Phase-field models for multi-phase fluid flows
by
Hyun Geun Lee
A dissertation submitted in partial satisfaction
of the requirements for the degree of
Doctor of Philosophy
in
Mathematics
in the
GRADUATE DIVISION
of the
KOREA UNIVERSITY
Approved:

Committee Member 1
Committee Member 2
Committee Member 3
Committee Member 4
Committee Member 5
Committee in Charge
2011
Contents

Abstract 3

Acknowledgments 5

Chapter 1. Introduction 6
  1.1. Phase-field models 9

Chapter 2. A binary fluid 12
  2.1. Variable density and viscosity 15
  2.2. The Boussinesq approximation 17
  2.3. Regularized Dirac delta functions 23
  2.4. Nondimensionalization 30
  2.5. Snow crystal growth 32
  2.6. The Wulff construction 34
  2.7. Governing equations of snow crystal growth 39
  2.8. Derivation of the thin film equation 41

Chapter 3. The multi-component Cahn–Hilliard system 48

Chapter 4. The multi-component Allen–Cahn system 53

Chapter 5. Numerical solutions 57
5.1. Cahn–Hilliard solver 57  
5.2. Multigrid methods 60  
5.3. Navier–Stokes solver 64  
5.4. Modified Allen–Cahn solver 69  
5.5. The multi-component Cahn–Hilliard solver 71  
5.6. The multi-component Allen–Cahn solver 73  

Chapter 6. Numerical results 76  
6.1. Numerical experiments for regularized Dirac delta functions 76  
6.2. Numerical experiments for snow crystal growth 94  
6.3. Numerical experiments for the multi-component Cahn–Hilliard system 100  
6.4. Numerical experiments for the multi-component Navier–Stokes–Cahn–Hilliard system 106  
6.5. Numerical experiments for the multi-component Allen–Cahn system 111  

Chapter 7. Conclusion 113  
Appendix 115  
Bibliography 121
Abstract

Multi-phase, multi-component fluid flows are used in a wide variety of industrial and scientific applications. Problems involving changes in the topology of interfaces between fluids are difficult to study both from the physical and computational points of view since the fluids are complex with the density, viscosity, diffusivity, and surface tension. In order to investigate the complex interactions, we develop numerical algorithms and perform numerical simulations. In particular, the methods allow us to solve more quickly, efficiently, and accurately the important equations in hydrodynamics: the Cahn–Hilliard, Allen–Cahn, and Navier–Stokes equations.

To model multi-component fluid flows, we consider the vector valued phase-field, velocity, and pressure which are governed by the $N$-component advective Cahn–Hilliard and modified Navier–Stokes equations. For the $N$-component Cahn–Hilliard equation, we present a practically unconditionally gradient stable conservative nonlinear numerical scheme. The scheme is based on a nonlinear splitting method and is solved by an efficient and accurate nonlinear multigrid method. And the scheme allows us to convert the $N$-component Cahn–Hilliard equation into a system of $N-1$ binary Cahn–Hilliard equations and significantly reduces the required computer memory and CPU time. The Cahn–Hilliard–Navier–Stokes equations can be applied to any number of fluid components. And, to investigate the buoyancy driven interpenetration of fluids
with different densities, we implement a time-dependent pressure boundary condition through a time-dependent density field at the boundary. Due to the pressure boundary treatment, we can perform long time evolutions resulting in an equilibrium state.

We present an unconditionally stable second-order hybrid numerical method for solving the Allen–Cahn equation. The proposed method is based on operator splitting techniques. The Allen–Cahn equation was divided into a linear and a nonlinear equation. First, the linear equation was discretized using an implicit Euler’s scheme and the resulting discrete system of equations was solved by a multigrid method. The nonlinear equation was then solved analytically due to the availability of a closed-form solution. In particular, we apply this technique to dendritic growth simulation. A great challenge in the simulation with various supercoolings is the large difference in time and length scales. This introduces a severe time step restriction for stability. The proposed scheme allows the use of a sufficiently large time step without the technical limitations.
Acknowledgments

Throughout my graduate studies I have been supported by a large number of people. First I would like to thank my advisor, Junseok Kim, for his guidance and teaching as well as the opportunity to work on fun and exciting projects.

Also, I offer my regards and blessings to all of those who supported me in any respect during the completion of this thesis. And I consider myself very lucky to work in my research group.

Finally, I am indebted to my family for their support and encouragement.
Chapter 1

Introduction

Multi-phase or multi-component fluid flows are used in a wide variety of applications such as extractors [74], polymer-dispersed liquid crystals [75], polymer blends [76], reactors [77], separators [78], sprays [79], and microfluidic technology [80, 81]. Problems involving changes in the topology of interfaces between fluids are difficult to study both from the physical and computational points of view since the fluids are complex with the density, viscosity, diffusivity, and surface tension. For this reason, mathematical modeling and numerical simulation of multi-phase or multi-component flows is a great challenge.

There are two main approaches to simulate multi-phase or multi-component flows: one is an interface tracking and the other is an interface capturing. In interface tracking methods (front-tracking [82, 25], immersed boundary [83, 84], and volume-of-fluid [83, 85, 86]), Lagrangian particles are used to track the interface and are advected by the velocity field. In interface capturing methods (level-set [87, 88, 40, 89] and phase-field [72, 65, 56]), the interface is implicitly captured by a contour of a particular scalar function.

The equations governing the motion of unsteady, viscous, incompressible, immiscible two-fluid system are the Navier-Stokes equations (the subscript $i$ denotes the $i$th
fluid component):

$$
\rho_i \left( \frac{\partial \mathbf{u}_i}{\partial t} + \mathbf{u}_i \cdot \nabla \mathbf{u}_i \right) = -\nabla p_i + \nabla \cdot [\eta_i (\nabla \mathbf{u}_i + \nabla \mathbf{u}_i^T)] + \rho_i \mathbf{g} \quad \text{in } \Omega_i,
$$

$$
\nabla \cdot \mathbf{u}_i = 0 \quad \text{in } \Omega_i,
$$

where \( \rho_i \) is the density, \( \mathbf{u}_i \) is the velocity, \( p_i \) is the pressure, and \( \eta_i \) is the viscosity of fluid \( i = 1, 2 \), the superscript \( T \) denotes the transpose, and \( \mathbf{g} \) is the gravitational force. A schematic of a two-phase domain is shown in Figure 1.1. \( \Gamma \) is the interface between two fluids, \( \mathbf{n} = (n_1, n_2) \) is the outward unit normal vector to the interface, and \( \mathbf{t} = (t_1, t_2) \) is the unit tangent vector to the interface.

