} \mathbf{n}\cdot\nabla\phi(\mathbf{x},t)\,ds + \int_\Omega \mathscr{L}(t)\,d\mathbf{x} \\
&= -\frac{1}{\varepsilon^2}\int_\Omega F'(\phi(\mathbf{x},t))\,d\mathbf{x} + \mathscr{L}(t)\int_\Omega d\mathbf{x} \\
&= 0
\end{aligned}$$

Hence, the time-dependent CAC Equation (2.1) ensures mass conservation over time.

To aid in understanding the characteristics of time-dependent CAC Equation, Fig. 2.1 visually illustrates the time evolution results of the time-dependent CAC equation. Consider the initial condition provided in Fig. 2.1(a). When solving the time-dependent CAC equation, as shown in Fig. 2.1(b), the mass is conserved. However, due to the time-dependent Lagrange multiplier $\mathscr{L}(t)$ adding a constant value across the entire computational domain, relatively small features with high curvature in the initial condition are lost.

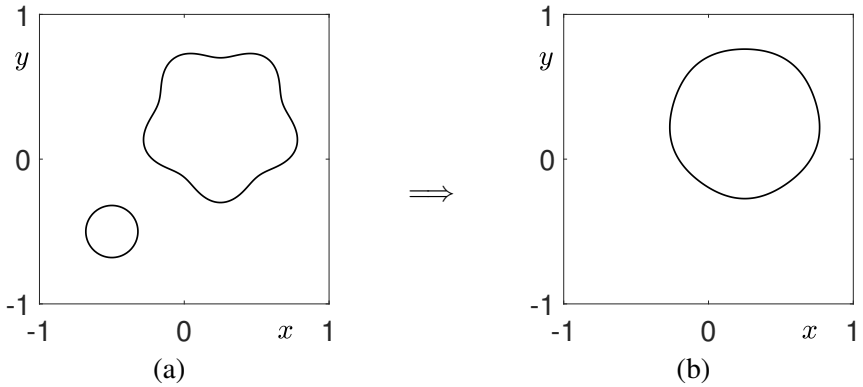


Figure 2.1: Schematic illustration of temporal evolution of the time-dependent CAC equation (2.1). Zero-level contours of (a) the initial condition and (b) the result of temporal evolution.

2.2 The time- and space-dependent Lagrange multiplier

The second of the two most popular CAC equations introduced incorporates a time- and space-dependent Lagrange multiplier. This time- and space-dependent CAC equation is expressed as follows [44, 45, 46, 47, 48]:

$$\frac{\partial \phi(\mathbf{x},t)}{\partial t} = -\frac{F'(\phi(\mathbf{x},t))}{\varepsilon^2} + \Delta\phi(\mathbf{x},t) + \mathscr{L}(t)\sqrt{2F(\phi(\mathbf{x},t))}, \tag{2.2}$$

where $\mathscr{L}(t)\sqrt{2F(\phi(\mathbf{x},t))}$ serves as the time- and space-dependent Lagrange multiplier, given by

$$\mathscr{L}(t) = \frac{\int_\Omega F'(\phi(\mathbf{x},t))\ d\mathbf{x}}{\varepsilon^2 \int_\Omega \sqrt{2F(\phi(\mathbf{x},t))}\ d\mathbf{x}}.$$

We analytically verified the mass conservation property of Eq. (2.2) as follows.

$$\begin{aligned}
\frac{d}{dt}\int_\Omega \phi(\mathbf{x},t)\,d\mathbf{x} &= \int_\Omega \frac{\partial \phi}{\partial t}\,d\mathbf{x} \\
&= \int_\Omega \left(-\frac{F'(\phi(\mathbf{x},t))}{\varepsilon^2} + \Delta\phi(\mathbf{x},t) + \mathscr{L}(t)\sqrt{2F(\phi(\mathbf{x},t))} \right) d\mathbf{x} \\
&= -\int_\Omega \frac{F'(\phi(\mathbf{x},t))}{\varepsilon^2}\,d\mathbf{x} + \int_{\partial\Omega} \mathbf{n}\cdot\nabla\phi(\mathbf{x},t)\,ds \\
&\quad + \int_\Omega \mathscr{L}(t)\sqrt{2F(\phi(\mathbf{x},t))}\,d\mathbf{x} \\
&= -\frac{1}{\varepsilon^2}\int_\Omega F'(\phi(\mathbf{x},t))\,d\mathbf{x} + \mathscr{L}(t)\int_\Omega \sqrt{2F(\phi(\mathbf{x},t))}\,d\mathbf{x} \\
&= 0
\end{aligned}$$

Therefore, time- and space-dependent CAC Equation (2.2) also conserves mass over time.

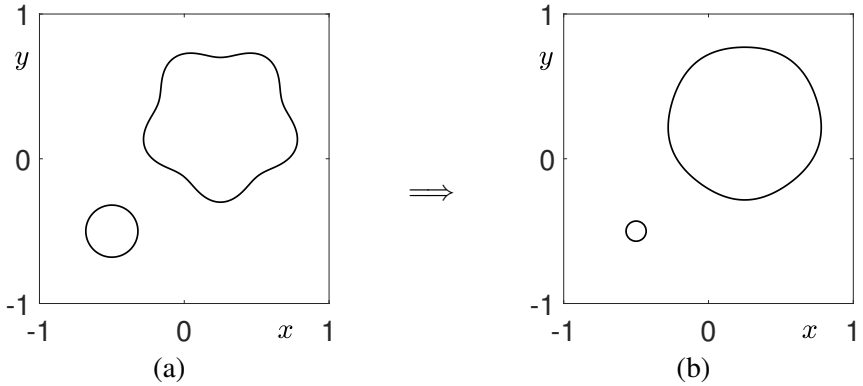


Figure 2.2: Schematic illustration of temporal evolution of the time- and space-dependent CAC equation (2.2). Zero-level contours of (a) the initial condition and (b) the result of temporal evolution.

Figure 2.2 demonstrates the characteristics of the time- and space-dependent CAC equation (2.2). Given the initial condition shown in Fig. 2.2(a), the time- and space-dependent CAC equation has a Lagrange multiplier $\mathscr{L}(t)\sqrt{2F(\phi(\mathbf{x},t))}$ that is dependent

on space. As a result, compared to the outcomes of the time-dependent CAC equation (2.1), the time- and space-dependent CAC equation (2.2) better preserves small features, as shown in Fig. 2.2(b). However, as time evolution progresses further, the small features eventually disappear due to motion by mean curvature.

2.3 The curvature-dependent Lagrange multiplier

In [49], the author proposed two numerical methods designed to conserve mass for the AC equation. These methods operate within the mass-conserving space and employ mass-projection techniques alongside energy-dissipation operators. In [50], the author devised a finite difference method tailored for a conservative CAC equation, ensuring both stability and preservation of features. In [51], the authors presented a new CAC equation. They rigorously demonstrated the existence, uniqueness, and boundedness properties of the solution to this equation. The CAC equation with a Lagrange multiplier that varies with time and space has been effectively employed in the study [47, 52] of multiphase fluid flows. However, conventional CAC models inherently exhibit motion by mean curvature with constraints. Consequently, in scenarios where two droplets of distinct sizes are present, the smaller droplet is gradually absorbed by the larger one and ultimately vanishes. This characteristic significantly limits the applicability of these models to scenarios such as two-phase fluid dynamics involving multiple components of varying sizes. To address these challenges, we introduce a novel CAC equation that includes a Lagrange multiplier dependent on curvature, that ensures robust feature-preserving properties. The proposed model minimizes motion dynamics through mean curvature while emphasizing interface smoothing properties. Consequently, it can serve as a foundational equation for modeling conservative phase-field applications, par-

ticularly in scenarios like two-phase fluid flows, where maintaining distinct phases of various sizes is crucial. This advancement enhances the versatility and applicability of CAC models in practical, complex fluid dynamics scenarios.

$$\frac{\partial \phi(\mathbf{x},t)}{\partial t} = -M\left(\frac{F'(\phi(\mathbf{x},t))}{\varepsilon^2} - \Delta\phi(\mathbf{x},t) - \mathscr{L}(t)\kappa(\phi(\mathbf{x},t))\sqrt{2F(\phi(\mathbf{x},t))}\right), \quad (2.3)$$

where M denotes the mobility coefficient [53], $\kappa(\phi) = \nabla\cdot(\nabla\phi/|\nabla\phi|)$ represents the interface curvature, and

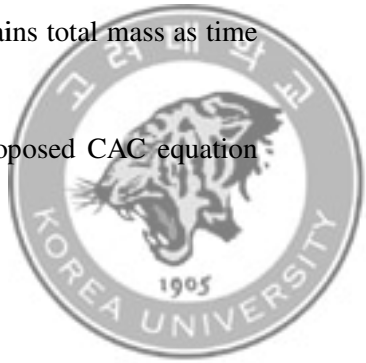
$$\mathscr{L}(t) = \frac{\int_\Omega F'(\phi(\mathbf{x},t))\,d\mathbf{x}}{\varepsilon^2\int_\Omega \kappa(\phi(\mathbf{x},t))\sqrt{2F(\phi(\mathbf{x},t))}\,d\mathbf{x}}.$$

As a result, the solution ϕ of Eq. (2.3) satisfies the following equation:

$$\begin{aligned}
\frac{d}{dt}\int_\Omega \phi\,d\mathbf{x} &= \int_\Omega \frac{\partial\phi(\mathbf{x},t)}{\partial t}\,d\mathbf{x} \\
&= \int_\Omega \left[-\frac{F'(\phi(\mathbf{x},t))}{\varepsilon^2} + \Delta\phi(\mathbf{x},t) + \mathscr{L}(t)\kappa(\phi(\mathbf{x},t))\sqrt{2F(\phi(\mathbf{x},t))}\right]\,d\mathbf{x} \\
&= -\frac{1}{\varepsilon^2}\int_\Omega F'(\phi(\mathbf{x},t))\,d\mathbf{x} + \int_{\partial\Omega}\mathbf{n}\cdot\nabla\phi(\mathbf{x},t)\,ds \\
&\quad + \mathscr{L}(t)\int_\Omega \kappa(\phi(\mathbf{x},t))\sqrt{2F(\phi(\mathbf{x},t))}\,d\mathbf{x} \\
&= -\frac{1}{\varepsilon^2}\int_\Omega F'(\phi(\mathbf{x},t))\,d\mathbf{x} + \mathscr{L}(t)\int_\Omega \kappa(\phi(\mathbf{x},t))\sqrt{2F(\phi(\mathbf{x},t))}\,d\mathbf{x} \\
&= 0. \qquad (2.4)
\end{aligned}$$

We confirmed that the proposed novel CAC equation (2.3) maintains total mass as time evolution progresses.

Figure illustrates the time evolution results of the newly proposed CAC equation



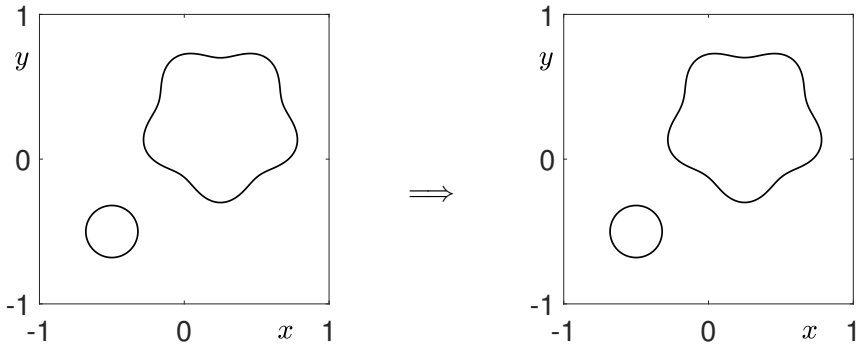


Figure 2.3: Schematic illustration of temporal evolution of the proposed new CAC equation (2.3). Zero-level contours of (a) the initial condition and (b) the result of temporal evolution.

(2.3). When compared to the results of the time-dependent CAC (2.1) and the time- and space-dependent CAC (2.2), under the same initial condition, it is evident that the newly proposed CAC equation (2.3) not only preserves small features effectively but also maintains the original interface features without adhering to mean curvature flow.

Chapter 3. Numerical solution algorithm

Before proceeding with discretization, we apply the operator splitting method (OSM) to solve the proposed CAC equation (2.3).

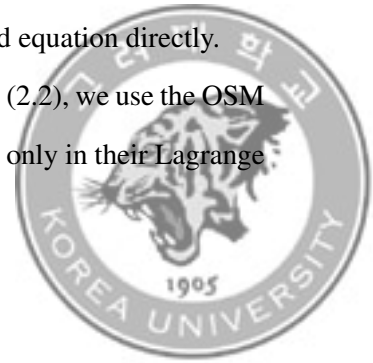
$$\phi_t(\mathbf{x},t) = \Delta\phi(\mathbf{x},t), \tag{3.1}$$

$$\phi_t(\mathbf{x},t) = -\frac{F'(\phi(\mathbf{x},t))}{\varepsilon^2}, \tag{3.2}$$

$$\phi_t(\mathbf{x},t) = \mathscr{L}(t)\kappa(\phi(\mathbf{x},t))\sqrt{2F(\phi(\mathbf{x},t))}. \tag{3.3}$$

The OSM is a computational technique employed to solve differential equations by decomposing the original equation into separate operators over discrete time step [55]. This method involves independently solving each component and then integrating the results to construct the complete solution. A classic instance of this method involves the separation of terms related to diffusion and advection within a convection-diffusion PDE. This method is also notably applied to reaction-diffusion PDEs in fields such as chemistry and biology. The concept of OSM extends naturally to equations that involve more than two operators. The primary computational benefit of this approach is that solving each component separately tends to be faster than solving the combined equation directly.

To compare the results of Eq. (2.3) with those of Eqs. (2.1) and (2.2), we use the OSM to solve Eqs. (2.1) and (2.2). Since the three CAC equations differ only in their Lagrange



multiplier terms, these solutions are obtained by substituting $\mathscr{L}(t)\kappa(\phi(\mathbf{x},t))\sqrt{2F(\phi(\mathbf{x},t))}$ in Eq. (3.3) with $\mathscr{L}(t)$ and $\mathscr{L}(t)\sqrt{2F(\phi(\mathbf{x},t))}$, respectively.

Now, we consider the discretization in a two-dimensional space. We describe a hybrid numerical method for solving the CAC equation on $\Omega = (L_x, R_x) \times (L_y, R_y)$. Define $x_i = L_x + (i-0.5)h$ for $i = 1, \ldots, N_x$ and $y_j = L_y + (j-0.5)h$ for $j = 1, \ldots, N_y$ where N_x and N_y are positive integers, and $h = (R_x - L_x)/N_x = (R_y - L_y)/N_y$. Here, ϕ_{ij}^n represents numerical approximations of $\phi(x_i, y_j, n\Delta t)$, with Δt that denotes the temporal step size. Figure 3.1 aids in understanding the discretized computational domain Ω. Depending on the numerical method used, x_0, x_{N_x+1}, y_0, and y_{N_y+1} are employed as ghost points, as shown in Fig. 3.1.

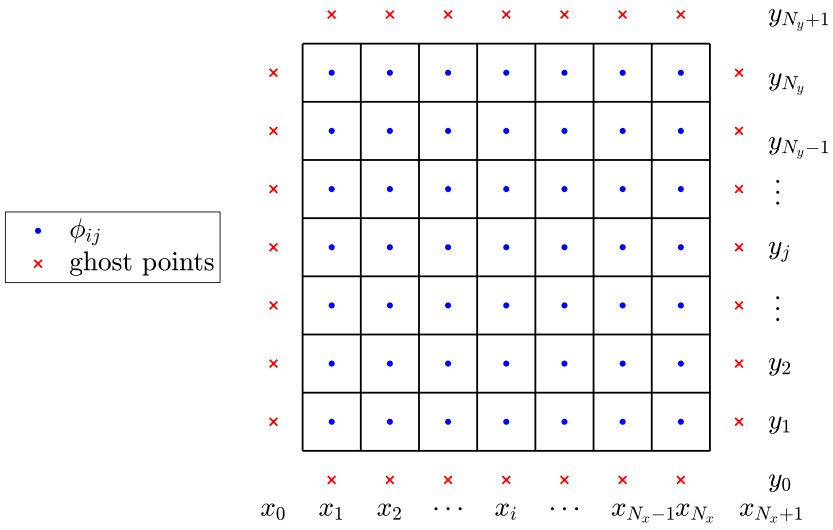


Figure 3.1: Schematic representation of computational domain Ω.