In fluid flow problems with moving interfaces, part of the interface of the computational domain is unknown and must be determined as part of the solution. On the unknown interface, the boundary conditions should be satisfied. The dynamic condition requires that the forces acting on the fluid at the interface be in equilibrium (momentum conservation at the interface). This means that the normal forces on either side of the interface are of equal magnitude and opposite direction, while the forces in the tangential direction are of equal magnitude and direction (neglecting the surface
tension):

\[(\mathbf{n} \cdot \tau)_1 \cdot \mathbf{n} = -(\mathbf{n} \cdot \tau)_2 \cdot \mathbf{n} \quad \text{and} \quad (\mathbf{n} \cdot \tau)_1 \cdot \mathbf{t} = (\mathbf{n} \cdot \tau)_2 \cdot \mathbf{t},\]

where \(\tau\) is the stress tensor (including pressure terms). The kinematic condition requires that the interface be a sharp boundary separating the two fluids that allows no flow through it. This states that the components of the velocity normal (or tangential) to the interface are equal for two fluids:

\[\mathbf{u}_1 \cdot \mathbf{n} = \mathbf{u}_2 \cdot \mathbf{n} \quad \text{and} \quad \mathbf{u}_1 \cdot \mathbf{t} = \mathbf{u}_2 \cdot \mathbf{t}.\]

Consider the effect of surface tension on a fluid interface. From the surface stress boundary condition at an interface \(\Gamma\) between two fluids, we have the normal jump condition

\[p_1 - p_2 + \sigma \kappa = \left(2\eta_1 n_k \frac{\partial u_k}{\partial n}\right)_1 - \left(2\eta_2 n_k \frac{\partial u_k}{\partial n}\right)_2,\]

where \(\sigma\) is the fluid surface tension coefficient and \(\kappa\) is the local surface curvature, \(R_1^{-1} + R_2^{-1}\), where \(R_1\) and \(R_2\) are the principle radii of curvature of the surface. The above equation implies that both the surface tension and viscous stress at the interface contribute to the pressure jump.

For incompressible and immiscible multi-component fluid flows, the governing equations can be written as

\[
\rho \left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u}\right) = -\nabla p + \nabla \cdot \left[\eta(\nabla \mathbf{u} + \nabla \mathbf{u}^T)\right] + \mathbf{SF}_{\text{sing}} + \rho \mathbf{g}, \quad (1.1)
\]

\[
\nabla \cdot \mathbf{u} = 0 \quad \text{in} \quad \Omega = \Omega_1 \cup \Omega_2, \quad (1.2)
\]

where \(\rho(x, t)\) is the density, \(\mathbf{u}(x, t)\) is the velocity, \(p(x, t)\) is the pressure, \(\eta(x, t)\) is the viscosity, \(\mathbf{SF}_{\text{sing}} = -\sigma \kappa \delta_{\Gamma} \mathbf{n}\) is the singular surface tension force (\(\delta_{\Gamma}\) is the surface delta function).
1.1. Phase-field models

Phase-field models are an increasingly popular choice for modeling the motion of multi-phase or multi-component fluid flows [1]. In the phase-field model, sharp fluid interfaces are replaced by thin but nonzero thickness transition regions where the interfacial forces are smoothly distributed [2]. The basic idea is to introduce a conserved order parameter (e.g., mass concentration) that varies continuously over thin interfacial layers and is mostly uniform in the bulk phases (see Fig. 1.2).

![Figure 1.2. A concentration profile across an interface with thickness, ξ.](image)

For density-matched binary liquids (let $\rho = 1$ for simplicity), the coupling of the convective Cahn-Hilliard equation for the mass concentration with a modified momentum equation that includes a phase field-dependent surface force is known as Model H [60]. In the case of fluids with different densities a phase-field model has been proposed by Lowengrub and Truskinovsky [61]. Complex flow morphologies and topological transitions such as coalescence and interface break-up can be captured naturally and in a mass-conservative and energy-dissipative fashion since there is an associated free energy functional. The phase field is governed by the following advective Cahn-Hilliard
1.1. PHASE-FIELD MODELS

**Equation:**

\[
\frac{\partial c}{\partial t} + \mathbf{u} \cdot \nabla c = \nabla \cdot [M(c) \nabla (F'(c) - \epsilon^2 \nabla c)],
\]

where \(M(c) = c(1-c)\) is the mobility, \(F(c) = \frac{1}{4} \epsilon^2 (1-c)^2\) is a Helmholtz free energy that describe the coexistence of immiscible phases, and \(\epsilon\) is a measure of interface thickness and \(\epsilon \sim \xi\) (see Fig. 1.2). It can be shown that in the sharp interface limit \(\epsilon \to 0\), the classical Navier-Stokes system equations and jump conditions are recovered [61]. The singular surface tension force is \(F_{\text{sing}} = -6\sqrt{2}\tau \epsilon \nabla \cdot (\nabla c \otimes \nabla c)\), where \(\tau\) is the surface tension coefficient [61]. An alternative surface tension force formulation based on the CSF is \(F_{\text{sing}} = -6\sqrt{2}\tau \epsilon \nabla \cdot \left( \frac{\nabla c}{|\nabla c|} \right) |\nabla c| \nabla c\), with which one can calculate pressure field directly from the momentum equations.

The advantages of the phase-field method are: (1) topology changes are automatically described; (2) the composition field \(c\) has a physical meaning not only on near interface but also in the bulk phases; (3) complex physics can easily be incorporated into the framework, the methods can be straightforwardly extended to multicomponent systems, and miscible, immiscible, partially miscible, and lamellar phases can be modeled. For example, the binary model can be straightforwardly extended to describe three component flows as follows. Consider a ternary mixture and denote the composition of components 1, 2, and 3, expressed as mass fractions, by \(c_1\), \(c_2\), and \(c_3\), respectively. Because the concentrations sum to unity, only two of them need to be determined, say \(c_1\), \(c_2\). The evolution of \(c_1\) and \(c_2\) is governed by the following advective ternary Cahn-Hilliard equation:

\[
\frac{\partial c_1}{\partial t} + \mathbf{u} \cdot \nabla c_1 = \nabla \cdot \left[ M(c_1, c_2) \nabla \left( \frac{F(c_1, c_2)}{\partial c_1} - \epsilon^2 \nabla c_1 - 0.5 \epsilon^2 \nabla c_2 \right) \right],
\]
\[
\frac{\partial c_2}{\partial t} + \mathbf{u} \cdot \nabla c_2 = \nabla \cdot \left[ M(c_1, c_2) \nabla \left( \frac{F(c_1, c_2)}{\partial c_2} - 0.5 \epsilon^2 \nabla c_1 - \epsilon^2 \nabla c_2 \right) \right],
\]

where \( M(c_1, c_2) = \sum_{i<j}^3 c_i c_j \) is the mobility, \( F(c_1, c_2) = 1/4 \sum_{i<j}^3 c_i^2 c_j^2 \) is the Helmholtz free energy that can be used to model the three immiscible mixtures, and \( c_3 = 1 - c_1 - c_2 \).

In the ternary fluid flows, the singular surface tension force in the momentum should be changed by is \( F_{\text{sing}} = -6 \sqrt{2} \sum_{i=1}^3 \tau_i \nabla \cdot (\nabla c_i \otimes \nabla c_i) \), where the physical surface tension coefficients \( \tau_{ij} \) between two fluids \( i \) and \( j \) are decomposed into the phase specific surface tensions \( \tau_i \) such that \( \tau_{ij} = \tau_i + \tau_j \).

Associated with diffuse-interfaces is a small scale \( \epsilon \), proportional to the width of the interface. In real physical systems describing immiscible fluids, \( \epsilon \) can be vanishingly small. However, for numerical accuracy \( \epsilon \) must be at least a few grid lengths in size. This can make computations expensive. One way of ameliorating this problem is to adaptively refine the grid only near the transition layer. Such methods are under development by various research groups.

Phase-field methods have been used to model viscoelastic flow, thermocapillary flow, spinodal decomposition, the mixing and interfacial stretching, in a shear flow, droplet breakup process, wave-breaking and sloshing, the fluid motion near a moving contact line, and the nucleation and annihilation of an equilibrium droplet (see the references in the review paper [1]).
Chapter 2

A binary fluid

We consider the incompressible flows of two immiscible fluids (1 and 2) of different density and viscosity. The phase-field is defined to be either the difference between \[\phi = \frac{m_1 - m_2}{m_1 + m_2}\] or the fraction of one of \[\frac{m_1}{m_1 + m_2}\] the concentrations of the two mixtures.

\[
\phi = \frac{m_1 - m_2}{m_1 + m_2} \quad \text{or} \quad c = \frac{m_1}{m_1 + m_2}, \tag{2.1}
\]

where \(m_1\) and \(m_2\) are the masses of the fluids 1 and 2. We note that \(-1 \leq \phi \leq 1\) and \(0 \leq c \leq 1\). Throughout this thesis, we will use \(\phi\) and \(c\) as phase-fields defined by Equation (2.1). The phase-field is also called by an order parameter \[63\]. The governing equation for the phase-field is the advective Cahn–Hilliard (CH) equation:

\[
\frac{\partial c}{\partial t} + \mathbf{u} \cdot \nabla c = \nabla \cdot (M(c)\nabla \mu), \tag{2.2}
\]

\[
\mu = F'(c) - \epsilon^2 \Delta c, \tag{2.3}
\]

where \(\mathbf{u}\) is the bulk velocity and \(M(c)\) is the phase-field dependent mobility. \(M(c) = c(1-c)\) is used in \[72\], \(M(\phi) = 1 - \phi^2\) is used in \[55\], and \(M(c) = 1\) is used in \[6\]. When we have a constant mobility, the CH dynamics is controlled by bulk diffusion. When we have a variable mobility, the bulk diffusion is severely reduced, which corresponds to the interface-diffusion-controlled dynamics, i.e., the coarsening process is mainly due to the diffusion along the interface between the two phases. \(F(c) = 0.25c^2(1 - c)^2\) is
the Helmholtz free energy (see Figure 2.1). \( \epsilon \) is a positive constant, which is related to the thickness of the interfacial transition of the phase-field.

\[ F(c) = 0.25c^2(1-c)^2. \]

**Figure 2.1.** A double well potential,\( F(c) = 0.25c^2(1-c)^2. \)

The CH equation without the flow arises from the Ginzburg–Landau free energy

\[ \mathcal{E}(c) := \int_{\Omega} \left( F(c) + \frac{\epsilon^2}{2} |\nabla c|^2 \right) \, dx, \tag{2.4} \]

where \( \Omega \) is the region of space occupied by the system. To obtain the CH equation with a variable mobility, one introduces a chemical potential \( \mu \) as the variational derivative of \( \mathcal{E} \),

\[ \mu := \frac{\delta \mathcal{E}}{\delta c} = F'(c) - \epsilon^2 \Delta c \]

and defines the flux, \( J := -M(c) \nabla \mu. \) As a consequence of mass conservation, we have

\[ \frac{\partial c}{\partial t} = -\nabla \cdot J, \]

which is the CH equation with a variable mobility. The natural and no-flux boundary conditions are

\[ \frac{\partial c}{\partial n} = J \cdot n = 0 \text{ on } \partial \Omega, \text{ where } n \text{ is normal to } \partial \Omega. \tag{2.5} \]
We differentiate the energy $\mathcal{E}$ and the total mass $\int_{\Omega} c \, dx$ to get

$$\frac{d}{dt} \mathcal{E}(t) = \int_{\Omega} (F'(c)t + \epsilon^2 \nabla c \cdot \nabla c_t) \, dx = \int_{\Omega} \mu c_t \, dx = \int_{\Omega} \mu \nabla \cdot (M(c) \nabla \mu) \, dx$$

$$= \int_{\partial \Omega} \mu M(c) \frac{\partial \mu}{\partial n} \, ds - \int_{\Omega} \nabla \mu \cdot (M(c) \nabla \mu) \, dx = - \int_{\Omega} M(c) |\nabla \mu|^2 \, dx \leq 0$$

and

$$\frac{d}{dt} \int_{\Omega} c \, dx = \int_{\Omega} c_t \, dx = \int_{\Omega} \nabla \cdot (M(c) \nabla \mu) \, dx = \int_{\partial \Omega} M(c) \frac{\partial \mu}{\partial n} \, ds = 0,$$

where we used the no-flux boundary condition (2.5). Therefore, the total energy is non-increasing in time and the total mass is conserved. If we put all the equations (1.1), (1.2), (2.2), and (2.3) together, then we have

$$\rho(c)(u_t + u \cdot \nabla u) = -\nabla p + \nabla \cdot [\eta(c)(\nabla u + (\nabla u)^T)] + SF + \rho(c)g, \quad (2.6)$$

$$\nabla \cdot u = 0, \quad (2.7)$$

$$c_t + u \cdot \nabla c = \nabla \cdot (M(c) \nabla \mu), \quad (2.8)$$

$$\mu = F'(c) - \epsilon^2 \Delta c. \quad (2.9)$$

Singular force $SF_{sing}$ becomes a regular force $SF$ in terms of phase-field.
2.1. Variable density and viscosity

The density and viscosity are linear functions of the phase-field \([55, 72]\). Let \(\rho_1\) and \(\rho_2\) be the densities of each fluid in the mixture, then the variable density of the mixture is defined as

\[
\rho(c) = \rho_1 c + \rho_2 (1 - c) \quad \text{or} \quad \rho(\phi) = \rho_1 \frac{1 + \phi}{2} + \rho_2 \frac{1 - \phi}{2}.
\]

Similarly, the variable viscosity of the mixture is defined as

\[
\eta(c) = \eta_1 c + \eta_2 (1 - c) \quad \text{or} \quad \eta(\phi) = \eta_1 \frac{1 + \phi}{2} + \eta_2 \frac{1 - \phi}{2}.
\]

Then, if we use the linear interpolation, the density and viscosity change across the interface with the same profiles to the scaled phase-field function. In [16], the harmonic interpolation for the variable density and viscosity is used.