3.1 Fourier-spectral method

To solve Eq. (3.1), we use the Fourier-spectral method [56]: For the given data $\{\phi_{ij}^n | i = 1, \cdots, N_x \text{ and } j = 1, \cdots, N_y\}$, the discrete cosine transform is defined as follows

$$\begin{aligned} \hat{\phi}_{pq}^n &= \alpha_p \beta_q \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} \phi_{ij}^n \cos \frac{(2i-1)(p-1)\pi}{2N_x} \cos \frac{(2j-1)(q-1)\pi}{2N_y}, \\ & p = 1, \cdots, N_x \text{ and } q = 1, \cdots, N_y, \end{aligned} \tag{3.4}$$

where

$$\alpha_p = \begin{cases} \sqrt{\dfrac{1}{N_x}}, & p = 1 \\ \sqrt{\dfrac{2}{N_x}}, & 2 \le p \le N_x \end{cases} \quad \text{and} \quad \beta_q = \begin{cases} \sqrt{\dfrac{1}{N_y}}, & q = 1 \\ \sqrt{\dfrac{2}{N_y}}, & 2 \le q \le N_y \end{cases}.$$

For simplicity of exposition, we assume $L_x = L_y = 0$. For the cases of non-zero L_x and L_y, please refer to [53]. Let $x_i = (2i-1)R_x/(2N_x)$, $y_j = (2j-1)R_y/(2N_y)$, $\xi_p = (p-1)/R_x$, and $\eta_q = (q-1)/R_y$. Then, we can rewrite Eq. (3.4) as

$$\hat{\phi}_{pq}^n = \alpha_p \beta_q \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} \phi_{ij}^n \cos(\xi_p \pi x_i) \cos(\eta_q \pi y_j).$$

The inverse discrete cosine transform is

$$\phi_{ij}^n = \sum_{p=1}^{N_x} \sum_{q=1}^{N_y} \alpha_p \beta_q \hat{\phi}_{pq}^n \cos(\xi_p \pi x_i) \cos(\eta_q \pi y_j). \tag{3.5}$$

Let us assume that

$$\phi(x,y,n\Delta t) = \sum_{p=1}^{N_x}\sum_{q=1}^{N_y} \alpha_p\beta_q\hat{\phi}_{pq}^n \cos(\xi_p\pi x)\cos(\eta_q\pi y).$$

Then, we have

$$\begin{aligned}
\frac{\partial^2\phi}{\partial x^2}(x,y,n\Delta t) &= -\sum_{p=1}^{N_x}\sum_{q=1}^{N_y} (\xi_p\pi)^2 \alpha_p\beta_q\hat{\phi}_{pq}^n \cos(\xi_p\pi x)\cos(\eta_q\pi y), \\
\frac{\partial^2\phi}{\partial y^2}(x,y,n\Delta t) &= -\sum_{p=1}^{N_x}\sum_{q=1}^{N_y} (\eta_q\pi)^2 \alpha_p\beta_q\hat{\phi}_{pq}^n \cos(\xi_p\pi x)\cos(\eta_q\pi y).
\end{aligned}$$

Therefore, the Laplacian operator is defined as

$$\begin{aligned}
\Delta\phi(x,y,n\Delta t) &= \frac{\partial^2\phi}{\partial x^2}(x,y,n\Delta t) + \frac{\partial^2\phi}{\partial y^2}(x,y,n\Delta t) \\
&= -\sum_{p=1}^{N_x}\sum_{q=1}^{N_y} \left[(\xi_p\pi)^2 + (\eta_q\pi)^2\right] \alpha_p\beta_q\hat{\phi}_{pq}^n \cos(\xi_p\pi x)\cos(\eta_q\pi y). \quad (3.6)
\end{aligned}$$

Using Eqs. (3.5) and (3.6), from Eq. (3.1) we have

$$\frac{d\hat{\phi}_{pq}}{dt} = -\left[(\xi_p\pi)^2 + (\eta_q\pi)^2\right]\hat{\phi}_{pq}.$$

Then, we have the following solution after time step Δt with the initial condition $\hat{\phi}_{pq}^n$:

$$\hat{\phi}_{pq}^* = \hat{\phi}_{pq}^n e^{-\Delta t[(\xi_p\pi)^2 + (\eta_q\pi)^2]}. \quad (3.7)$$

Then, the numerical solution ϕ_{ij}^* is obtained using Eqs. (3.5) and (3.7), i.e.,

$$\phi_{ij}^* = \sum_{p=1}^{N_x}\sum_{q=1}^{N_y} \alpha_p\beta_q\hat{\phi}_{pq}^* \cos(\xi_p\pi x_i)\cos(\eta_q\pi y_j).$$

Second, Eq. (3.2) is solved analytically by the method of separation of variables:

$$\begin{aligned}
&\frac{d\phi}{dt} = -\frac{F'(\phi)}{\varepsilon^2} \\
&\Rightarrow \frac{d\phi}{-F'(\phi)} = \frac{1}{\varepsilon^2}dt \\
&\Rightarrow \frac{1}{\phi-\phi^3}d\phi = \frac{1}{\varepsilon^2}dt \\
&\Rightarrow \frac{1}{\phi(1-\phi)(1+\phi)}d\phi = \frac{1}{\varepsilon^2}dt \\
&\Rightarrow \left(\frac{1}{\phi} + \frac{1}{2(1-\phi)} - \frac{1}{2(1+\phi)}\right)d\phi = \frac{1}{\varepsilon^2}dt \\
&\Rightarrow \int_{\phi_{ij}^*}^{\phi_{ij}^{**}} \left(\frac{1}{\phi} + \frac{1}{2(1-\phi)} - \frac{1}{2(1+\phi)}\right)d\phi = \int_{n\Delta t}^{(n+1)\Delta t} \frac{1}{\varepsilon^2}dt. \qquad (3.8)
\end{aligned}$$

The right hand side of Eq. (3.8) is calculated as follows:

$$\int_{n\Delta t}^{(n+1)\Delta t} \frac{1}{\varepsilon^2}dt = \frac{1}{\varepsilon^2}\Delta t.$$

And the left hand side of Eq. (3.8) is calculated as follows:

$$\begin{aligned}
\int_{\phi_{ij}^*}^{\phi_{ij}^{**}} \left(\frac{1}{\phi} + \frac{1}{2(1-\phi)} - \frac{1}{2(1+\phi)}\right)d\phi &= \left[\ln\left|\frac{\phi}{\sqrt{(1-\phi)(1+\phi)}}\right|\right]_{\phi_{ij}^*}^{\phi_{ij}^{**}} \\
&= \ln\left|\frac{\phi_{ij}^{**}}{\sqrt{1-(\phi_{ij}^{**})^2}}\right| - \ln\left|\frac{\phi_{ij}^{*}}{\sqrt{1-(\phi_{ij}^{*})^2}}\right|.
\end{aligned}$$

Therefore,

$$\ln\left|\frac{\phi_{ij}^{**}}{\sqrt{1-(\phi_{ij}^{**})^2}}\right| - \ln\left|\frac{\phi_{ij}^{*}}{\sqrt{1-(\phi_{ij}^{*})^2}}\right| = \frac{1}{\varepsilon^2}\Delta t. \qquad (3.9)$$

Equation (3.9) is rewritten as follows

$$\left| \frac{\phi_{ij}^{**}}{\sqrt{1-(\phi_{ij}^{**})^2}} \right| = \left| \frac{\phi_{ij}^{*}}{\sqrt{1-(\phi_{ij}^{*})^2}} \right| e^{\frac{\Delta t}{\varepsilon^2}}. \tag{3.10}$$

Rewriting Eq. (3.10) in term of ϕ_{ij}^{**} gives

$$\phi_{ij}^{**} = \frac{\phi_{ij}^{*}}{\sqrt{\left(\phi_{ij}^{*}\right)^2 \left(1-e^{-\frac{2\Delta t}{\varepsilon^2}}\right) + e^{-\frac{2\Delta t}{\varepsilon^2}}}}. \tag{3.11}$$

Third, we discretize Eq. (3.3) as

$$\frac{\phi_{ij}^{n+1} - \phi_{ij}^{**}}{\Delta t} = \mathscr{L}^{**} \kappa(\phi_{ij}^{**}) \sqrt{2F(\phi_{ij}^{**})}. \tag{3.12}$$

By Eq. (3.12), we get

$$\phi_{ij}^{n+1} = \phi_{ij}^{**} + \Delta t \mathscr{L}^{**} \kappa_{ij}^{**} \sqrt{2F(\phi_{ij}^{**})}, \tag{3.13}$$

then by the property of Eq. (2.4),

$$\begin{aligned}
&\frac{1}{\Delta t}\left(\sum_{i=1}^{N_x}\sum_{j=1}^{N_y}\phi_{ij}^{n+1} - \sum_{i=1}^{N_x}\sum_{j=1}^{N_y}\phi_{ij}^{n}\right) = 0, \\
&\sum_{i=1}^{N_x}\sum_{j=1}^{N_y}\phi_{ij}^{0} = \sum_{i=1}^{N_x}\sum_{j=1}^{N_y}\phi_{ij}^{n+1} = \sum_{i=1}^{N_x}\sum_{j=1}^{N_y}\left(\phi_{ij}^{**} + \Delta t \mathscr{L}^{**} \kappa_{ij}^{**} \sqrt{2F(\phi_{ij}^{**})}\right).
\end{aligned}$$

Thus,

$$\mathscr{L}^{**} = \frac{\sum_{i=1}^{N_x}\sum_{j=1}^{N_y}(\phi_{ij}^0 - \phi_{ij}^{**})}{\Delta t \sum_{i=1}^{N_x}\sum_{j=1}^{N_y}\kappa_{ij}^{**}\sqrt{2F(\phi_{ij}^{**})}}. \tag{3.14}$$

Here, κ_{ij}^{**} represents the discrete curvature.

3.2 Discrete curvature

When calculating the discrete curvature, we take into account both anisotropic and isotropic curvatures.

3.2.1 Anisotropic curvature

Before the discrete curvature is defined, let us define the gradients at corners of the cell Ω_{ij} as

$$\begin{aligned}
\nabla\phi_{i-\frac{1}{2},j-\frac{1}{2}} &= (\phi_{x,i-\frac{1}{2},j-\frac{1}{2}}, \phi_{y,i-\frac{1}{2},j-\frac{1}{2}}) \\
&= \left(\frac{\phi_{i,j-1}+\phi_{ij}-\phi_{i-1,j-1}-\phi_{i-1,j}}{2h}, \frac{\phi_{i-1,j}+\phi_{ij}-\phi_{i-1,j-1}-\phi_{i,j-1}}{2h}\right), \\
\nabla\phi_{i+\frac{1}{2},j-\frac{1}{2}} &= (\phi_{x,i+\frac{1}{2},j-\frac{1}{2}}, \phi_{y,i+\frac{1}{2},j-\frac{1}{2}}) \\
&= \left(\frac{\phi_{i+1,j-1}+\phi_{i+1,j}-\phi_{i,j-1}-\phi_{ij}}{2h}, \frac{\phi_{ij}+\phi_{i+1,j}-\phi_{i,j-1}-\phi_{i+1,j-1}}{2h}\right), \\
\nabla\phi_{i+\frac{1}{2},j+\frac{1}{2}} &= (\phi_{x,i+\frac{1}{2},j+\frac{1}{2}}, \phi_{y,i+\frac{1}{2},j+\frac{1}{2}}) \\
&= \left(\frac{\phi_{i+1,j}+\phi_{i+1,j+1}-\phi_{ij}-\phi_{i,j+1}}{2h}, \frac{\phi_{i,j+1}+\phi_{i+1,j+1}-\phi_{ij}-\phi_{i+1,j}}{2h}\right), \\
\nabla\phi_{i-\frac{1}{2},j+\frac{1}{2}} &= (\phi_{x,i-\frac{1}{2},j+\frac{1}{2}}, \phi_{y,i-\frac{1}{2},j+\frac{1}{2}}) \\
&= \left(\frac{\phi_{ij}+\phi_{i,j+1}-\phi_{i-1,j}-\phi_{i-1,j+1}}{2h}, \frac{\phi_{i-1,j+1}+\phi_{i,j+1}-\phi_{i-1,j}-\phi_{ij}}{2h}\right).
\end{aligned}$$

Then, the curvature at the cell center is

$$\begin{aligned}\nabla\cdot\left(\frac{\nabla\phi}{|\nabla\phi|}\right)_{ij} &= \frac{1}{2h}\left(\frac{\phi_{x,i+\frac{1}{2},j+\frac{1}{2}}+\phi_{y,i+\frac{1}{2},j+\frac{1}{2}}}{|\nabla\phi_{i+\frac{1}{2},j+\frac{1}{2}}|}+\frac{\phi_{x,i+\frac{1}{2},j-\frac{1}{2}}-\phi_{y,i+\frac{1}{2},j-\frac{1}{2}}}{|\nabla\phi_{i+\frac{1}{2},j-\frac{1}{2}}|}\right.\\ &\left.-\frac{\phi_{x,i-\frac{1}{2},j+\frac{1}{2}}-\phi_{y,i-\frac{1}{2},j+\frac{1}{2}}}{|\nabla\phi_{i-\frac{1}{2},j+\frac{1}{2}}|}-\frac{\phi_{x,i-\frac{1}{2},j-\frac{1}{2}}+\phi_{yi-\frac{1}{2},j-\frac{1}{2}}}{|\nabla\phi_{i-\frac{1}{2},j-\frac{1}{2}}|}\right).\end{aligned}$$

To prevent numerical singularity when $|\nabla\phi|$ approaches zero, we set $\kappa_{ij}=0$ for regions where $|\phi_{ij}|>0.98$, effectively outside the interface.

3.2.2 Isotropic curvature

Isotropic calculations are widely used in various fields to eliminate differences in a specific direction. To ensure the accuracy of curvature calculations, we consider isotropic discretization.

$$\begin{aligned}\nabla_x\phi_{ij} &= \frac{\phi_{i+1,j+1}-\phi_{i-1,j+1}+4(\phi_{i+1,j}-\phi_{i-1,j})+\phi_{i+1,j-1}-\phi_{i-1,j-1}}{12h},\\ \nabla_y\phi_{ij} &= \frac{\phi_{i+1,j+1}-\phi_{i+1,j-1}+4(\phi_{i,j+1}-\phi_{i,j-1})+\phi_{i-1,j+1}-\phi_{i-1,j-1}}{12h},\\ m_{ij} &= \frac{\nabla\phi_{ij}}{|\nabla\phi_{ij}|}\\ &= \frac{\nabla_x\phi_{ij}+\nabla_y\phi_{ij}}{\sqrt{(\nabla_x\phi_{ij})^2+(\nabla_y\phi_{ij})^2}}.\end{aligned}$$

Therefore, discrete curvature is defined as follows:

$$\begin{aligned}\nabla\cdot\left(\frac{\nabla\phi}{|\nabla\phi|}\right)_{ij} &= \frac{1}{12h}\left[m^n_{x,i+1,j+1}-m^n_{x,i-1,j+1}+4(m^n_{x,i+1,j}-m^n_{x,i-1,j})\right.\\ &+m_{x,i+1,j-1}-m_{x,i-1,j-1}+m_{y,i+1,j+1}-m_{y,i+1,j-1}\\ &\left.+4(m_{y,i,j+1}-m_{y,i,j-1})+m_{y,i-1,j+1}-m_{y,i-1,j-1}\right].\end{aligned}$$

3.3 Alternating direction explicit method

We employ various numerical methods in addition to the spectral method to solve Eq. (3.1). Since we use the OSM, Eqs. (3.2) and (3.3) are solved equivalently as Eqs. (3.11) and (3.13). In this section, we utilize the alternating direction explicit (ADE) method, also known as the Saul'yev scheme [57, 58, 59], to solve Eq. (3.1). In two dimensional domain, there are four types of loops. We discretize Eq. (3.1) for one of four types loop cases. The fully discrete form of Eq. (3.1) is expressed as follows:

$$\begin{aligned}\frac{\phi_{ij}^{*}-\phi_{ij}^{n}}{\Delta t} &= \frac{\phi_{i+1,j}^{n}-\phi_{ij}^{n}}{h^2}+\frac{\phi_{ij}^{*}-\phi_{i-1,j}^{*}}{h^2}+\frac{\phi_{i,j+1}^{n}-\phi_{ij}^{n}}{h^2}+\frac{\phi_{ij}^{*}-\phi_{i,j-1}^{*}}{h^2}\\ &= \frac{\phi_{i+1,j}^{n}+\phi_{i-1,j}^{*}-2\phi_{ij}^{n}-2\phi_{ij}^{*}+\phi_{i,j+1}^{n}+\phi_{i,j-1}^{*}}{h^2},\end{aligned}$$

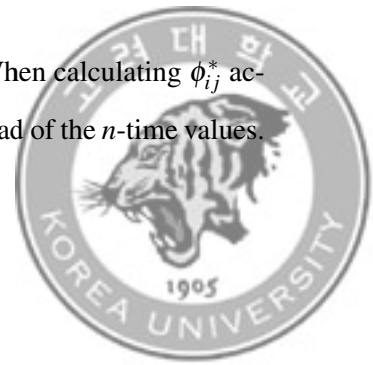
for $j=1,\ 2,\ \ldots,\ N_y$, for $i=1,\ 2,\ \ldots,\ N_x$. Then we obtain

$$\phi^{*}=\frac{\Delta t\left(\phi_{i+1,j}^{n}+\phi_{i-1,j}^{*}+\phi_{i,j+1}^{n}+\phi_{i,j-1}^{*}\right)+\left(h^2-2\Delta t\right)\phi_{ij}^{n}}{h^2+2\Delta t}.$$

Analogous discretizations of Eq. (3.1) can be derived for the remaining three loop configurations. The other three loop cases are as follows:

- For $j=1,\ 2,\ldots,\ N_y$, for $i=N_x,\ N_x-1,\ldots,\ 1$.
- For $j=N_y,\ N_y-1,\ldots,\ 1$, for $i=1,\ 2,\ldots,\ N_x$.
- For $j=N_y,\ N_y-1,\ldots,\ 1$, for $i=N_x,\ N_x-1,\ldots,\ 1$.