\[
\frac{1}{\rho(\phi)} = \frac{1 + \phi}{2\rho_1} + \frac{1 - \phi}{2\rho_2} \quad \text{and} \quad \frac{1}{\eta(\phi)} = \frac{1 + \phi}{2\eta_1} + \frac{1 - \phi}{2\eta_2}.
\]

For the density, the harmonic interpolation is used and the linear interpolation is used for the viscosity in [7].

\[
\frac{1}{\rho(\phi)} = \frac{1 + \phi}{2\rho_1} + \frac{1 - \phi}{2\rho_2} \quad \text{and} \quad \eta(\phi) = \frac{1 + \phi}{2} \eta_1 + \frac{1 - \phi}{2} \eta_2.
\]

One of the reasons to choose the harmonic interpolation is that the solution of the Cahn-Hilliard equation does not satisfy the maximal principle [16]. Hence, the linear interpolation can not be guaranteed to be bounded away from zero. However, due to the \(L^\infty\)-bound of the solution [8], the harmonic interpolations lead to desired properties. In Figure 2.2, solid and dashed lines represent linear and harmonic interpolations of density, respectively. If the value of \(c\) is negative, then the linear interpolated density
can be negative which is not a physical value. However, if the harmonic interpolation is used, then we always have positive values of density.

Figure 2.2. Linear (solid line) and harmonic (dotted line) interpolations of density $\rho$. 
2.2. The Boussinesq approximation

In fluids, the dynamic variables typically required to describe the motion are the velocity, pressure, viscosity, and density. The density is a very important quantity of fluids since it determines fluid properties such as compressibility. Because of the complexity of density variations in fluids, some assumptions have been introduced to reduce the degree of the complexity. One of these assumptions is the Boussinesq approximation [9], which is used in the field of buoyancy-driven flows.

Assume that the densities \( \rho_1 \) and \( \rho_2 \) (\( \rho_1 \) and \( \rho_2 \) are the densities of the heavier and lighter fluid, respectively) are uniform on each side of the interface, and that the density difference is small in the sense that the Atwood number \( At = (\rho_1 - \rho_2)/(\rho_1 + \rho_2) \) is small. The Boussinesq approximation can be made in this case and is also related to the assumption that all of the accelerations of flow are small compared to the gravitational acceleration. In this approximation, the background density can be treated as a constant \( \rho_\ast \), i.e., the variation of background density is neglected. And the difference between the actual density and \( \rho_\ast \) will contribute only to the buoyancy force term of the momentum equation [10]. Because of its simplicity in practical implementations (we solve a constant instead of a variable coefficient Poisson equation), the Boussinesq approximation is employed in many previous papers [11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24].

Aref and Tryggvason [13] proposed a model for the development of the Rayleigh-Taylor instability in the Boussinesq approximation using concentrations of vorticity along the interface. In their case, the density was not coupled to a scalar field, and the buoyancy term in the Boussinesq Navier-Stokes equations was the only term related to the weak stratification. Han and Tryggvason [14] examined the deformation and
breakup of axisymmetric drops, accelerated by a constant body force, for small density differences between the drops and the surrounding fluid. In their work, a density ratio was $\rho_1/\rho_2 = 1.15$ and a front tracking numerical technique was used to solve the unsteady Navier-Stokes equations for both the drops and the surrounding fluid. They showed, in the Boussinesq limit, as the Eötvös number increases the drops break up in a backward facing bag, a transient breakup, and a forward facing bag mode. Young et al. [15] investigated the miscible Rayleigh-Taylor instability in both two and three dimensions. In their work, the fluid was assumed incompressible under the Boussinesq approximation to make the problem more tractable. They found the three-dimensional mixing zone expands two times faster than the two-dimensional mixing zone through the simulation of randomly perturbed interfaces and identified three phases of evolution for the three-dimensional mixing zone: the free-falling phase, the mixing phase, and the another free-falling phase. Liu and Shen [16] modeled the mixture of two incompressible fluids with a phase field model. A semi-discrete Fourier-spectral method for the numerical approximation of a Navier-Stokes system coupled with a Cahn-Hilliard equation was proposed and analyzed. They used the Boussinesq approximation to model the case where the two fluids have different densities.

Vladimirova and Rosner studied the fully nonlinear behavior of premixed flames in a gravitationally stratified medium, subject to the Boussinesq approximation in [18, 19]. Vladimirova [20] simulated a bubble of reaction products rising in the reactant fluid under the influence of gravity using the Boussinesq buoyancy approximation. The Atwood numbers in their experiments were in the range of $0.075 \sim 0.16$. The author showed the evolution of the bubble can be divided into two stages: the bubble grows
radially in an essentially motionless fluid in the first stage and the bubble rises and is
distorted by the flow during the second stage. Celani et al. [21] studied the Rayleigh-
Taylor instability of two immiscible fluids in the limit of small Atwood numbers by
means of a phase-field description. They analytically rederived the known gravity-
capillary dispersion relation in the limit of vanishing mixing energy density and capillary
width. And numerical results were compared with known analytical results, both for
the linearly stable and unstable cases, and for the weakly nonlinear stages of the latter.
Forbes [22] studied the development of the Rayleigh-Taylor instability for inviscid and
viscous fluids. The author used the approximate Boussinesq approach rather than the
full Navier-Stokes equations of viscous flow and the density ratio was $\rho_1/\rho_2 = 1.05$.
A separate spectral method was presented to study the Rayleigh-Taylor instability in
a viscous Boussinesq fluid. The results were shown to agree closely with the inviscid
calculations for small to moderate times. In [24], Boffetta et al. investigated the
Rayleigh-Taylor turbulence in three dimensions at small Atwood number and at Prandtl
number one by means of high resolution direct numerical simulations of the Boussinesq
equations. The authors extended the mean-field analysis for velocity and temperature
fluctuations and showed that small-scale velocity and temperature fluctuations develop
intermittent distributions with structure function scaling exponents consistent with
Navier-Stokes turbulence advecting a passive scalar.

In the Boussinesq approximation, the background density can be treated as a con-
stant $\rho_*$, i.e., the variation of background density is neglected. And the difference
between the actual density and $\rho_*$ will contribute only to the buoyancy force term of
2.2. THE BOUSSINESQ APPROXIMATION

the momentum equation [10]. Hence, Equation (2.6) becomes

\[
\rho_*(u_t + u \cdot \nabla u) = -\nabla p + \nabla \cdot [\eta(\phi)(\nabla u + \nabla u^T)] + SF + \rho(\phi)g, \tag{2.10}
\]

where \( \rho_* = (\rho_1 + \rho_2)/2 \) is the mean density [11, 21].

If we add and subtract the term \( \rho_* g \) to and from Equation (2.10), we get

\[
\rho_*(u_t + u \cdot \nabla u) = -\nabla p + \rho_* g + \nabla \cdot [\eta(\phi)(\nabla u + \nabla u^T)] + SF + (\rho(\phi) - \rho_*)g. \tag{2.11}
\]

In the two-dimensional case, we can write Equation (2.11) as follows:

\[
\rho_*(u_t + u \cdot \nabla u) = -\nabla (p + \rho_* gy) + \nabla \cdot [\eta(\phi)(\nabla u + \nabla u^T)] + SF + (\rho(\phi) - \rho_*)g. \tag{2.12}
\]

If we reset the pressure field as \( p = p + \rho_* gy \) and divide by \( \rho_* \), then Equation (2.12) becomes

\[
u_t + u \cdot \nabla u = -\frac{1}{\rho_*} \nabla p + \frac{1}{\rho_*} \nabla \cdot [\eta(\phi)(\nabla u + \nabla u^T)] + \frac{1}{\rho_*} SF + \frac{\rho(\phi) - \rho_*}{\rho_*} g.
\]

The buoyancy contribution can be rewritten in terms of \( \rho_1, \rho_2, \) and \( \phi \) as:

\[
\frac{\rho(\phi) - \rho_*}{\rho_*} g = \frac{\rho_1(1+\phi)}{\rho_*} + \rho_2(1-\phi) - \frac{\rho_*}{\rho_*} g = \text{At}\phi g.
\]

On the other hand, applying the similar procedure to the case of the variable density model, the buoyancy contribution in Equation (2.6) can be rewritten as

\[
\frac{\rho(\phi) - \rho_*}{\rho(\phi)} g = \frac{\text{At}\phi}{1 + \text{At}\phi} g.
\]

In our phase-field model, \( \phi \) varies from \(-1\) to \(1\). Note that \( \text{At}\phi \) (the Boussinesq case) and \( \text{At}\phi/(1 + \text{At}\phi) \) (the variable density case) are linear and nonlinear functions with respect to the phase-field \( \phi \). Figure 2.3 shows profiles of \( \text{At}\phi \) and \( \text{At}\phi/(1 + \text{At}\phi) \) for two different Atwood numbers \( \text{At} = 0.01 \) and \( \text{At} = 0.5 \). For \( \text{At} = 0.01 \), there is almost no difference between \( \text{At}\phi \) and \( \text{At}\phi/(1 + \text{At}\phi) \). And both \( \text{At}\phi \) and \( \text{At}\phi/(1 + \text{At}\phi) \) are nearly
zero. But, for $At = 0.5$, there is a difference between $At\phi$ and $At\phi/(1 + At\phi)$. This difference implies the difference of the buoyancy force. Figures 2.4 (a) and (b) show the buoyancy force for the Boussinesq and the variable density models with $At = 0.5$, respectively. In the case of the Boussinesq model, the buoyancy force is symmetric along the interface. But, in the case of the variable density model, the buoyancy force is not symmetric along the interface although the phase-field is symmetric. This difference of the buoyancy force causes a difference of the interface evolution for the Boussinesq and the variable density models. If the density variation goes to zero, i.e., $At \ll 0$, then the difference between the Boussinesq and the non-Boussinesq fluids disappears. But in general there is a difference between the Boussinesq and the non-Boussinesq fluids.

![Figure 2.3](image_url)

**Figure 2.3.** Profiles of $At\phi$ (the Boussinesq case) and $At\phi/(1 + At\phi)$ (the variable density case) for two different Atwood numbers $At_1 = 0.01$ and $At_2 = 0.5$. 
Figure 2.4. The buoyancy force for (a) the Boussinesq approximation and (b) the variable density models with $At = 0.5$. This difference causes a difference of the interface evolution for the two different models.
2.3. Regularized Dirac delta functions

Many numerical techniques, including immersed boundary [90, 91, 92, 93, 94, 95, 96, 97], volume-of-fluid [98, 99, 100, 101, 102, 103, 104, 105, 106, 107], and level-set [108, 109, 110, 111, 112, 113, 114, 115], use the concept of a regularized Dirac delta function to account for interfacial effects and many previous studies show that an appropriate delta function is required to obtain more accurate results.

In the immersed boundary method, the elastic boundary is represented by a set of Lagrangian points and the singular force at the Lagrangian points is determined by the generalized Hooke’s law. This force is spread to the surrounding Eulerian points using a delta function. In the immersed boundary method, most commonly used delta functions are 2-point [92, 95, 97], 3-point [94, 95, 97], 6-point [90, 91, 95, 96], 4-point cosine [93, 97], and 4-point [91, 93, 95, 97] functions. For all \( r \), where \( r \) is the parameter representing the position of the submerged boundary point and is scaled with respect to the grid size \( h \), the one-dimensional delta functions are listed below.

- **2-point delta function**
  \[
  \delta(r) = \begin{cases} 
  1 - |r| & \text{if } |r| \leq 1, \\
  0 & \text{otherwise.}
  \end{cases}
  \]

- **3-point delta function**
  \[
  \delta(r) = \begin{cases} 
  \frac{1}{3}(1 + \sqrt{-3r^2 + 1}) & \text{if } |r| \leq 0.5, \\
  \frac{1}{6}(5 - 3|r| - \sqrt{-3(1 - |r|^2) + 1}) & \text{if } 0.5 \leq |r| \leq 1.5, \\
  0 & \text{otherwise.}
  \end{cases}
  \]

- **6-point delta function**
  \[
  \delta(r) = \begin{cases} 
  \frac{61}{172} - \frac{11}{46}|r| - \frac{11}{55}|r|^2 + \frac{1}{12}|r|^3 + \frac{\sqrt{3}}{36}(243 + 1584|r| - 748|r|^2 - 1560|r|^3 + 500|r|^4 + 336|r|^5 - 112|r|^6)^{1/2} & \text{if } 0 \leq |r| \leq 1, \\
  \frac{21}{16} + \frac{7}{32}|r| - \frac{7}{8}|r|^2 + \frac{1}{4}|r|^3 - \frac{3}{2}\delta(|r| - 1) & \text{if } 1 \leq |r| \leq 2, \\
  \frac{3}{5} - \frac{21}{32}|r| + \frac{3}{8}|r|^2 - \frac{1}{12}|r|^3 + \frac{1}{2}\delta(|r| - 2) & \text{if } 2 \leq |r| \leq 3, \\
  0 & \text{otherwise.}
  \end{cases}
  \]
• 4-point cosine delta function

\[ \delta(r) = \begin{cases} 
\frac{1}{4}(1 + \cos(\frac{\pi r}{2})) & \text{if } |r| \leq 2, \\
0 & \text{otherwise.} 
\end{cases} \]

• 4-point delta function

\[ \delta(r) = \begin{cases} 
\frac{1}{8}(3 - 2|r| + \sqrt{1 + 4|r| - 4r^2}) & \text{if } 0 \leq |r| \leq 1, \\
\frac{1}{8}(5 - 2|r| - \sqrt{-7 + 12|r| - 4r^2}) & \text{if } 1 \leq |r| \leq 2, \\
0 & \text{otherwise.} 
\end{cases} \]