Figure 3.2 depicts schematics representing the four loop cases. When calculating ϕ_{ij}^{*} according to the loop, the values within the green circles use ϕ^{*} instead of the n-time values.



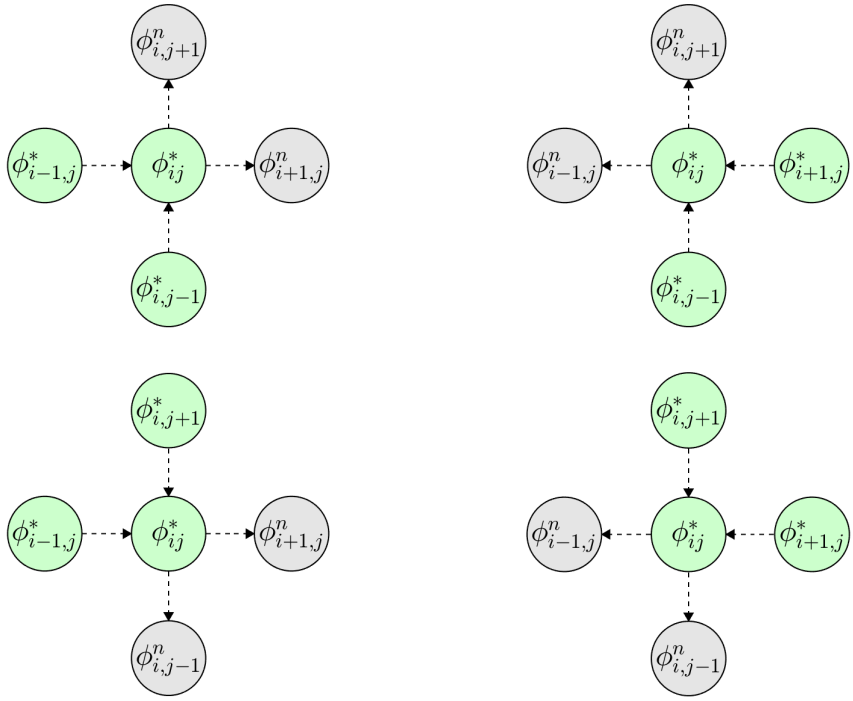


Figure 3.2: Schematic diagram of the four loop configurations in two dimensional space.

3.4 Runge–Kutta–Fehlberg method

In this section we employ the Runge–Kutta–Fehlberg (RKF) method, which is widely used as a high-order method, to solve Eq. (3.1). First, we define $\Delta_h\phi$, the discretization of $\Delta\phi$.

$$\Delta_h\phi_{ij} = \frac{\phi_{i+1,j} + \phi_{i-1,j} + \phi_{i,j+1} + \phi_{i,j-1} - 4\phi_{ij}}{h^2}.$$

Next, applying the RKF method to Eq. (3.1) yields the following solution.

$$\phi^*_{ij} = \phi^n_{ij} + \frac{25}{216}k_1 + \frac{1408}{2565}k_3 + \frac{2197}{4104}k_4 - \frac{1}{5}k_5,$$

where the coefficients are

$$
\begin{aligned}
k_1 &= \Delta t \Delta_h \left(\phi_{ij}^n\right), \\
k_2 &= \Delta t \Delta_h \left(\phi_{ij}^n + \frac{1}{4}k_1\right), \\
k_3 &= \Delta t \Delta_h \left(\phi_{ij}^n + \frac{3}{32}k_1 + \frac{9}{32}k_2\right), \\
k_4 &= \Delta t \Delta_h \left(\phi_{ij}^n + \frac{1932}{2197}k_1 - \frac{7200}{2197}k_2 + \frac{7296}{2197}k_3\right), \\
k_5 &= \Delta t \Delta_h \left(\phi_{ij}^n + \frac{439}{216}k_1 - 8k_2 + \frac{3680}{513}k_3 - \frac{845}{4104}k_4\right), \\
k_6 &= \Delta t \Delta_h \left(\phi_{ij}^n - \frac{8}{27}k_1 + 2k_2 - \frac{3544}{2565}k_3 + \frac{1859}{4104}k_4 - \frac{11}{40}k_5\right).
\end{aligned}
$$

3.5 Algorithm Summary

Let us consider into the fundamental mechanism underlying the proposed model (2.3) to understand its core principles and functioning. This examination provide insight into how the model operates and its potential applications.

Figure 3.3 presents a detailed schematic illustration of the algorithms for three distinct CAC equations. These equations, specifically (2.1), (2.2), and (2.3), are depicted step-by-step to provide a comprehensive understanding. Each step in the illustration highlights the unique computational processes and methodologies employed for solving these CAC equations, ensuring clarity and precision in the algorithmic representation. Figure 3.3(a) illustrates the initial condition ϕ^0 comprising two disks of different sizes. Figure 3.3(b) depicts the numerical solution derived from the AC equation, specifically following steps (3.1) and (3.2). Due to the characteristic of motion driven by

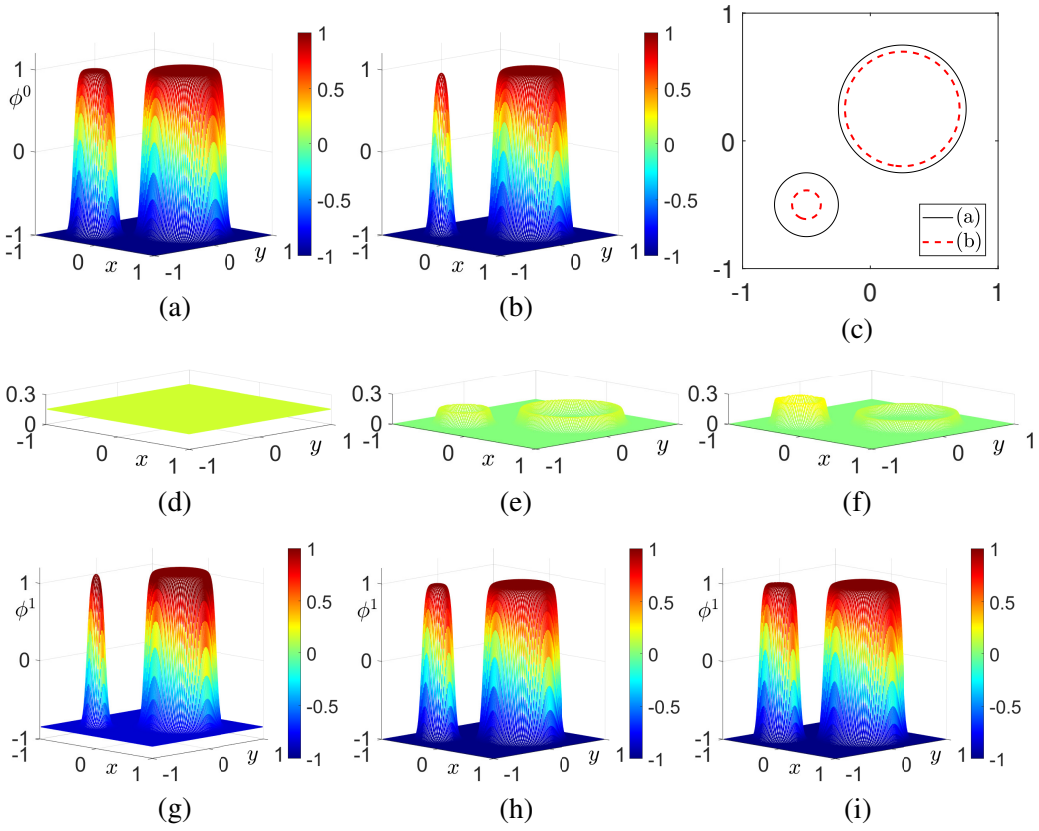


Figure 3.3: A visual representation of conservative algorithms employing the OSM: (a) Initial condition ϕ^0; (b) Result after solving the AC equation; (c) Zero-level contours of (a) and (b); (d)–(f) represent three different Lagrange multiplier as conservative corrections: $\mathscr{L}(t)$, $\mathscr{L}(t)\sqrt{2F(\phi)}$, and $\mathscr{L}(t)\kappa(\phi)\sqrt{2F(\phi)}$, respectively; (g)–(i) show solutions ϕ^1 of the CAC equation with $\mathscr{L}(t)$, $\mathscr{L}(t)\sqrt{2F(\phi)}$, and $\mathscr{L}(t)\kappa(\phi)\sqrt{2F(\phi)}$, respectively.

mean curvature flow, the smaller disk shrinks more significantly compared to the larger disk. This phenomenon is evident in Fig. 3.3(c), which shows the zero-level contours of Figs. 3.3(a) and (b). Figures 3.3(d), (e), and (f) depict the conservative corrections $\mathscr{L}(t)$, $\mathscr{L}(t)\sqrt{2F(\phi)}$, and $\mathscr{L}(t)\kappa(\phi)\sqrt{2F(\phi)}$, respectively. Unlike conventional mass correction methods, which involve shifting a constant or smoothed Dirac delta function-like profile across the interface transition layer, the proposed mass correction scheme employs curvature-dependent profiles, as illustrated in Fig. 3.3(f). Figures 3.3(g), 3.3(h), and 3.3(i) show the solutions ϕ^1 of the CAC equation using $\mathscr{L}(t)$, $\mathscr{L}(t)\sqrt{2F(\phi)}$, and $\mathscr{L}(t)\kappa(\phi)\sqrt{2F(\phi)}$, respectively. The core mechanism of the proposed model is to correct mass loss due to motion by mean curvature using a curvature-dependent Lagrange multiplier.

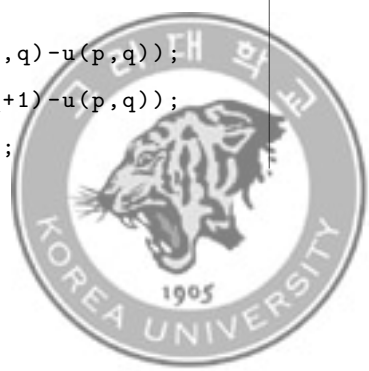
3.6 MATLAB code

This section provides MATLAB code to solve the curvature-dependent CAC equation using the proposed numerical method. First, here is MATLAB code implemented with the spectral method.

Listing 3.1: The main code for solving curvature-dependent CAC equations using the spectral method.

```
clear;
Nx = 128; Ny= Nx;
Lx = -2; Rx = 2; Ly = -2; Ry = 2;
h = (Rx-Lx)/Nx;
x = linspace(Lx+0.5*h,Rx-0.5*h,Nx);
y = linspace(Ly+0.5*h,Ry-0.5*h,Ny);
[xx,yy] = ndgrid(x,y);
```

```
ep = 5*h/(2*sqrt(2)*atanh(0.9)); ep2 = ep^2;
dt = 0.1*h^2; maxit = 500;
v = tanh((1-sqrt(xx.^2+yy.^2))/(sqrt(2)*eps));
ss = sum(sum(v));
kx = pi*(0:Nx-1)/(Rx-Lx); ky = pi*(0:Ny-1)/(Ry-Ly);
kx2 = kx.^2; ky2 = ky.^2;
[kkx2, kky2] = ndgrid(ky2,kx2);

figure(1); clf; hold on; grid on; view(-30,20);
mesh(x,y,real(v));
colormap turbo; clim([-1 1]); colorbar;
axis image;

figure(2); clf; hold on; box on
contour(x,y,real(v),[0 0],'k-');
axis image;

for iter=1:maxit
    v_hat = dct2(v).*exp(-dt*(kkx2+kky2));
    v = real(idct2(v_hat));
    v = v./sqrt((1.0-v.^2)*exp(-2.0*dt/ep2)+v.^2);
    w = 0*v;
    for i = 2:Nx-1
        for j = 2:Ny-1
            u = v(i-1:i+1,j-1:j+1);
            for p = 1:2
                for q = 1:2
                    tx = (u(p+1,q+1)-u(p,q+1)+u(p+1,q)-u(p,q));
                    ty = (u(p+1,q+1)-u(p+1,q)+u(p,q+1)-u(p,q));
                    nux(p,q) = tx/(sqrt(tx^2+ty^2));
```

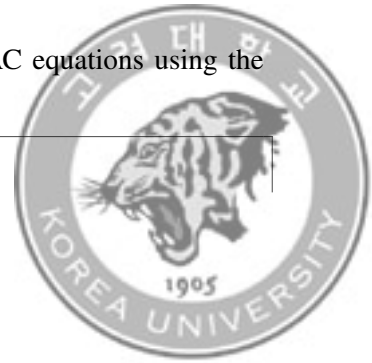



```
                    nuy(p,q) = ty/(sqrt(tx^2+ty^2));
                end
            end
            if abs(v(i,j))<0.98
                w(i,j) = ((nux(2,2)-nux(1,2)+nux(2,1)-nux(1,1))/h...
                    +(nuy(2,2)-nuy(2,1)+nuy(1,2)-nuy(1,1))/h)*0.5;
            end
        end
    end
    nume = sum(sum(v));
    deno = sum(sum(abs(1-v.^2)./sqrt(2).*w));
    b = (ss-nume)/deno;
    v = v+b*abs(1-v.^2)./sqrt(2).*w;

    if mod(iter,100) == 0
        figure(1); clf; hold on; grid on; view(-30,20);
        mesh(x,y,real(v));
        colormap turbo; clim([-1 1]); colorbar;
        axis image;

        figure(2);
        contour(x,y,real(v),[0 0],'r-.');
        axis image;
    end
end
```

The second code is MATLAB code that applies the ADE method.

Listing 3.2: The main code for solving curvature-dependent CAC equations using the ADE method.

```
clear;
```

```
Nx = 128; Ny= Nx;
Lx = -2; Rx = 2; Ly = -2; Ry = 2;
h = (Rx-Lx)/Nx;
x = linspace(Lx-0.5*h,Rx+0.5*h,Nx+2);
y = linspace(Ly-0.5*h,Ry+0.5*h,Ny+2);
[xx,yy]=ndgrid(x,y);
ep=5*h/(2*sqrt(2)*atanh(0.9)); ep2=ep^2;
dt = 0.1*h^2; maxit=500;
v = tanh((1-sqrt(xx.^2+yy.^2))/(sqrt(2)*eps));
ss=sum(sum(v(2:Nx+1,2:Ny+1)));

figure(1); clf; hold on; grid on; view(-30,20);
mesh(x,y,real(v));
colormap turbo; clim([-1 1]); colorbar;
axis image;

figure(2); clf; hold on; box on
contour(x,y,real(v),[0 0],'k-');
axis image;
%%%% for 4-cases loop %%%%
tmp=zeros(Nx+2,Ny+2);
tmp(2:Nx+1,2:Ny+1)=1;
[lpx,lpy]=find(tmp==1);
llp=length(lpx);
lpx(:,2)=flipud(lpx(:,1));
lpy(:,2)=flipud(lpy(:,1));
k=0;
for j=Ny+1:-1:2
    for i=2:Nx+1
        k=k+1;
```


```
        lpx(k,3)=i;
        lpy(k,3)=j;
    end
end
lpx(:,4)=flipud(lpx(:,3));
lpy(:,4)=flipud(lpy(:,3));
%%%%%%%%%%%%%%%%%%%%%%%%%
for iter=1:maxit
    odr=mod(iter,4)+1;
    for k=1:llp
        i=lpx(k,odr);
        j=lpy(k,odr);
        if i==lpx(1,odr)
            Lapx = (v(i-sign(i-0.5*(Nx+2)),j)-v(i,j))/h^2;
            r=0;
        elseif i==lpx(end,odr)
            Lapx = v(i-sign(i-0.5*(Nx+2)),j)/h^2;
            r = 1/h^2;
        else
            Lapx = (v(i+1,j)-v(i,j)+v(i-1,j))/h^2;
            r = 1/h^2;
        end
        if j==lpy(1,odr)
            Lapy=(v(i,j-sign(j-0.5*(Ny+2)))-v(i,j))/h^2;
            r=r;
        elseif j==lpy(end,odr)
            Lapy = v(i,j-sign(j-0.5*(Ny+2)))/h^2;
            r = r+1/h^2;
        else
            Lapy=(v(i,j+1)-v(i,j)+v(i,j-1))/h^2;
```


```
            r = r+1/h^2;
        end
        r = r+1/dt;
        v(i,j)=(v(i,j)/dt+(Lapx+Lapy))/r;
    end
    v = v./sqrt((1.0-v.^2)*exp(-2.0*dt/ep2)+v.^2);

    w=0*v(2:Nx+1,2:Ny+1);
    for i=2:Nx+1
        for j=2:Ny+1
            u=v(i-1:i+1,j-1:j+1);
            for p=1:2
                for q=1:2
                    tx = (u(p+1,q+1)-u(p,q+1)+u(p+1,q)-u(p,q));
                    ty = (u(p+1,q+1)-u(p+1,q)+u(p,q+1)-u(p,q));
                    nux(p,q) = tx/(sqrt(tx^2+ty^2));
                    nuy(p,q) = ty/(sqrt(tx^2+ty^2));
                end
            end
            if abs(v(i,j))<0.98
                w(i-1,j-1) = ((nux(2,2)-nux(1,2)+nux(2,1)-nux(1,1))/
                    h ...
                    +(nuy(2,2)-nuy(2,1)+nuy(1,2)-nuy(1,1))/h)*0.5;
            end
        end
    end

    nume = sum(sum(v(2:Nx+1,2:Ny+1)));
    deno = sum(sum(abs(1-v(2:Nx+1,2:Ny+1).^2)./sqrt(2).*w));
    b = (ss-nume)/deno;
```


```
    v(2:Nx+1,2:Ny+1) = v(2:Nx+1,2:Ny+1)+b*abs(1-v(2:Nx+1,2:Ny+1).^2)
        ...
        ./sqrt(2).*w;

    v(1,:)=v(2,:); v(Nx+2,:)=v(Nx+1,:);
    v(:,1)=v(:,2); v(:,Ny+2)=v(:,Ny+1);

    if mod(iter,100) == 0
        figure(1); clf; hold on; grid on; view(-30,20);
        mesh(x(2:Nx+1),y(2:Ny+1),real(v(2:Nx+1,2:Ny+1)));
        colormap turbo; clim([-1 1]); colorbar;
        axis image;

        figure(2);
        contour(x(2:Nx+1),y(2:Ny+1),real(v(2:Nx+1,2:Ny+1)),[0 0],'r
            -.')
        axis image;
    end
end
```

Lastly, this code is MATLAB code using RKF.