The above five types of delta functions are shown in Figure 2.5. Shin et al. [95] analyzed the stability regimes of the feedback forcing gains in the feedback forcing method for several types of delta functions and showed that non-growing oscillations became smaller for the delta function supported by more points. Yang et al. [97] found that the non-physical oscillations are mainly due to the fact that the derivatives of the regular discrete delta functions do not satisfy certain moment conditions and demonstrated that the smoothed discrete delta functions can effectively suppress the non-physical oscillations in the volume forces and improve the accuracy of the immersed boundary method with direct forcing in moving boundary simulations.
The volume-of-fluid (VOF) method was proposed by Hirt and Nichols [30]. In VOF method, the interface is reconstructed from the values of a color function which represents the volume fraction of one of the fluids in each cell. The continuum surface force (CSF) of Brackbill et al. [120] has been widely used to model surface tension in multi-phase fluid flows in VOF method. In the CSF model [100, 101, 107, 55, 39, 63, 121], the surface tension force is converted into a volume force via a delta function: \( f = \sigma \kappa \mathbf{n} \delta \), where \( \sigma \) is the surface tension coefficient, \( \kappa \) is the curvature, \( \mathbf{n} \) is the normal to the surface, and \( \delta \) is a delta function. In VOF method, the most commonly used delta function is \( \delta(\tilde{c}) = |\nabla \tilde{c}| \), where \( \tilde{c} \) is a smoothed version of the volume fraction.

The CSF model is simple and robust, and involves only the solving of a field equation for a smoothed phase field \( \tilde{c} \). However, the method is known to produce strong and spurious currents near the interface. For this reason, many researchers have developed new methods to reduce spurious currents [104, 107, 122, 123, 124]. Meier et al. [104] reduced spurious currents using the piecewise-linear interface construction (PLIC-VOF) method. In [124], a parabolic reconstruction of surface tension (PROST) algorithm is used to gain higher order accuracy for the surface tension force.

In the level-set method, first devised and introduced by Osher and Sethian in [125], delta functions are often used to distribute a singular force or compute a surface area [113, 126, 127, 128]. Most commonly used delta functions are listed below. Here, \( \epsilon \) is proportional to the grid size, i.e., \( \epsilon = mh \) for a positive number \( m \).

- Delta function in [108, 110, 112, 114, 115]

\[
\delta_\epsilon(x) = \begin{cases} 
\frac{1}{\epsilon} \left(1 - \frac{|x|}{\epsilon}\right) & \text{if } |x| \leq \epsilon, \\
0 & \text{otherwise}. 
\end{cases}
\]
2.3. REGULARIZED DIRAC DELTA FUNCTIONS

• Delta function in [108, 109, 110, 111, 113, 114, 115]

\[ \delta_\epsilon(x) = \begin{cases} \frac{1}{2\epsilon} \left(1 + \cos\left(\frac{\pi x}{\epsilon}\right)\right) & \text{if } |x| \leq \epsilon, \\ 0 & \text{otherwise.} \end{cases} \]

Tornberg and Engquist [129] pointed out that the most common technique for regularization of delta functions in level-set simulations is not consistent and analyzed the accuracy of regularization of delta functions. Smereka [112] presented methods for constructing consistent approximations to Dirac delta measures concentrated on piecewise smooth curves or surfaces. Towers [130] proposed second-order finite difference methods for approximating Heaviside functions and showed that the methods are more accurate than a commonly used approximate Heaviside function.

We now present eight types of delta functions for phase-field models. From the Ginzburg–Landau free energy (2.4), the phase-field \( c \) at the equilibrium state satisfies the following equation:

\[ \frac{1}{4} c^2 (1 - c)^2 = \frac{\epsilon^2}{2} |
\nabla c|^2. \tag{2.13} \]

By using Equation (2.13) and delta functions used in phase-field models [65, 131, 132, 133], we get new delta functions for the present model. In this thesis, eight types of delta functions are chosen:

• Delta function in [65, 131] \( \delta_1(c) = 6\sqrt{2}\epsilon|\nabla c|^2. \)

• New delta function \( \delta_2(c) = 3\sqrt{2}\epsilon c(1 - c)^2/\epsilon. \)

• Delta function in [132, 133] \( \delta_3(c) = 0.5(\delta_1(c) + \delta_2(c)). \)

• Delta function in [65] \( \delta_4(c) = 30\sqrt{2}\epsilon c(1 - c)|\nabla c|^2. \)

• New delta function \( \delta_5(c) = c(1 - c)/\sqrt{2}\epsilon. \)

• New delta function \( \delta_6(c) = |
\nabla c|. \)
2.3. REGULARIZED DIRAC DELTA FUNCTIONS

- New delta function $\delta_7(c) = 0.5(\delta_5(c) + \delta_6(c))$.
- New delta function $\delta_8(c) = 630\sqrt{2} c^3(1 - c^3)|\nabla c|^2$.

For a flat interface with an equilibrium profile,

$$\int_{-\infty}^{\infty} \delta_i(c^{eq}) \, dx = 1 \text{ for all } i = 1, 2, \ldots, 8,$$

where $c^{eq}(x, y) = 0.5(1+\tanh(x/(2\sqrt{\epsilon})))$ is an equilibrium profile in the infinite domain when the chemical potential is given by Equation (2.3) [64] and it is a good approximation in the finite domain. Figure 2.6 shows delta functions used in this paper. We note that new delta functions $\delta_5(c)$ and $\delta_8(c)$ have wider and narrower supports than $\delta_1(c)$ and $\delta_4(c)$, respectively.

![Figure 2.6. Delta functions for phase-field models.](image)

We now consider a line of unit length on a unit domain $\Omega = (0, 1) \times (0, 1)$:

$$c(x, y) = \frac{1}{2} \left( 1 + \tanh \left( \frac{0.5 - x}{2\sqrt{2}\epsilon} \right) \right)$$
for different values of $a$. The value of $a$ is related to the interface thickness. For $a = 0.5\epsilon$, $\epsilon$, and $2\epsilon$, the length of line is given by

$$f_i(a) = \int_{\Omega} \delta_i(c) \, dx \, dy.$$ 

This integral is calculated using the Mathematica code given in the Appendix. The results are given in Table 2.1. Next, we consider a circle of radius $0.25$ centered at the center of a domain $\Omega = (0, 1) \times (0, 1)$:

$$c(x, y) = \frac{1}{2} \left( 1 + \tanh \left( \frac{0.25 - \sqrt{(x - 0.5)^2 + (y - 0.5)^2}}{2\sqrt{2a}} \right) \right).$$

The circumference for different values of $a$ is given in Table 2.1. In both cases, $\delta_i(c)$ is insensitive to the value of $a$. This means that the length of line or circle can be computed accurately using $\delta_i(c)$ regardless of the interface thickness. We will numerically discuss this in more detail in Section 6.1. Figure 2.7 shows the graphs of $f_i(a)$ for $i = 1, 2, \ldots, 8$.

**Table 2.1.** The length of line and circle with different interface thicknesses for each delta function.

<table>
<thead>
<tr>
<th>Case 1: line</th>
<th>Case 2: circle</th>
<th>$f_i(a)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a$</td>
<td>$0.5\epsilon$</td>
<td>$\epsilon$</td>
</tr>
<tr>
<td>$\delta_1$</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>$\delta_2$</td>
<td>0.5</td>
<td>1</td>
</tr>
<tr>
<td>$\delta_3$</td>
<td>1.25</td>
<td>1</td>
</tr>
<tr>
<td>$\delta_4$</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>$\delta_5$</td>
<td>0.5</td>
<td>1</td>
</tr>
<tr>
<td>$\delta_6$</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$\delta_7$</td>
<td>0.75</td>
<td>1</td>
</tr>
<tr>
<td>$\delta_8$</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>
Figure 2.7. The graphs of $f_i(a)$ for $i = 1, 2, \ldots, 8$. 
2.4. Nondimensionalization

To restate the dimensional NSCH system in dimensionless form, we define the dimensionless variables as

\[ x' = \frac{x}{L_c}, \quad u' = \frac{u}{U_c}, \quad t' = \frac{tU_c}{L_c}, \quad p' = \frac{p}{\rho_c U_c^2}, \quad g' = \frac{g}{g}, \quad \rho' = \frac{\rho}{\rho_c}, \quad \mu' = \frac{\mu}{\mu_c}, \]

where \( L_c, U_c, \) and \( \rho_c \) are the characteristic length, velocity, and density, respectively. \( g \) is the gravitational acceleration and \( \mu_c \) is \( \alpha \), which is \( \mu = \alpha(\phi^3 - \frac{\beta}{\alpha}\phi - \frac{k}{\alpha^2}\Delta\phi) = \mu_c\mu' \).

Substituting these variables into the governing equations (2.6) - (2.9) we have

\[
\rho'(u' \cdot \nabla u') = -\nabla p' + \frac{\eta}{\rho_c U_c L_c} \Delta' u' + \frac{g L_c}{U_c^2} \rho' g' - \frac{3\sqrt{2}\epsilon}{4\rho_c L_c U_c^2} \nabla' \cdot \left( \frac{\nabla' \phi}{|\nabla' \phi|} |\nabla' \phi| \nabla' \phi \right), \tag{2.14}
\]

\[
\nabla' \cdot u' = 0, \tag{2.15}
\]

\[
\phi_t + \nabla' \cdot (\phi u') = \frac{M \mu_c}{U_c L_c} \Delta' \mu', \tag{2.16}
\]

\[
\mu = \phi^3 - \phi - \epsilon^2 \Delta \phi. \tag{2.17}
\]

Dropping the primes and considering \( \alpha = \beta \), Equations (2.14) - (2.17) become

\[
\rho(u_t + u \cdot \nabla u) = -\nabla p + \frac{1}{Re} \Delta u + \frac{1}{We} SF + \frac{\rho}{Fr} g, \tag{2.18}
\]

\[
\nabla \cdot u = 0, \tag{2.19}
\]

\[
\phi_t + \nabla \cdot (\phi u) = \frac{1}{Pe} \Delta \mu, \tag{2.20}
\]

\[
\mu = \phi^3 - \phi - \epsilon^2 \Delta \phi, \tag{2.21}
\]

where \( SF = -0.75\sqrt{2}\epsilon \nabla \cdot (\nabla \phi/|\nabla \phi|) |\nabla \phi| \nabla \phi \). The dimensionless physical parameters are the Reynolds number, \( Re \), Weber number, \( We \), Froude number, \( Fr \), Peclet number,
2.4. NONDIMENSIONALIZATION

$Pe$, Cahn number, $\epsilon$, given by

$Re = \frac{\rho_c U_c L_c}{\eta}$, $We = \frac{\rho_c L_c U_c^2}{\sigma}$, $Fr = \frac{U_c^2}{gL_c}$, $Pe = \frac{U_c L_c}{M \mu_c}$, $\epsilon = \sqrt{\frac{k}{\alpha L_c^2}}$. 
2.5. SNOW CRYSTAL GROWTH

In nonlinear dynamical systems, the physics of phase transformations has attracted considerable interest. Crystal growth is an essential part of phase transformations from the liquid phase to the solid phase via heat transfer. To simulate crystal growth, cellular automaton [255, 273, 274, 275, 276], Monte-Carlo [260, 263], boundary integral [256, 257, 264, 266], front-tracking [232, 243, 251], level-set [237, 241, 248, 270], and phase-field [235, 236, 238, 239, 242, 245, 246, 247, 249, 258, 259, 261, 262, 265, 267, 268, 269, 272] methods have been developed. In this thesis, we focus on the phase-field method for crystal growth simulations.

Many numerical methods such as explicit [243, 244, 247, 261, 268], mixed implicit-explicit [259, 269, 272], and adaptive methods [236, 238, 258, 262, 265] have been proposed for crystal growth problems. In particular, a multiple time-step algorithm that uses a larger time step for the flow-field calculations while reserving a finer time step for the phase-field evolution was proposed in [267]. One of the main difficulties in the numerical simulations of crystal growth with various anisotropic interfacial energies is the large difference in time and length scales. The use of space and temporal mesh adaptivity is a natural choice to overcome this problem. Therefore, we need a scheme that allows the use of a sufficiently large time step without the stability limitations. In this paper we present a new accurate and efficient operator splitting algorithm for solving the phase-field model of crystal growth. In particular, we focus on six-fold symmetric crystal growth, which is related to snow crystal. Water has the unique chemical property known as a hydrogen bond. The attractive interaction between the hydrogen and oxygen atoms in different water molecules arranges the solid state water
molecules to form a hexagonal shape. For such a reason, in specific temperature, snow crystal grows into six-fold symmetric crystal [254]. The thickness and width of snow crystal are in the ratio of 1 : 50, so snow crystal problems can be simplified into two-dimensional problems. We consider here the solidification of a pure substance from its supercooled melt in two-dimensional space.
2.6. The Wulff construction

The equilibrium crystal shape is determined by minimizing the total interfacial free energy. We use the following $k$-fold symmetric interfacial energy equation:

$$
\epsilon(\theta) = \epsilon_0 \left[ 1 + \epsilon_k \cos(k\theta) \right],
$$

where $\epsilon_0$ is the mean interfacial tension and $0 \leq \epsilon_k < 1$ is the anisotropy parameter.

![Diagram](image)

Figure 2.8. The Wulff construction. (a) Interfacial free-energy density $\epsilon(\theta)$ in the polar coordinates. (b) Equilibrium crystal shape (bold line) for $k = 6$, $\epsilon_0 = 1$, and $\epsilon_6 = 0.1$.

In this paper, we focus on $k = 6$ case. The equilibrium shape is easily constructed by the Wulff’s theorem [271]. We describe the construction of the equilibrium shape geometrically [234]. Let $M = (\epsilon(\theta), \theta)$ be a point on the interfacial energy function in the polar coordinates (see Figure 2.8(a)). The construction starts from the origin $O$ and draw the line segment $OM$ to the point $M$. Draw the perpendicular line $\overrightarrow{AB}$ to the line segment $OM$. Then the inner convex hull made from all such perpendiculars is an equilibrium crystal shape as shown in Figure 2.8(b).