Listing 3.3: The main code for solving curvature-dependent CAC equations using the RKF method.

```
clear;
Nx = 128; Ny= Nx;
Lx = -2; Rx = 2; Ly = -2; Ry = 2;
h = (Rx-Lx)/Nx;
x = linspace(Lx-0.5*h,Rx+0.5*h,Nx+2);
y = linspace(Ly-0.5*h,Ry+0.5*h,Ny+2);
[xx,yy]=ndgrid(x,y);
```


```
ep=5*h/(2*sqrt(2)*atanh(0.9)); ep2=ep^2;
dt = 0.1*h^2; maxit=500;
v = tanh((1-sqrt(xx.^2+yy.^2))/(sqrt(2)*eps));

ss=sum(sum(v(2:Nx+1,2:Ny+1)));
parameter=struct('Nx',Nx,'Ny',Ny,'h',h,'ep2',ep2,'v',v,'dt',dt);
coef21=1.0/4.0;
coef31=3.0/32.0;coef32=9.0/32.0;
coef41=1932.0/2197.0;coef42=-7200.0/2197.0;coef43=7296.0/2197.0;
coef51=439.0/216.0;coef52=-8.0;coef53=3680.0/513.0;
coef54=-845.0/4104.0;
coef61=-8.0/27.0;coef62=2.0;coef63=-3544.0/2565.0;coef64
    =1859.0/4104.0;
coef65=-11.0/40.0;
coefRR=[1.0/360.0,0.0,-128.0/4275.0,-2197.0/75240.0,0.02,2.0/55.0]/
    dt;
coefM=[25.0/216.0,0,1408/2565.0,2197.0/4104.0,-0.2];

figure(1); clf; hold on; grid on; view(-30,20);
mesh(x,y,real(v));
colormap turbo; clim([-1 1]); colorbar;
axis image;

figure(2); clf; hold on; box on
contour(x,y,real(v),[0 0],'k-');
axis image;

for iter=1:maxit
    K1=dt*AC(v,parameter);
    K2=dt*AC(v+coef21*K1,parameter);
```


```
K3=dt*AC(v+coef31*K1+coef32*K2,parameter);
K4=dt*AC(v+coef41*K1+coef42*K2+coef43*K3,parameter);
K5=dt*AC(v+coef51*K1+coef52*K2+coef53*K3+coef54*K4,parameter);
K6=dt*AC(v+coef61*K1+coef62*K2+coef63*K3+coef64*K4+coef65*K5,...
    parameter);
v = v+coefM(1)*K1+coefM(3)*K3+coefM(4)*K4+coefM(5)*K5;
v = v./sqrt((1.0-v.^2)*exp(-2.0*dt/ep2)+v.^2);

w=0*v(2:Nx+1,2:Ny+1);
for i=2:Nx+1
    for j=2:Ny+1
        u=v(i-1:i+1,j-1:j+1);
        for p=1:2
            for q=1:2
                tx = (u(p+1,q+1)-u(p,q+1)+u(p+1,q)-u(p,q));
                ty = (u(p+1,q+1)-u(p+1,q)+u(p,q+1)-u(p,q));
                nux(p,q) = tx/(sqrt(tx^2+ty^2));
                nuy(p,q) = ty/(sqrt(tx^2+ty^2));
            end
        end
        if abs(v(i,j))<0.98
            w(i-1,j-1) = 0.5*(nux(2,2)-nux(1,2)+nux(2,1)-nux
                (1,1))/h ...
                        +0.5*(nuy(2,2)-nuy(2,1)+nuy(1,2)-nuy
                            (1,1))/h;
        end
    end
end

nume = sum(sum(v(2:Nx+1,2:Ny+1)));
```


```
64      deno = sum(sum(abs(1-v(2:Nx+1,2:Ny+1).^2)./sqrt(2).*w));
65      b = (ss-nume)/deno;
66      v(2:Nx+1,2:Ny+1) = v(2:Nx+1,2:Ny+1)+b*abs(1-v(2:Nx+1,2:Ny+1).^2)
            ...
67                              ./sqrt(2).*w;
68
69      v(1,:)=v(2,:); v(Nx+2,:)=v(Nx+1,:);
70      v(:,1)=v(:,2); v(:,Ny+2)=v(:,Ny+1);
71
72      if mod(iter,100) == 0
73          figure(1); clf; hold on; grid on; view(-30,20);
74          mesh(x,y,real(v));
75          colormap turbo; clim([-1 1]); colorbar;
76          axis image;
77
78          figure(2);
79          contour(x,y,real(v),[0 0],'r-.');
80          axis image;
81      end
82  end
83
84  function v=AC(p,param)
85  Nx=param.Nx; Ny=param.Ny; ep2=param.ep2; h=param.h;
86  v=param.v; dt=param.dt;
87  p(1,:)=p(2,:);
88  p(Nx+2,:)=p(Nx+1,:);
89  p(:,1)=p(:,2);
90  p(:,Ny+2)=p(:,Ny+1);
91
92  for i=2:Nx+1
```


```
93      for j=2:Ny+1
94          v(i,j)= (p(i-1,j)+p(i+1,j)-4.0*p(i,j)+p(i,j-1)+p(i,j+1))/h
                ^2;
95      end
96  end
97
98  end
```

All three codes use the same initial condition, and later, various numerical experiments are performed by changing the initial condition, computational domain, and parameters.

Chapter 4. Numerical experiments

We now present several numerical computational experiments to validate the enhanced performance of the presented CAC equation with curvature-dependent Lagrange multiplier. For clarity, we define ε_m as $\varepsilon_m = hm/[2\sqrt{2}\tanh^{-1}(0.9)]$, where h denotes the grid size and m is a positive integer. This ensures that the transition layer across the interface is approximately hm; refer to [60] for a detailed and thorough explanation of ε_m. In our simulations, unless otherwise specified, we set $\varepsilon = \varepsilon_5$. These tests are designed to highlight the accuracy and robustness of the CAC equation in maintaining the integrity of the interface during time evolution, as well as its efficiency in computational performance. By examining various initial conditions and background fields, we can demonstrate how well the proposed model adapts to different scenarios and maintains stability.

4.1 Comparison of numerical schemes

Before performing various numerical experiments, we compare the results of different numerical schemes and select one algorithm. In the computational domain $\Omega = (-1,\ 1) \times (-1,\ 1)$, We compare the CPU time and numerical solution for a following

initial condition:

$$\phi(x,y,0) = \tanh(\frac{0.5 - \sqrt{x.^2 + y.^2}}{\sqrt{2}\varepsilon}).$$

The parameters were used as follows: $N_x = N_y = 100$, $h = 0.02$, $\Delta t = 0.1h^2$. The CPU time for each scheme was measured over 500 iterations. The CPU times for each scheme are provided in Table 4.1.

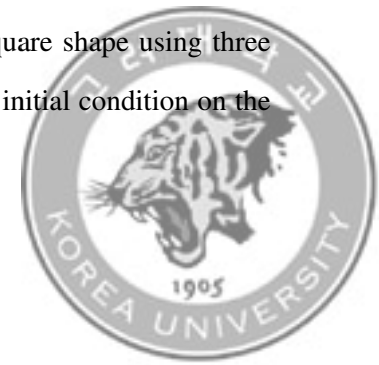
Table 4.1: The CPU time(s) for 500 iterations.

	Spectral method	ADE	RKF
CPU time(s)	2.6011	2.5083	2.6966

The results indicate that the ADE scheme is the fastest, while the RKF method is the slowest. Despite this, we have chosen to employ the spectral method for upcoming numerical tests due to its superior ability to preserve the characteristics of the stiff initial condition, as illustrated in Fig. 4.1. The decision to use the spectral method is driven by its higher accuracy in maintaining the intricate details of the initial state. This method ensures that the nuances of the initial condition are accurately captured and retained throughout the computations, which is crucial for the reliability of our tests. Furthermore, the spectral method's robustness in handling complex interface dynamics justifies its selection despite the slower computation speed.

4.2 Feature preserving property

Let us consider the snapshots of evolution of an initially square shape using three types of the CAC equations. The top row in Fig. 4.2 depicts the initial condition on the



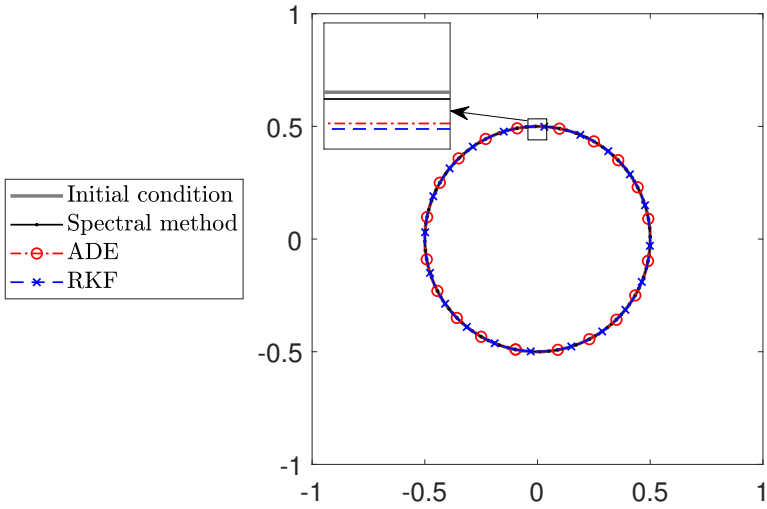


Figure 4.1: The zero-level contour of the initial condition and the solutions after 500 iterations.

computational domain $\Omega = (-2,2) \times (-2,2)$:

$$\phi(x,y,0) = \begin{cases} 1, & x,y \in (-1,1), \\ -1, & \text{otherwise.} \end{cases}$$

Here, we use $h = 0.625$ and $\Delta t = 0.1h^2$. Figures 4.2(a), (b), and (c) show the snapshots of the numerical evolution with different Lagrange multipliers: $\mathscr{L}(t)$, $\mathscr{L}(t)\sqrt{2F(\phi)}$, and $\mathscr{L}(t)\kappa(\phi)\sqrt{2F(\phi)}$, respectively. The snapshots are captured at $t = 0$, $100\Delta t$, and $500\Delta t$ from top to bottom. The fourth row in Fig. 4.2 presents the evolution of the zero-level contours of the numerical solutions using the three different Lagrange multipliers. Both conventional CAC equations result in circular shapes due to the motion by mean curvature with a mass constraint. In contrast, the proposed CAC equation successfully preserves the original square shape, maintaining a smoothed interface transition layer. This demonstrates the proposed method's effectiveness in retaining the integrity of complex

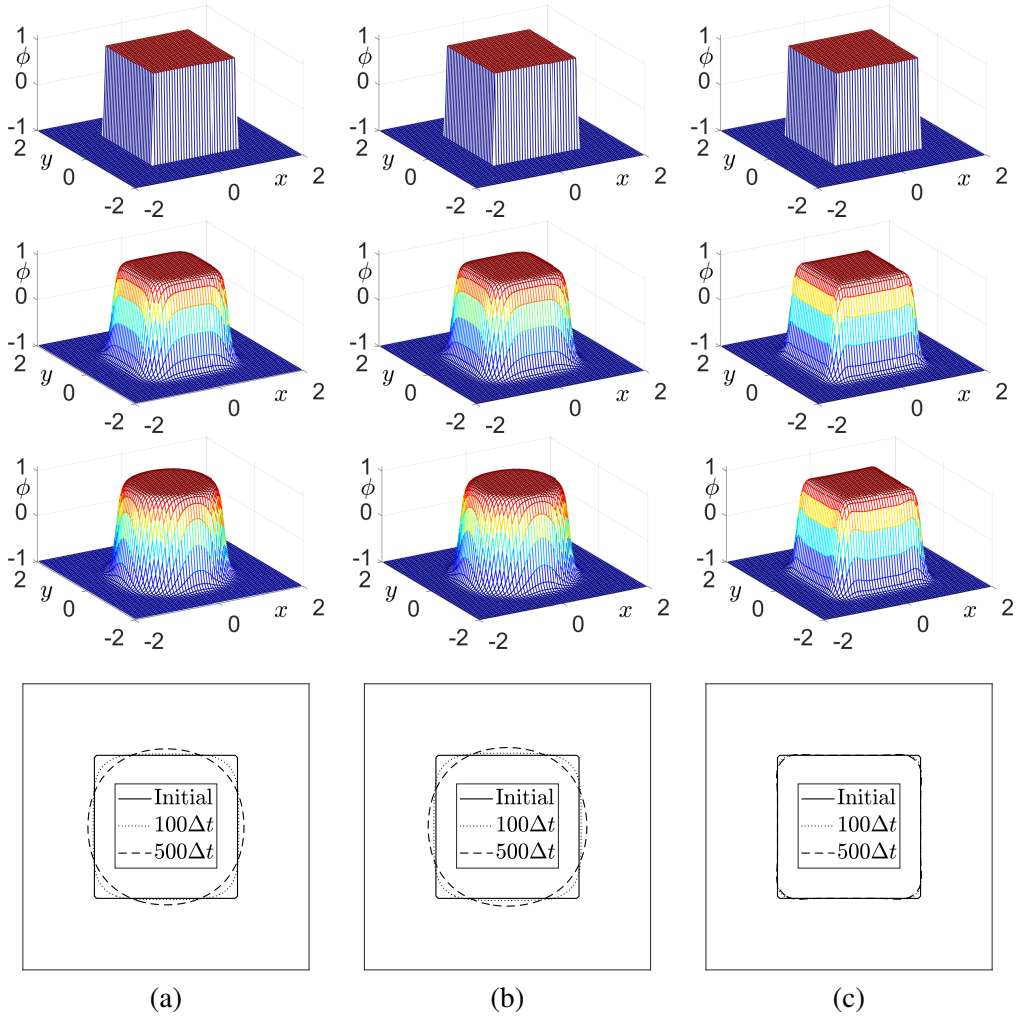


Figure 4.2: Snapshots of evolution of the numerical results with different Lagrange multipliers: (a) $\mathscr{L}(t)$, (b) $\mathscr{L}(t)\sqrt{2F(\phi)}$, and (c) $\mathscr{L}(t)\kappa(\phi)\sqrt{2F(\phi)}$. From top to bottom, times are at $t = 0$, $100\Delta t$, and $500\Delta t$.

geometric features over time, highlighting its potential for applications requiring precise interface preservation. Additionally, the stability and accuracy of the proposed method make it a robust choice for modeling intricate phase-field dynamics.

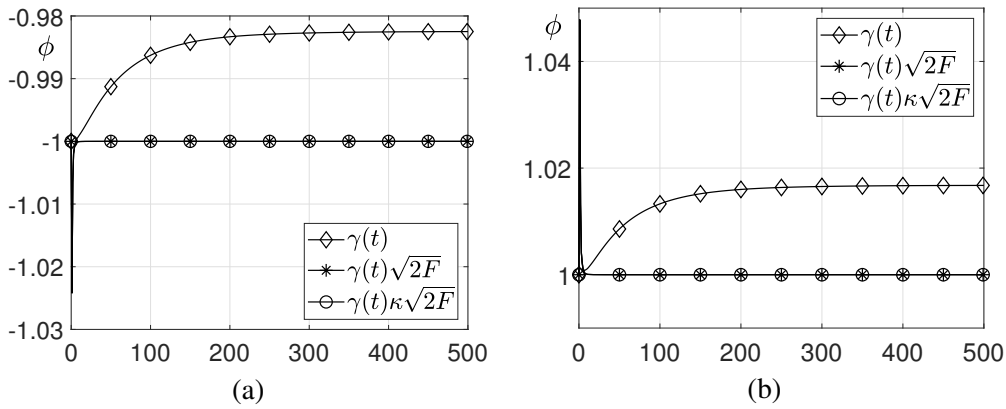


Figure 4.3: Snapshots of evolution of the (a) minimum (b) maximum values of the numerical solutions with three different Lagrange multipliers.

Figure 4.3(a) and (b) display the snapshots of the minimum and maximum values, respectively, during the evolution of the numerical solutions with three different Lagrange multipliers. The results indicate that the solution using the Lagrange multiplier $\mathscr{L}(t)$ deviates from the expected double well values of ± 1.

Next, let us consider the snapshots of evolution of an initially small square shape using three types of the CAC equations. The top row in Fig. 4.4 is the initial condition on $\Omega = (-2,2) \times (-2,2)$:

$$\phi(x,y,0) = \begin{cases} 1, & x,y \in (-0.2,0.2), \\ -1, & \text{otherwise.} \end{cases}$$

Here, we use $h = 0.625$ and $\Delta t = 0.1h^2$.

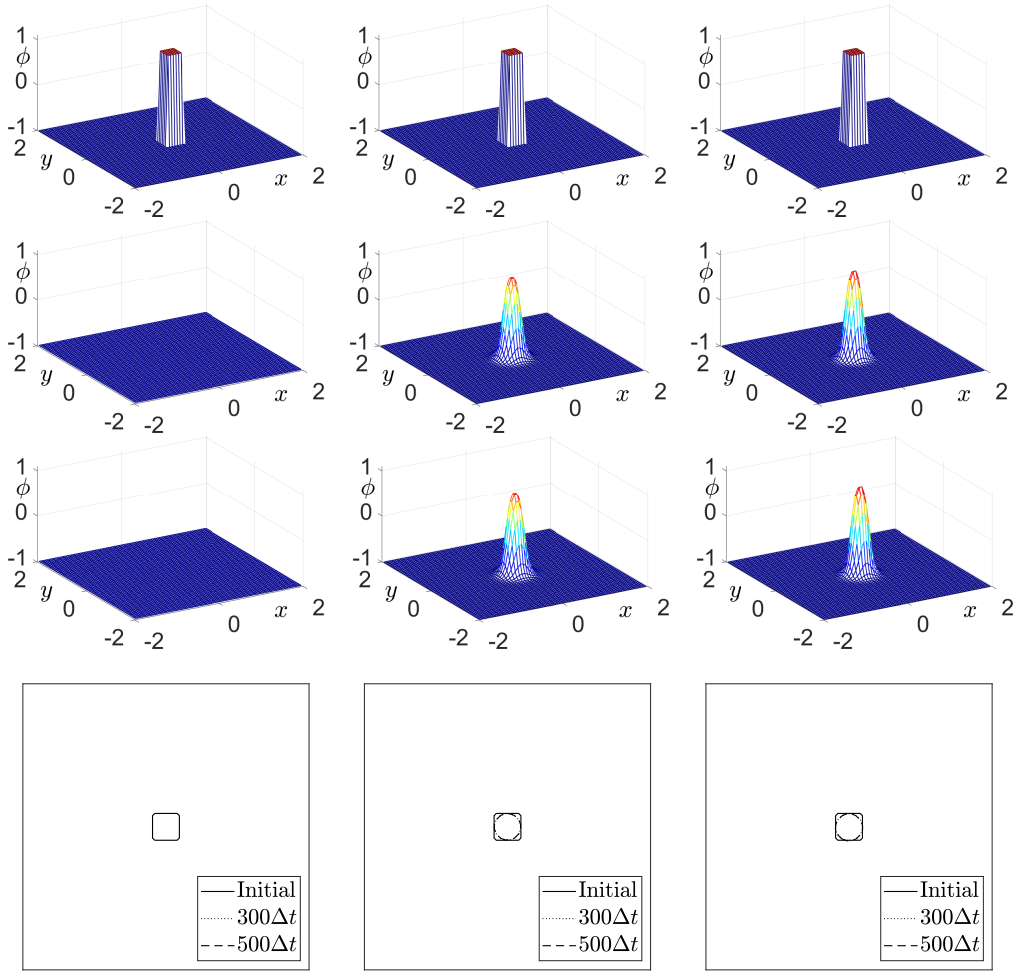


Figure 4.4: Snapshots of evolution of the numerical results with different Lagrange multipliers: (a) $\mathscr{L}(t)$, (b) $\mathscr{L}(t)\sqrt{2F(\phi)}$, and (c) $\mathscr{L}(t)\kappa(\phi)\sqrt{2F(\phi)}$. From top to bottom, times are at $t = 0$, $300\Delta t$, and $500\Delta t$.

Figures 4.4(a), (b), and (c) are the snapshots of evolution of the numerical results with different Lagrange multipliers: $\mathscr{L}(t)$, $\mathscr{L}(t)\sqrt{2F(\phi)}$, and $\mathscr{L}(t)\kappa(\phi)\sqrt{2F(\phi)}$, respectively. From top to bottom, times are at $t = 0$, $300\Delta t$, and $500\Delta t$. The fourth row in Fig. 4.4 shows the snapshots of evolution of the zero-level contours of the numerical solutions using three different Lagrange multipliers. The result from the conventional CAC equation with the Lagrange multiplier $\mathscr{L}(t)$ shows the initially small square shape eventually disappears and becomes flat profile. However, the results from the other two Lagrange multipliers have similar results showing circular shapes.

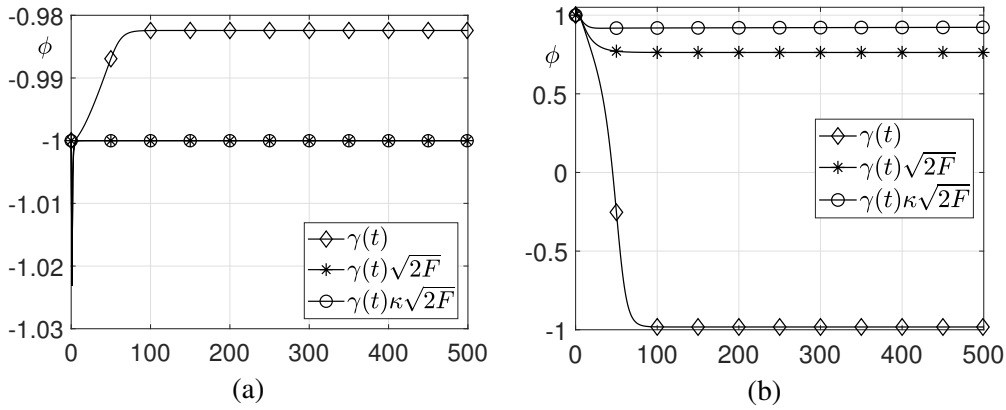


Figure 4.5: Snapshots of evolution of the (a) minimum (b) maximum values of the numerical solutions with three different Lagrange multipliers.

Figure 4.5(a) and (b) show the snapshots of evolution of the minimum and maximum values, respectively, of the numerical solutions with three different Lagrange multipliers. We can observe the result from the Lagrange multiplier $\mathscr{L}(t)$ becomes constant and the propose model has the best result among the others.

We now undertake a numerical test designed to illustrate the unique dynamics of the proposed CAC equation in contrast to the two conventional CAC equations. To achieve this, we consider the initial condition defined on the domain $\Omega = (-2,2) \times (-2,2)$ as

follows:

$$\begin{aligned}\phi(x,y,0) &= 1+\tanh\left(\frac{0.5-\sqrt{(x+1)^2+(y+1)^2}}{\sqrt{2}\varepsilon}\right)\\ &\quad+\tanh\left(\frac{1-\sqrt{(x-0.5)^2+(y-0.5)^2}}{\sqrt{2}\varepsilon}\right).\end{aligned}$$

The parameters used in the simulation are $h = 0.0625$ and $\Delta t = 0.1h^2$. As shown in Fig. 4.6, and schematically illustrated in Fig. 3.3, the behavior of the disks varies with different Lagrange multipliers. In the cases of $\mathscr{L}(t)$ and $\mathscr{L}(t)\sqrt{2F(\phi)}$ (top and middle rows), the smaller disk gradually shrinks while the larger disk expands, ultimately resulting in a single remaining disk. Conversely, with the proposed Lagrange multiplier $\mathscr{L}(t)\kappa(\phi)\sqrt{2F(\phi)}$ (bottom row), the initial shapes of the disks are preserved over time.

Furthermore, Fig. 4.6(d) presents the temporal evolution of the average concentration ϕ_{ave} and the relative deviation $\|\phi^0-\phi^n\|_2/\|\phi^0\|_2$, where $\|\cdot\|_2$ denotes the discrete l_2-norm. It is evident that the average concentration remains constant across all three cases. However, only the proposed method maintains a minimal relative deviation from the initial profiles, indicated by the very small value of $\|\phi^0-\phi^n\|_2/\|\phi^0\|_2$.

Let us consider two different sized squares on $\Omega = (-2,2)\times(-2,2)$ as shown in the first row in Fig. 4.7:

$$\phi(x,y,0) = \begin{cases} 1, & x,y \in (-1.5,-0.5) \text{ or } x,y \in (-0.2,1.3) \\ -1, & \text{otherwise} \end{cases}$$

The simulation uses parameters $h = 0.625$ and $\Delta t = 0.1h^2$. When using the Lagrange multipliers $\mathscr{L}(t)$ and $\mathscr{L}(t)\sqrt{2F(\phi)}$, the smaller square diminishes in size while the larger

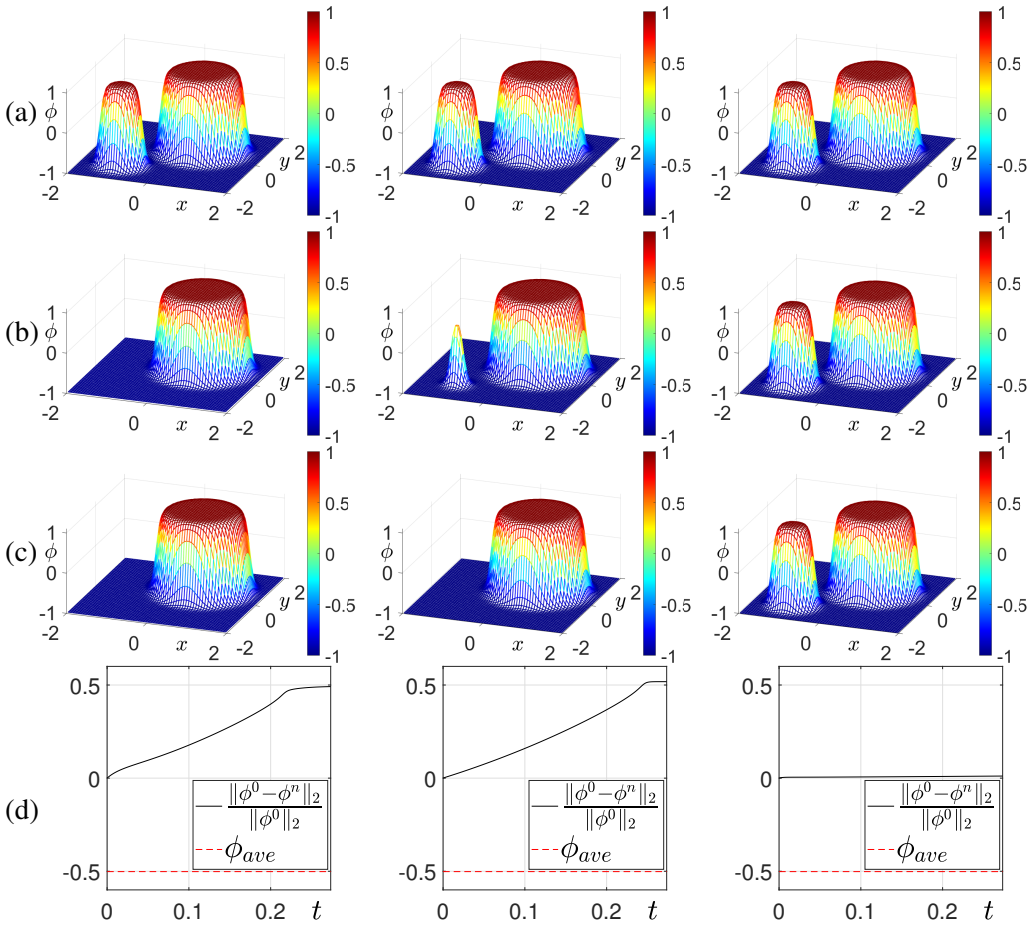


Figure 4.6: Temporal evolution of the numerical results with different Lagrange multipliers: $\mathscr{L}(t)$, $\mathscr{L}(t)\sqrt{2F(\phi)}$, and $\mathscr{L}(t)\kappa(\phi)\sqrt{2F(\phi)}$ from left to right colons, respectively. (a), (b), and (c) are the results at times $t = 0$, $600\Delta t$, and $700\Delta t$, respectively. (d) is the temporal evolution of the average concentration ϕ_{ave} and $\|\phi^0 - \phi^n\|_2/\|\phi^0\|_2$.

square enlarges, with both gradually becoming circular. Eventually, this process culminates in a single remaining disk.

Conversely, with the proposed Lagrange multiplier $\mathscr{L}(t)\kappa(\phi)\sqrt{2F(\phi)}$, the initial sharp transition softens, and the original shapes are preserved throughout the simulation. This demonstrates the proposed method's effectiveness in maintaining the original geometries, ensuring that the shapes remain intact as time progresses.

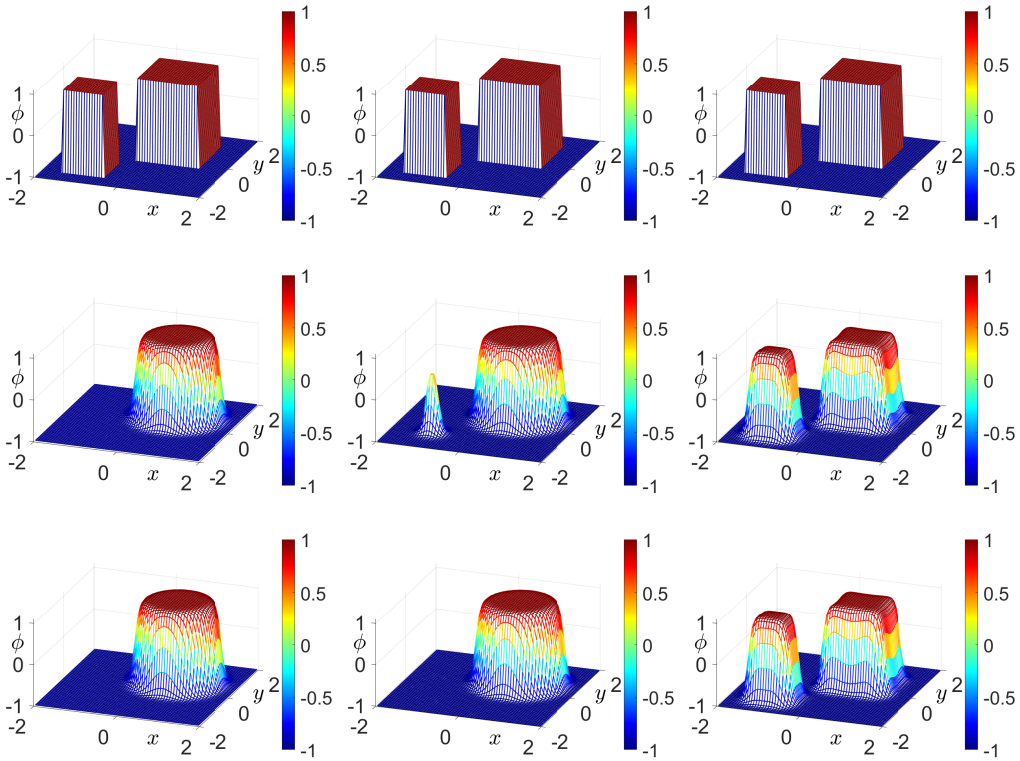


Figure 4.7: Snapshots depicting of evolution of the numerical results with three different Lagrange multipliers: (a) $\mathscr{L}(t)$, (b) $\mathscr{L}(t)\sqrt{2F(\phi)}$, and (c) $\mathscr{L}(t)\kappa(\phi)\sqrt{2F(\phi)}$. From top to bottom, the snapshots correspond to times $t = 0$, $1000\Delta t$, and $1100\Delta t$.

To further confirm the performance of the proposed CAC equation, let us consider more complex shape on $\Omega = (-2,2) \times (-2,2)$ as shown in Fig. 4.8(a):

$$\phi(x,y,0) = \tanh\left(\frac{2\left(\frac{x^2-y^2}{x^2+y^2}\right)^3 - 1.5\left(\frac{x^2-y^2}{x^2+y^2}\right) + 1.3 - \sqrt{x^2+y^2}}{\sqrt{2}\varepsilon} \right).$$

Here, we use $h = 1/32$ and $\Delta t = 0.1h^2$. The computational outcomes at $t = 1000\Delta t$ for the three distinct Lagrange multipliers $\mathscr{L}(t)$, $\mathscr{L}(t)\sqrt{2F(\phi)}$, and $\mathscr{L}(t)\kappa(\phi)\sqrt{2F(\phi)}$ are shown in Figs. 4.8(b), (c), and (d), respectively. Figures 4.8(e) and (f) depict the zero-level contours of ϕ at $t = 1000\Delta t$ and at the equilibrium state.

We define the numerical equilibrium state as occurring when $\|\phi^{n+1} - \phi^n\|_2 < 10^{-5}$. Numerical solutions reached equilibrium at $n = 3109$ for $\mathscr{L}(t)$, $n = 3128$ for $\mathscr{L}(t)\sqrt{2F(\phi)}$, and $n = 4387$ $\mathscr{L}(t)\kappa(\phi)\sqrt{2F(\phi)}$. The solutions derived from the conventional CAC equations transition into circular shapes due to the motion driven by mean curvature while maintaining mass constraints. In contrast, the proposed CAC equation manages to preserve the initial complex shapes over time, highlighting its effectiveness in maintaining the original geometry throughout the simulation.

Furthermore, Fig. 4.9 illustrates the temporal evolution of the maximum norm of each term in the proposed equation. This graph reveals how the terms interact and balance each other over time. The observed behavior shows that the three terms dynamically counteract each other, maintaining equilibrium throughout the evolution. This balance is crucial for the stability and accuracy of the proposed model, underscoring its robustness in handling complex phase-field dynamics. The data reinforces the effectiveness of the proposed method in preserving the integrity of the solution by ensuring that no single term dominates, thus facilitating a well-balanced numerical simulation.

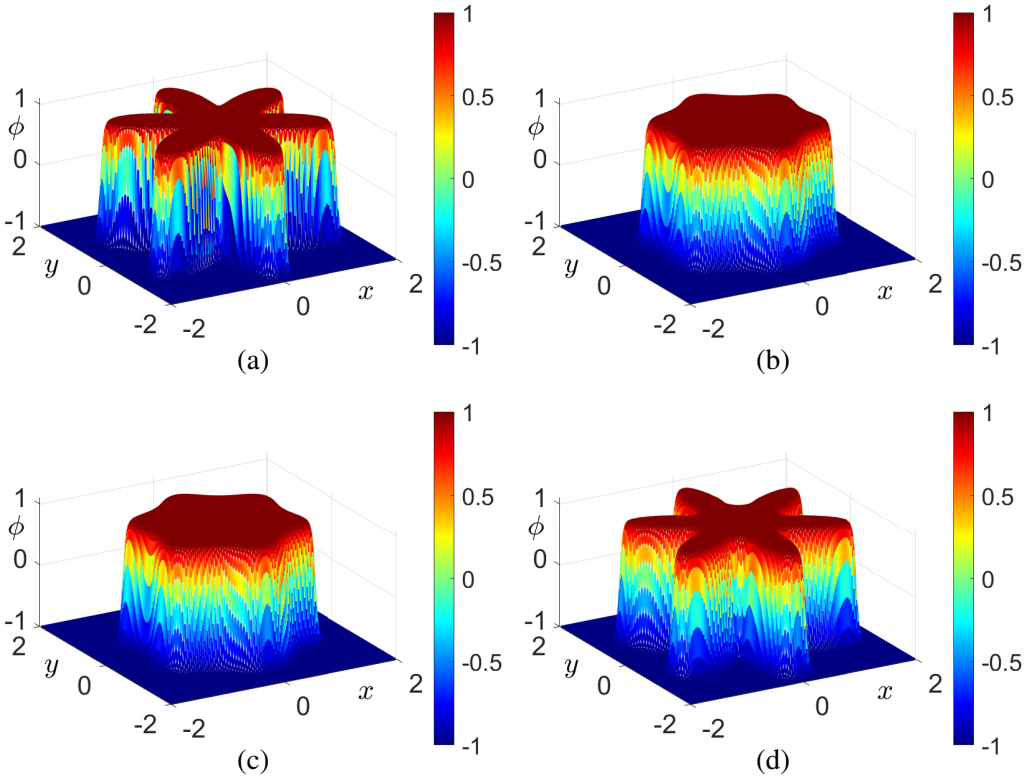


Figure 4.8: (a) depicts the initial condition. (b), (c), and (d) show snapshots of numerical results at time $t = 1000\Delta t$ using various Lagrange multipliers: $\mathscr{L}(t)$, $\mathscr{L}(t)\sqrt{2F(\phi)}$, and $\mathscr{L}(t)\kappa(\phi)\sqrt{2F(\phi)}$, respectively. (e) illustrates the zero-level contours from (b) to (d). (f) displays the zero-level contours of ϕ in the equilibrium state.

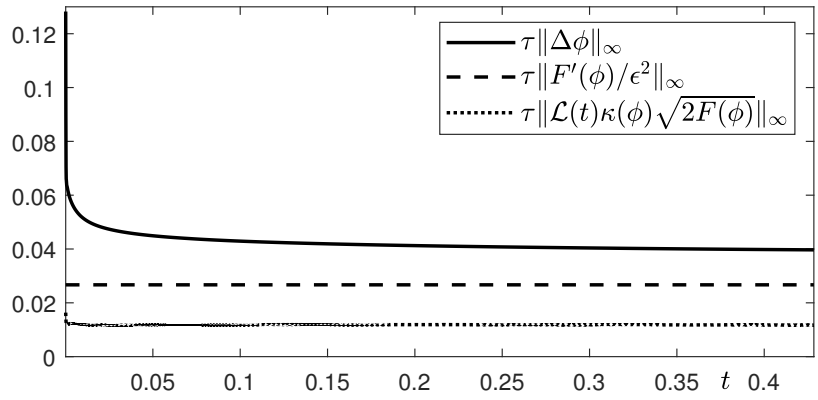


Figure 4.9: The time evolution of the maximum norm of each term in the proposed equation.

Let us consider one circular disk and flat interface on $\Omega = (0,1) \times (0,2)$ as shown in the first row in Fig. 4.10:

$$\phi(x,y,0) = 1 + \tanh\left(\frac{0.2 - \sqrt{(x-0.5)^2 + (y-1.5)^2}}{\sqrt{2}\varepsilon}\right) + \tanh\left(\frac{0.5-y}{\sqrt{2}\varepsilon}\right).$$

Here, we use $h = 1/128$, $\Delta t = 0.1h^2$, and $\varepsilon = \varepsilon_8$. In both the results from the conventional CAC equations, the disk shrinks and disappears because of the motion by mean curvature with mass constraint. The mass spontaneously transfers from the disk to the flat interface. However, the proposed CAC equation preserves the original disk and flat interface shapes as time evolves because the flat interface has zero curvature.

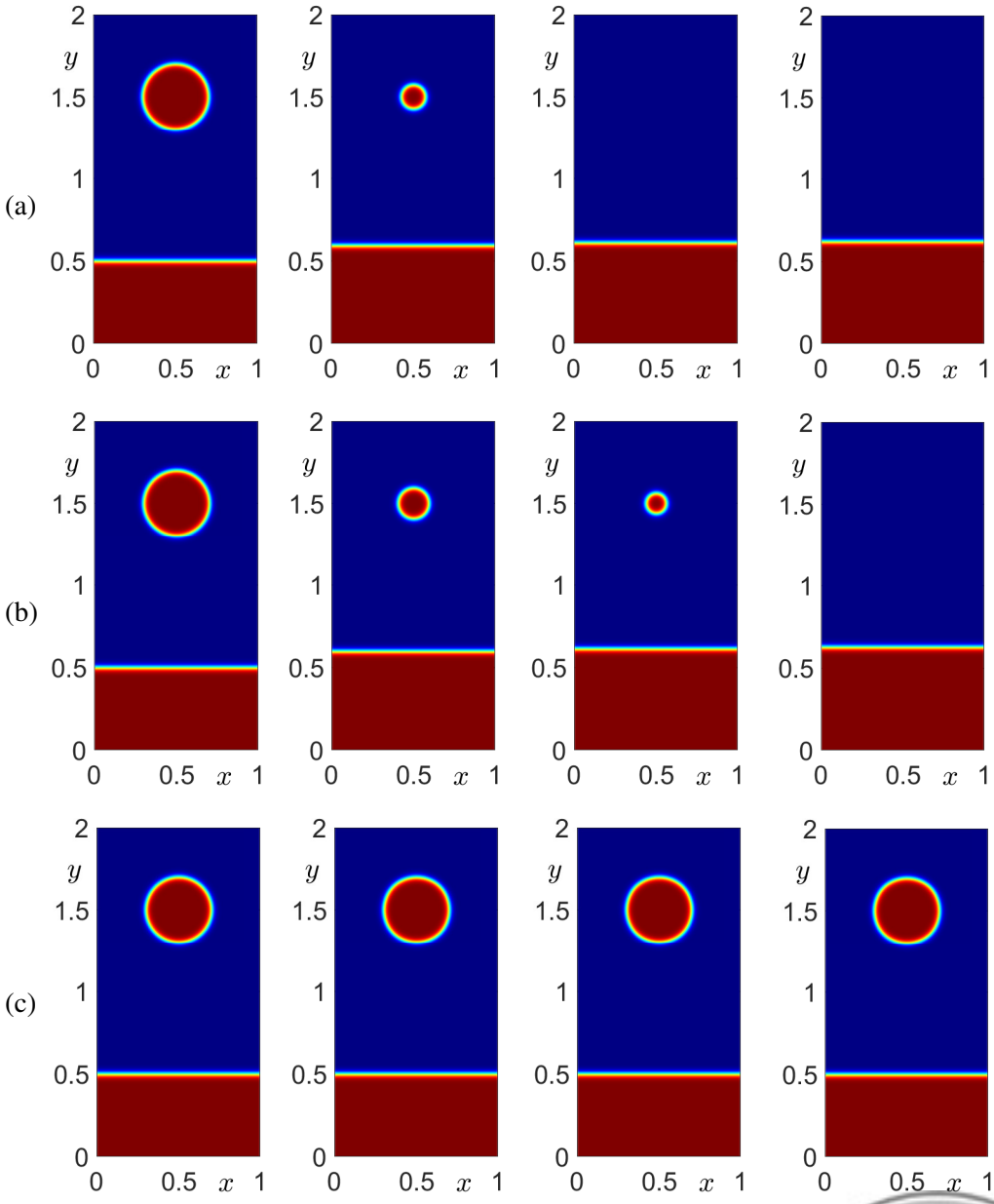


Figure 4.10: Snapshots of evolution of the numerical results using various Lagrange multipliers: (a) $\mathscr{L}(t)$, (b) $\mathscr{L}(t)\sqrt{2F(\phi)}$, and (c) $\mathscr{L}(t)\kappa(\phi)\sqrt{2F(\phi)}$. From left to right, times are at $t = 0$, $5000\Delta t$, $5600\Delta t$, and $6200\Delta t$.

4.3 Deformation of Droplet in swirling flow

In this section, we examine the deformation of droplets subjected to a background swirling flow. Understanding the dynamics of droplet deformation in such a flow is crucial for various applications in fluid mechanics and engineering, including mixing processes, chemical reactions, and material sciences. The governing equation is as follows:

$$\begin{aligned}\frac{\partial \phi(\mathbf{x},t)}{\partial t} + \nabla \cdot [\phi(\mathbf{x},t)\mathbf{u}(\mathbf{x})] &= -\frac{F'(\phi(\mathbf{x},t))}{\varepsilon^2} + \Delta\phi(\mathbf{x},t) \\ &\quad + \mathscr{L}(t)\kappa(\phi(\mathbf{x},t))\sqrt{2F(\phi(\mathbf{x},t))}. \end{aligned} \tag{4.1}$$

We solve Eq. (4.1) by operator splitting method as follows:

$$\phi_t = -\nabla \cdot [\phi(\mathbf{x},t)\mathbf{u}(\mathbf{x})], \tag{4.2}$$

$$\phi_t = \Delta\phi(\mathbf{x},t), \tag{4.3}$$

$$\phi_t = -\frac{F'(\phi(\mathbf{x},t))}{\varepsilon^2}, \tag{4.4}$$

$$\phi_t = \mathscr{L}(t)\kappa(\phi(\mathbf{x},t))\sqrt{2F(\phi(\mathbf{x},t))}. \tag{4.5}$$

Equations (4.3)–(4.5) are solved by the same procedure as described above. The advection term is solved by the FDM.

$$\begin{aligned}\frac{\phi_{ij}^{n+1} - \phi_{ij}^{n}}{\Delta t} &= -\frac{(\phi_{i+1,j}^{n} + \phi_{ij}^{n})u_{i+\frac{1}{2},j} - (\phi_{ij}^{n} + \phi_{i-1,j}^{n})u_{i-\frac{1}{2},j}}{2h} \\ &\quad + \frac{(\phi_{i,j+1}^{n} + \phi_{ij}^{n})v_{i,j+\frac{1}{2}} - (\phi_{ij}^{n} + \phi_{i,j-1}^{n})v_{i,j-\frac{1}{2}}}{2h}. \end{aligned}$$

Here, $\mathbf{u}(\mathbf{x}) = (u(x,y),\ v(x,y))$ is the given velocity field:

$$\begin{aligned} u(x,y) &= -2.5\sin^2(\pi x)\sin(2\pi y), \\ v(x,y) &= 2.5\sin^2(\pi y)\sin(2\pi x). \end{aligned}$$

The initial condition is defined on the computational domain $\Omega = (0,1)\times(0,1)$ as

$$\phi(x,y,0) = \tanh\left(\frac{0.2-\sqrt{(x-0.5)^2+(y-0.7)^2}}{\sqrt{2}\varepsilon}\right).$$

The parameters used are $h = 1/128$, $\Delta t = 0.2h^2$, and $\varepsilon = \varepsilon_8$. For the given u, v, and initial position of interface $(X_k(0),Y_k(0))$, $k = 1,\ldots,N_k$, the exact reference solution of interfacial position $(X_k(t),Y_k(t))$ can be obtained by solving $dX(t)/dt = u(X(t),Y(t))$ and $dY(t)/dt = v(X(t),Y(t))$. Here, we use an improved Euler method with time step $0.2h^2$. Figure 4.11(a), (b), and (c) are the snapshots of interfacial position using $\mathscr{L}(t)$, $\mathscr{L}(t)\sqrt{2F(\phi)}$, and $\mathscr{L}(t)\kappa\sqrt{2F(\phi)}$, respectively. It can be observed that the numerical solution obtained by the proposed model shows good agreement with the exact reference solution.

4.4 Comparison with Cahn–Hilliard equation

The dynamics of the proposed CAC equation are compared to those of the CH equation. The CH equation is given by:

$$\begin{aligned} \frac{\partial\phi(\mathbf{x},t)}{\partial t} &= M\Delta\left(F'(\phi(\mathbf{x},t)) - \varepsilon^2\Delta\phi(\mathbf{x},t)\right),\ \mathbf{x}\in\Omega,\ t\geq 0, \\ \mathbf{n}\cdot\nabla\phi(\mathbf{x},t) &= 0,\ \mathbf{n}\cdot\nabla\Delta\phi(\mathbf{x},t) = 0, \quad \mathbf{x}\in\partial\Omega. \end{aligned}$$

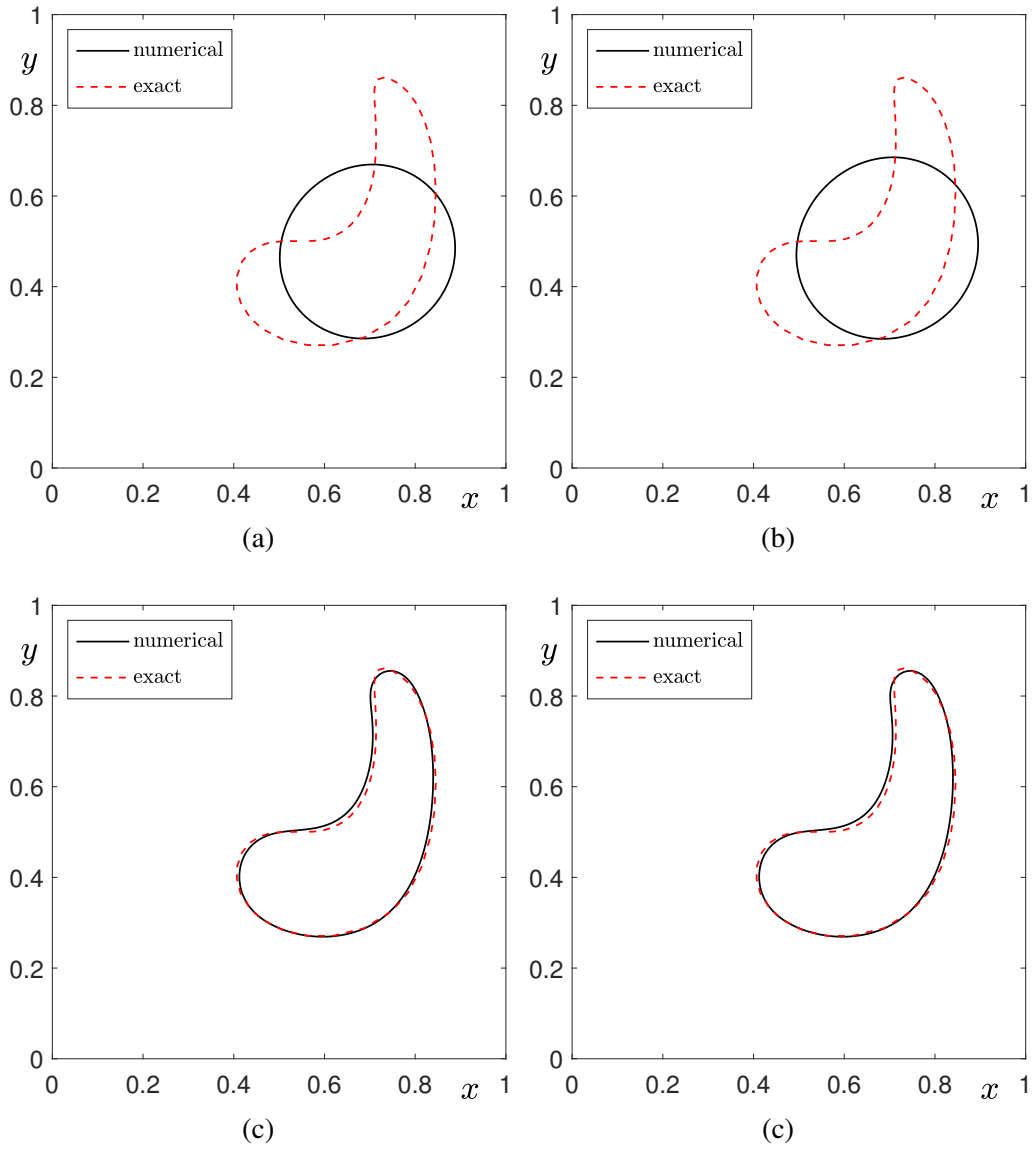


Figure 4.11: (a), (b), and (c) are droplet deformations in a background swirling flow using $\mathscr{L}(t)$, $\mathscr{L}(t)\sqrt{2F(\phi)}$, and $\mathscr{L}(t)\kappa(\phi)\sqrt{2F(\phi)}$ at $t = 0.195$, respectively. The numerical solutions are shown as black solid lines, while the exact solutions are indicated by red dotted lines.

Here, $\Omega \subset \mathbb{R}^d$ ($d = 1, 2, \ldots$) is a bounded domain, t is time, M is the mobility coefficient (in this dissertation, we take $M = 1$ for simplicity), $\mathbf{n}$ is the unit normal vector on $\partial\Omega$, and $\Delta\phi = \nabla \cdot \nabla\phi$ is the Laplacian of ϕ.

For the numerical experiments, the CH equation is solved using the spectral method as described in [67]. Let us consider one circular disk and flat interface on $\Omega = (0,1) \times (0,2)$ as shown in the first row in Fig. 4.10:

$$\phi(x,y,0) = 1 + \tanh\left(\frac{0.2 - \sqrt{(x-0.5)^2 + (y-1.5)^2}}{\sqrt{2}\varepsilon}\right) + \tanh\left(\frac{0.5-y}{\sqrt{2}\varepsilon}\right).$$

Here, we use $h = 1/128$, $\Delta t = 0.1h^2$, and $\varepsilon = \varepsilon_8$.

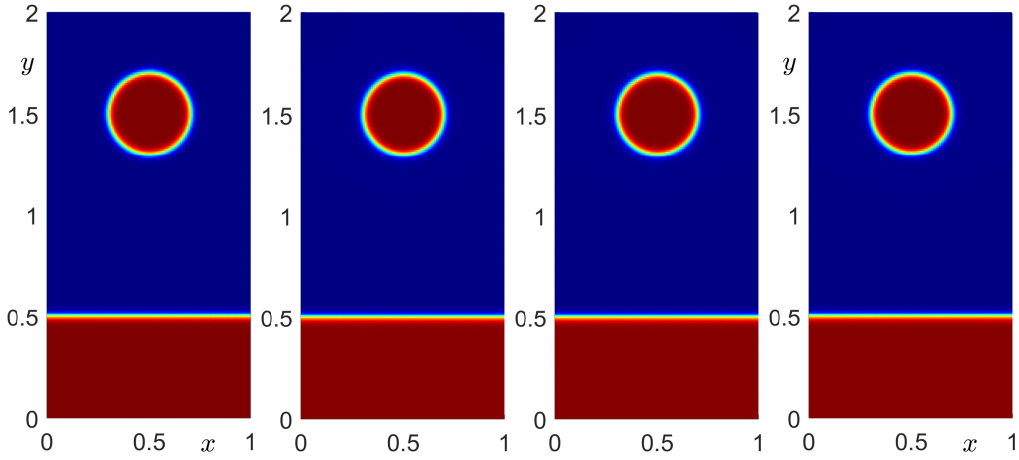


Figure 4.12: Sequential snapshots depicting the numerical results for the CH equation at times $t = 0$, $5000\Delta t$, $5600\Delta t$, and $6200\Delta t$, from left to right.

Figure 4.12 shows the numerical dynamic results of the CH equation for $t = 0$, $5000\Delta t$, $5600\Delta t$, and $6200\Delta t$. In both the results from the conventional CAC equations, the disk shrinks and disappears because of the motion by mean curvature with mass constraint. The mass spontaneously transfers from the disk to the flat interface. However, the proposed CAC equation preserves the original disk and flat interface shapes as time

evolves because the flat interface has zero curvature. Moreover, it shows results similar to the CH dynamics.

We also observe the results for droplets near the flat interface. Let us consider one circular disk and flat interface on $\Omega = (0,1) \times (0,2)$.

$$\phi(x,y,0) = 1 + \tanh\left(\frac{0.2 - \sqrt{(x-0.5)^2 + (y-0.75)^2}}{\sqrt{2}\varepsilon}\right) + \tanh\left(\frac{0.5-y}{\sqrt{2}\varepsilon}\right).$$

Here, we use $h = 1/128$, $\Delta t = 0.1h^2$, and $\varepsilon = \varepsilon_8$.

Figure 4.13 presents the numerical results for both the CH equation and the novel curvature-dependent CAC equation. Theoretically, the CH equation is expected to preserve convexity throughout its evolution. However, the numerical results indicate that the dynamics of the CAC equation closely resemble those predicted by the theoretical CH equation. This comparison underscores the efficacy of the CAC equation in accurately capturing the expected behavior, despite the inherent numerical approximations.

4.5 Isotropic curvature

In Fig. 4.13, it was observed that the numerical solution of the curvature-dependent CAC equation was unstable in the region where the flat interface and the droplet were close. To increase the accuracy of the solution, we consider isotropic curvature. Observe the numerical solution for the initial condition shown in Fig. 4.13 using discrete isotropic curvature. Let us consider one circular disk and flat interface on $\Omega = (0,1) \times (0,2)$.

$$\phi(x,y,0) = 1 + \tanh\left(\frac{0.2 - \sqrt{(x-0.5)^2 + (y-0.75)^2}}{\sqrt{2}\varepsilon}\right) + \tanh\left(\frac{0.5-y}{\sqrt{2}\varepsilon}\right).$$

Here, we use $h = 1/258$, $\Delta t = 0.2h^2$, $\varepsilon = \varepsilon_8$, and $T = 20000\Delta t$.

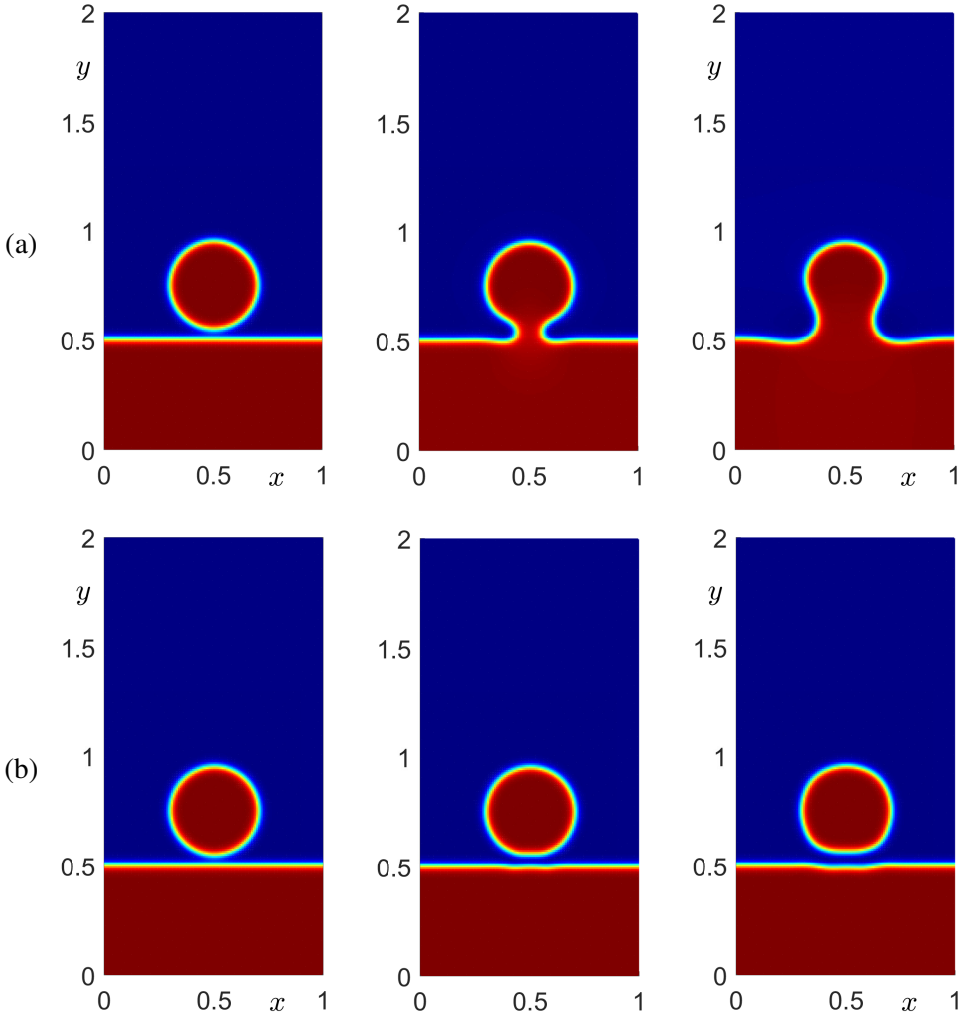


Figure 4.13: (a) CH equation. (b) Curvature-dependent CAC equation. The snapshots, from left to right, correspond to times $t = 0$, $2000\Delta t$, and $10000\Delta t$.

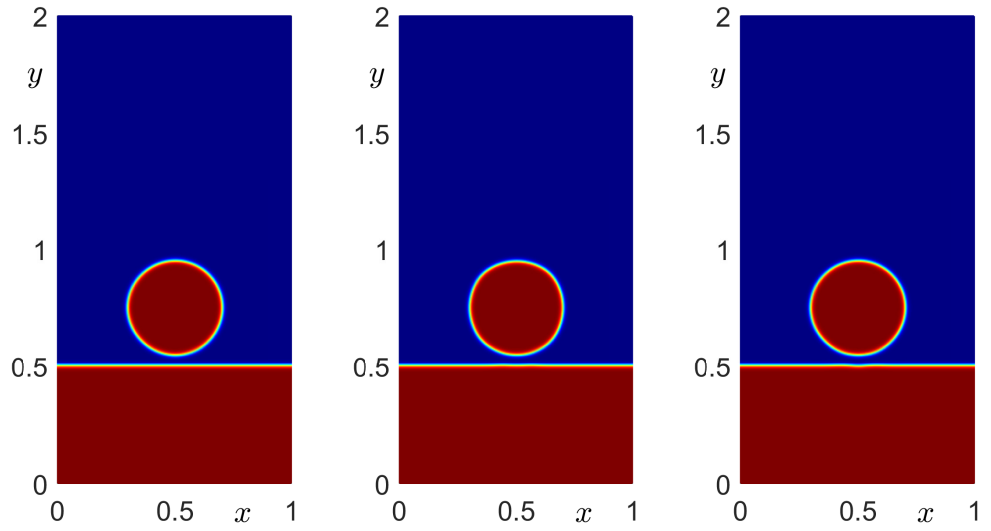


Figure 4.14: (a) Initial condition. (b) Using previous curvature. (c) Using isotropic curvature.

Figure 4.14 shows the results of the curvature-dependent CAC equation using isotropic curvature and a mesh that is twice as fine as that used in Fig. 4.13.

To facilitate a more precise comparison between the results presented in Figs. 4.13 and 4.14, we visually depict the relative difference $\|\phi^0 - \phi^n\|_2 / \|\phi^0\|_2$ in Fig. 4.15. This visualization allows for a clearer assessment of the deviations and improvements in solution accuracy achieved through the application of isotropic curvature in the numerical simulations.

4.6 Rotation of a Zalesak's disk

To conduct a more comprehensive benchmark test, we examine the rotation of Zalesak's disk. This test is a well-established standard in evaluating various interface capturing methods, as demonstrated in studies such as [61, 62]. The initial configuration on

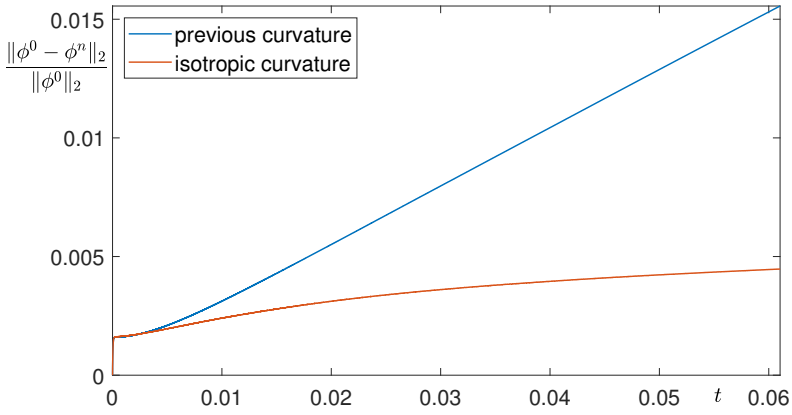
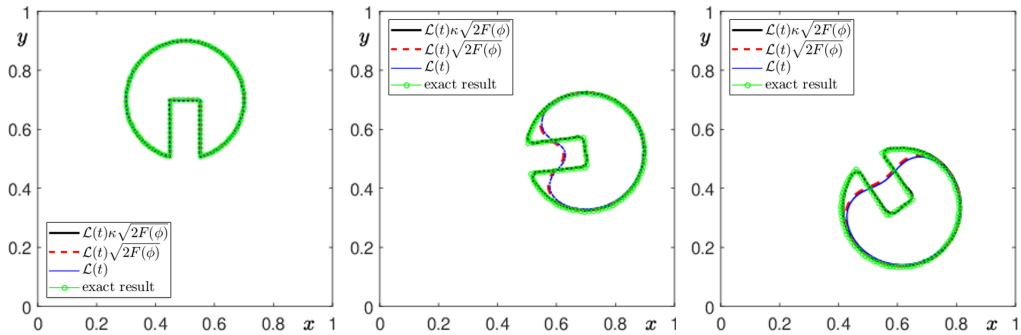


Figure 4.15: The temporal evolution of $\|\phi^0-\phi^n\|_2/\|\phi^0\|_2$

the domain $\Omega=(0,1)\times(0,1)$ is shown in Fig. 4.16(a). The background velocity field is defined by $u(x,y)=600(y-0.5)$, and $v(x,y)=-600(x-0.5)$, with grid step size $h=1/256$, time step $\Delta t=0.01h^2$, and interface width parameter $\varepsilon=\varepsilon_8$. The exact solution, derived from the initial interface position and background velocity field, is computed using the second-order accurate modified Euler method [63]. In Figs. 4.16(a), (b), and (c), we present snapshots of the numerical results at times $t=0,\ 0.0024$, and 0.0043 for the Lagrange multipliers $\mathscr{L}(t)$, $\mathscr{L}(t)\sqrt{2F(\phi)}$, and $\mathscr{L}(t)\kappa\sqrt{2F(\phi)}$, respectively. As evident from these figures, the proposed model effectively preserves the initial shape throughout the simulation.



(a) $t = 0$ (b) $t = 0.0024$ (c) $t = 0.0043$

Figure 4.16: Rotation of a Zalesak's disk with $\mathscr{L}(t)$, $\mathscr{L}(t)\sqrt{2F(\phi)}$, and $\mathscr{L}(t)\kappa(\phi)\sqrt{2F(\phi)}$. The corresponding computational times are indicated below each figure.

Chapter 5. Conclusions

In this paper, we noted the non-conserving mass property of the conventional AC equation through a brief review. To address this issue, we introduced CAC equations. While the two widely used CAC equations follow motion by mean curvature, the proposed new CAC equation employs curvature-dependent Lagrange multipliers, exhibiting superior feature-preserving properties. In contrast to traditional CAC equations, which involve motion by mean curvature with area or volume constraints, the model presented here introduces a different approach. This new model minimizes the dynamics of motion by mean curvature. Instead, it emphasizes the smoothing properties of the interface transition layer. To elaborate, classical CAC equations typically govern the evolution of interfaces under the influence of mean curvature, while enforcing constraints on the area or volume of the phases. However, the novel model we propose shifts focus away from these traditional dynamics. It reduces the role of mean curvature-driven motion, which is known to cause significant morphological changes over time. Instead, the primary feature of our model is the enhanced smoothing effect on the interface transition layer. This property facilitates a more stable and gradual transition between phases, thereby maintaining the integrity of the interface structure without the aggressive changes induced by mean curvature forces. By prioritizing the smoothing characteristic, our model offers a more controlled and stable interface evolution, which can be advantageous in applications re-

quiring precise interface management and reduced topological changes. This approach allows for a clearer understanding and prediction of the interface behavior over time, offering potential improvements in numerical stability and computational efficiency. Consequently, it can serve as a foundational equation for modeling conservative phase-field applications, such as two-phase fluid flows. Based on the OSM, we provided a numerical algorithm. Various numerical methods were applied to solve the diffusion term. Additionally, we conducted a comparative analysis of the results generated by each numerical scheme. This comparative study allowed us to assess the relative performance and accuracy of the different methods employed. In addition to the methodological comparison, we executed a series of numerical experiments to rigorously evaluate the performance of the proposed CAC equation. These experiments were designed to test the algorithm under various conditions and scenarios. The primary focus of our experiments was to verify the enhanced performance of the CAC equation, with particular attention to its ability to preserve structural properties. Through these extensive numerical tests, we confirmed that our proposed CAC equation not only maintains but also enhances the structural integrity of the interface transition layer. This improvement is crucial for applications requiring precise control over interface dynamics and feature preservation. By systematically analyzing the results, we demonstrated the robustness and efficacy of our approach, highlighting its potential for broader application in computational mathematics and related fields. The enhanced performance in preserving structural properties underscores the value of our proposed numerical algorithm and its suitability for complex simulations.

Reference

[1] S.M. Allen, and J.W. Cahn, A microscopic theory for antiphase boundary motion and its application to antiphase domain coarsening, Acta metallurgica, 27(6) (2022) 1085–1095.

[2] S. Kim and J. Kim, Automatic Binary Data Classification Using a Modified Allen–Cahn Equation, International Journal of Pattern Recognition and Artificial Intelligence, 35(04), (2021), 2150013.

[3] D. Lee, S. Kim, H.G. Lee, S. Kwak, J. Wang, and J. Kim, Classification of ternary data using the ternary Allen–Cahn system for small datasets, AIP Advances, 12(6), (2022).

[4] A.L. Bertozzi and A. Flenner, Diffuse interface models on graphs for classification of high dimensional data, SIAM Review, 58(2), (2016), 293–328.

[5] J. Budd, Y. van Gennip, and J. Latz, Classification and image processing with a semi-discrete scheme for fidelity forced Allen–Cahn on graphs, GAMM-Mitteilungen, 44(1), (2021), e202100004.

[6] M. Beneš, V. Chalupecký, K. Mikula, Geometrical image segmentation by the Allen–Cahn equation, Applied Numerical Mathematics, 51(2-3), (2004), 187–205.

[7] D. Lee and S. Lee, Image segmentation based on modified fractional Allen–Cahn equation, Mathematical Problems in Engineering, 2019(1), (2019), 3980181.

[8] C. Liu, Z. Qiao, and Q. Zhang, Multi-phase image segmentation by the Allen–Cahn Chan–Vese model, Computers & Mathematics with Applications, 141, (2023), 207–220.

[9] Z. Rong, L.L. Wang, and X.C. Tai, Adaptive wavelet collocation methods for image segmentation using TV–Allen–Cahn type models, Advances in Computational Mathematics, 38(1), (2013), 101–131.

[10] J. Wang, Z. Han, and J. Kim, An efficient and explicit local image inpainting method using the Allen–Cahn equation, Zeitschrift für angewandte Mathematik und Physik, 75(2), (2024), 44.

[11] Y. Li, D. Jeong, J.I. Choi, S. Lee, and J. Kim, Fast local image inpainting based on the Allen–Cahn model, Digital Signal Processing, 37, (2015), 65–74.

[12] Y. Li, S. Lan, X. Liu, B. Lu, and L. Wang, An efficient volume repairing method by using a modified Allen–Cahn equation, Pattern Recognition, 107, (2020), 107478.

[13] Y. Li, X. Song, S. Kwak, and J. Kim, Weighted 3D volume reconstruction from series of slice data using a modified Allen–Cahn equation, Pattern Recognition, 132, (2022), 108914.

[14] A.L. Brkić and A. Novak, A nonlocal image inpainting problem using the linear Allen–Cahn equation, in Advances in Non-Integer Order Calculus and Its Applications: Proceedings of the 10th International Conference on Non-Integer Order

Calculus and Its Applications 10, Springer International Publishing, (2020), 229–239.

[15] H. Kim, C. Lee, S. Kwak, Y. Hwang, S. Kim, Y. Choi, and J. Kim, Three-dimensional volume reconstruction from multi-slice data using a shape transformation, Computers & Mathematics with Applications, 113, (2022), 52–58.

[16] Z. Han, H. Xu, and J. Wang, A simple shape transformation method based on phase-field model, Computers & Mathematics with Applications, 147, (2023), 121–129.

[17] H. Kim, S. Kang, G. Lee, S. Yoon, and J. Kim, Shape transformation on curved surfaces using a phase-field model, Communications in Nonlinear Science and Numerical Simulation, 133, (2024), 107956.

[18] Q. Li, N. Cui, S. Zheng, and L. Mei, A new Allen–Cahn type two-model phase-field crystal model for fcc ordering and its numerical approximation, Applied Mathematics Letters, 132, (2022), 108211.

[19] X. Xiao and X. Feng, A second-order maximum bound principle preserving operator splitting method for the Allen–Cahn equation with applications in multi-phase systems, Mathematics and Computers in Simulation, 202, (2022), 36–58.

[20] D. Jeong and J. Kim, An explicit hybrid finite difference scheme for the Allen–Cahn equation, Journal of Computational and Applied Mathematics, 340, (2018), 247–255.

[21] X. Jing and Q. Wang, Linear second order energy stable schemes for phase field crystal growth models with nonlocal constraints, Computers & Mathematics with Applications, 79(3), (2020), 764–788.

[22] H.G. Lee and J.Y. Lee, A second order operator splitting method for Allen–Cahn type equations with nonlinear source terms, Physica A: Statistical Mechanics and its Applications, 432, (2015), 24–34.

[23] J. Wang, C. Lee, H. G. Lee, Q. Zhang, J. Yang, S. Yoon, J. Park, and J. Kim, Phase-field modeling and numerical simulation for ice melting, Numerical Mathematics: Theory, Methods and Applications, 14(2), (2021), 540–558.

[24] S. Daubner, P.K. Amos, E. Schoof, J. Santoki, D. Schneider, and B. Nestler, Multiphase-field modeling of spinodal decomposition during intercalation in an Allen–Cahn framework, Physical Review Materials, 5(3), (2021), 035406.

[25] J. Feng, Y. Zhou, and T. Hou, A maximum-principle preserving and unconditionally energy-stable linear second-order finite difference scheme for Allen–Cahn equations, Applied Mathematics Letters, 118, (2021), 107179.

[26] J. Zhao, A revisit of the energy quadratization method with a relaxation technique, Applied Mathematics Letters, 120, (2021), 107331.

[27] J.W. Choi, H.G. Lee, D. Jeong, and J. Kim, An unconditionally gradient stable numerical method for solving the Allen–Cahn equation, Physica A: Statistical Mechanics and its Applications, 388(9), (2009), 1791–1803.

[28] D.S. Lee and J.S. Kim, Mean curvature flow by the Allen–Cahn equation, European Journal of Applied Mathematics, 26(4), (2015), 535–559.

[29] L.C. Evans, H.M. Soner, and P.E. Souganidis, Phase transitions and generalized motion by mean curvature, Communications on Pure and Applied Mathematics, 45(9), (1992), 1097–1123.

[30] C. Lee, H. Kim, S. Yoon, S. Kim, D. Lee, J. Park, S. Kwak, J. Yang, J. Wang, and J. Kim, An unconditionally stable scheme for the Allen–Cahn equation with high-order polynomial free energy, Communications in Nonlinear Science and Numerical Simulation, 95, (2021), 105658.

[31] A. Begmohammadi, R. Haghani-Hassan-Abadi, A. Fakhari, and D. Bolster, Study of phase-field lattice Boltzmann models based on the conservative Allen–Cahn equation, Physical Review E, 102(2), (2020), 023305.

[32] S. Kwak, J. Yang, and J. Kim, A conservative Allen–Cahn equation with a curvature-dependent Lagrange multiplier, Applied Mathematics Letters, 126, (2022), 107838.

[33] J. Li, L. Ju, Y. Cai, and X. Feng, Unconditionally maximum bound principle preserving linear schemes for the conservative Allen–Cahn equation with nonlocal constraint, Journal of Scientific Computing, 87(3), (2021), 1–32.

[34] M. Sugimoto, Y. Sawada, M. Kaneda, and K. Suga, Consistent evaporation formulation for the phase-field lattice Boltzmann method, Physical Review E 103(5), (2021), 053307.

[35] L. Chen and J. Zhao, A novel second-order linear scheme for the Cahn–Hilliard–Navier–Stokes equations, Journal of Computational Physics, 423, (2021), 109782.

[36] C. Liu, F. Frank, C. Thiele, F. O. Alpak, S. Berg, W. Chapman, and B. Riviere, An efficient numerical algorithm for solving viscosity contrast Cahn–Hilliard–Navier–Stokes system in porous media, Journal of Computational Physics, 400, (2020), 108948.

[37] N. Adam, F. Franke, and S. Aland, A simple parallel solution method for the Navier–Stokes Cahn–Hilliard equations, Mathematics, 8(8), (2020), 1224.

[38] J. Rubinstein and P. Sternberg, Nonlocal reaction-diffusion equations and nucleation, IMA Journal of Applied Mathematics, 48(3), (1992), 249–264.

[39] B. Xia, Y. Li, and Z. Li, Second-order unconditionally stable direct methods for Allen–Cahn and conservative Allen–Cahn equations on surfaces, Mathematics, 8(9), (2020), 1486.

[40] A. Shah, M. Sabir, M. Qasim, and P. Bastian, Efficient numerical scheme for solving the Allen–Cahn equation, Numerical Methods for Partial Differential Equations, 34(5), (2018), 1820–1833.

[41] H. Li, Z. Song, and F. Zhang, A reduced-order modified finite difference method preserving unconditional energy-stability for the Allen–Cahn equation, Numerical Methods for Partial Differential Equations, 37(3), (2021), 1869–1885.

[42] X. Wang, J. Kou, and J. Cai, Stabilized energy factorization approach for Allen–Cahn equation with logarithmic Flory–Huggins potential, Journal of Scientific Computing, 82(2), (2020), 25.

[43] H. Li, Z. Song, and J. Hu, Numerical analysis of a second-order IPDGFE method for the Allen–Cahn equation and the curvature-driven geometric flow, Computers & Mathematics with Applications, 86, (2021), 49–62.

[44] M. Brassel and E. Bretin, A modified phase field approximation for mean curvature flow with conservation of the volume, Mathematical Methods in the Applied Sciences, 34(10), (2011), 1157–1180.

[45] Z. Weng and Q. Zhuang, Numerical approximation of the conservative Allen–Cahn equation by operator splitting method, Math. Meth. Appl. Sci., 40(12), (2017), 4462–4480.

[46] Z. Huang, G. Lin, and A.M. Ardekani, Consistent and conservative scheme for incompressible two-phase flows using the conservative Allen–Cahn model, Journal of Computational Physics, 420, (2020), 109718.

[47] V. Joshi and R.K. Jaiman, A positivity preserving and conservative variational scheme for phase-field modeling of two-phase flows, Journal of Computational Physics, 360, (2018), 137–166.

[48] P.H. Chiu, A coupled phase field framework for solving incompressible two-phase flows, Journal of Computational Physics, 392, (2019), 115–140.

[49] D. Lee, The numerical solutions for the energy-dissipative and mass-conservative Allen–Cahn equation, Computers & Mathematics with Applications, 80(1), (2020), 263–284.

[50] M. Okumura, A stable and structure-preserving scheme for a non-local Allen–Cahn equation, Japan Journal of Industrial and Applied Mathematics, 35(3), (2018), 1245–1281.

[51] D. Lee and Y. Kim, Novel mass-conserving Allen–Cahn equation for the boundedness of an order parameter, Communications in Nonlinear Science and Numerical Simulation, 85, (2020), 105224.

[52] X. Mao, V. Joshi, and R. Jaiman, A variational interface-preserving and conservative phase-field method for the surface tension effect in two-phase flows, Journal of Computational Physics, 433, (2021), 110166.

[53] Q. Hong, Y. Gong, J. Zhao, and Q. Wang, Arbitrarily high order structure-preserving algorithms for the Allen–Cahn model with a nonlocal constraint, Applied Numerical Mathematics, 170, (2021), 321–339.

[54] Z. Weng and L. Tang, Analysis of the operator splitting scheme for the Allen–Cahn equation, Numerical Heat Transfer, Part B: Fundamentals, 70(5), (2016), 472–483.

[55] S. MacNamara and G. Strang, Operator splitting, in Splitting methods in communication, imaging, science, and engineering, (2016), 95–114.

[56] H.G. Lee and J.Y. Lee, A semi-analytical Fourier spectral method for the Allen–Cahn equation, Computers & Mathematics with Applications, 68(3), (2014), 174–184.

[57] M. Dehghan, The one-dimensional heat equation subject to a boundary integral specification, Chaos, Solitons & Fractals, 32(2), (2007), 661–675.

[58] Y. Jin, S. Kwak, S. Ham, and J. Kim, A fast and efficient numerical algorithm for image segmentation and denoising, AIMS Mathematics, 9(2), (2024), 5015–5027.

[59] J. Yang, Y. Li, C. Lee, H.G. Lee, S. Kwak, Y. Hwang, X. Xin, and J. Kim, An explicit conservative Saul'yev scheme for the Cahn–Hilliard equation, International Journal of Mechanical Sciences, 217, (2022), 106985.

[60] J. Kim, Phase-field models for multi-component fluid flows, Communications in Computational Physics, 12(3), (2012), 613–661.

[61] D. Kim, C.B. Ivey, F.H. Ham, and L.G. Bravo, An efficient high-resolution Volume-of-Fluid method with low numerical diffusion on unstructured grids, Journal of Computational Physics, 446, (2021), 110606.

[62] M. Gutforth, P.T. Barton, and N. Nikiforakis, An efficient moment-of-fluid interface tracking method, Computers & Fluids, 224, (2021), 104964.

[63] J. Yang, Y. Li, C. Lee, and J. Kim, Conservative Allen–Cahn equation with a nonstandard variable mobility, Acta Mechanica, 231, (2020), 561–576.

[64] J. Kim and H.G. Lee, A new conservative vector-valued Allen–Cahn equation and its fast numerical method, Computer Physics Communications, 221, (2017), 102–108.

[65] S. Aihara, T. Takaki, and N. Takada, Multi-phase-field modeling using a conservative Allen–Cahn equation for multiphase flow, Computers & Fluids, 178, (2019), 141–151.

[66] L. Zheng, S. Zheng, and Q. Zhai, Multiphase flows of N immiscible incompressible fluids: Conservative Allen–Cahn equation and lattice Boltzmann equation method, Physical review E, 101(1), (2020), 013305.

[67] D. Lee, J.Y. Huh, D. Jeong, J. Shin, A. Yun, and J. Kim, Physical, mathematical, and numerical derivations of the Cahn–Hilliard equation, Computational Materials Science, 81, (2014), 216–225.

[68] S. Hochreiter and J. Schmidhuber, Long short-term memory, Neural computation, 9(8), (1997), 1735–1780.

[69] G.H. Hardy, Course of pure mathematics, Courier Dover Publications, (2018).