Conversely, let us assume the equilibrium shape is known and $(r, \theta)$ be the polar coordinates of a point $T$ of the crystal boundary $S$, that is, $T = (r, \theta)$. And let
$T = (x(\phi), y(\phi))$ be the corresponding Cartesian coordinates, where $\phi$ is a parameter and is the angle between $x$-axis and the perpendicular line to the tangent line $\overrightarrow{AB}$ at the point $T$. Let $M$ be the intersection point of the line $\overrightarrow{AB}$ and the perpendicular line containing the origin to $\overrightarrow{AB}$. Let the length of the line segment $\overrightarrow{OM}$ be $p(\phi)$. In Figure 2.9, we can see these parameter definitions. Then $p(\phi)$ can be obtained from the right triangle $\triangle OTM$: \[
p(\phi) = r \cos(\phi - \theta) = r \cos \phi \cos \theta + r \sin \phi \sin \theta = x(\phi) \cos \phi + y(\phi) \sin \phi.
\] (2.22)

We can express $(x(\phi), y(\phi))$ in terms of $p(\phi)$. First, take a derivative to $p(\phi)$, then we have \[
p_\phi(\phi) = x_\phi(\phi) \cos \phi - x(\phi) \sin \phi + y_\phi(\phi) \sin \phi + y(\phi) \cos \phi.
\] (2.23)
Since the normal vector \((\cos \phi, \sin \phi)\) and the tangent vector \((x_\phi, y_\phi)\) are orthogonal, that is, \((\cos \phi, \sin \phi) \cdot (x_\phi, y_\phi) = 0\), we can simplify Equation (2.23) as

\[ p_\phi = -x \sin \phi + y \cos \phi. \tag{2.24} \]

Now, by solving Equations (2.22) and (2.24) we have

\[ x(\phi) = p(\phi) \cos \phi - p_\phi(\phi) \sin \phi, \quad y(\phi) = p(\phi) \sin \phi + p_\phi(\phi) \cos \phi. \tag{2.25} \]

Let \(F\) and \(A\) be the total edge free energy and the area of crystal, respectively and be defined as

\[ F = \int \epsilon(\phi) \sqrt{(x_\phi(\phi))^2 + (y_\phi(\phi))^2} d\phi, \tag{2.26} \]
\[ A = \frac{1}{2} \int (x(\phi)y_\phi(\phi) - y(\phi)x_\phi(\phi)) d\phi. \tag{2.27} \]

Using Equation (2.25), we can rewrite Equations (2.26) and (2.27) in the form

\[ F = \int \epsilon(\phi)(p(\phi) + p_\phi(\phi)) d\phi, \]
\[ A = \frac{1}{2} \int p(\phi)(p(\phi) + p_\phi(\phi)) d\phi. \]

We want to minimize \(F\) with subject to a constant area constraint of \(A\). Using the Lagrange multiplier \(\lambda\), we seek to minimize

\[ F + \lambda A = \int \left( \epsilon(\phi) + \frac{\lambda}{2} p(\phi) \right) (p(\phi) + p_\phi(\phi)) d\phi. \]

And then, the Euler–Lagrange equation is

\[ \frac{\partial Q}{\partial p} - \frac{d}{d\phi} \left( \frac{\partial Q}{\partial p_\phi} \right) + \frac{d^2}{d\phi^2} \left( \frac{\partial Q}{\partial p_\phi} \right) = 0, \tag{2.28} \]

where

\[ Q = \left( \epsilon + \frac{\lambda}{2} p \right) (p + p_\phi). \tag{2.29} \]
From these two Equations (2.28) and (2.29), we get
\[ p + p_{\phi\phi} = -\frac{1}{\lambda} (\epsilon + \epsilon_{\phi\phi}). \] (2.30)

A solution of differential equation (2.30) is
\[ p(\phi) = -\frac{1}{\lambda} \epsilon(\phi). \]

This result implies that in a crystal at equilibrium, the distances of the faces from
the center of the crystal are proportional to their surface free energies per unit area
[234].

For large \( \epsilon_6 \) values, the crystal shape will be energy minimizing when certain orienta-
tions are missing. Missing orientations occur when the polar plot of \( r = 1/\epsilon(\theta) \) changes
convexity [240]. The curvature of a polar plot \( r(\theta) \) is
\[ \kappa = \frac{(r^2 + 2r_\theta^2 - rr_{\theta\theta})/(r^2 + r_\theta^2)^{3/2}}{r^2 + (r_\theta^2)^{3/2}}. \]
For \( r(\theta) = 1/\epsilon(\theta) \), the curvature is
\[ \kappa = (\epsilon + \epsilon_{\theta\theta})/[1 + (\frac{\epsilon_6}{\epsilon})^2]^{3/2}. \] So convexity changes whenever
\[ \epsilon + \epsilon_{\theta\theta} = \epsilon_0 (1 - 35\epsilon_6 \cos 6\theta) < 0. \]

If values of \( \epsilon_6 \) are larger than \( 1/35 \), then missing orientations occur. Figure 2.10 shows
the 6-fold Wulff equilibrium shapes \((x(\phi), y(\phi))\) for \( 0 \leq \phi \leq 2\pi \) with two different
\( \epsilon_6 \) values: (a) \( \epsilon_6 = 1/50 \) and (b) \( \epsilon_6 = 1/10 \) (which shows the missing orientation).

Figure 2.11 shows the trace of \((x(\phi), y(\phi))\) with different intervals. \( \phi_m \) is defined as the
smallest non-zero value which satisfies \( y(\phi_m) = 0 \).
2.6. THE WULFF CONSTRUCTION

(a) $\epsilon_6 = 1/50$

(b) $\epsilon_6 = 1/10$

Figure 2.10. The 6-fold Wulff equilibrium shapes with two different $\epsilon_6$ values.

(a) $0 \leq \phi \leq \phi_m$

(b) $\phi_m \leq \phi \leq \pi/3 - \phi_m$

(c) $\pi/3 - \phi_m \leq \phi \leq \pi/3$

Figure 2.11. Trace of $(x(\phi), y(\phi))$ with different intervals and $y(\phi_m) = 0$. 
2.7. Governing equations of snow crystal growth

The phase-field model for the crystal growth is given by

\[ \epsilon^2(c) \frac{\partial c}{\partial t} = \nabla \cdot (\epsilon^2(c) \nabla c) + [c - \lambda U(1 - c^2)](1 - c^2) \]

\[ + \left( |\nabla c|^2 \epsilon(c) \frac{\partial c}{\partial c_x} \right)_x + \left( |\nabla c|^2 \epsilon(c) \frac{\partial c}{\partial c_y} \right)_y \]  

\[ \frac{\partial U}{\partial t} = D \Delta U + \frac{1}{2} \frac{\partial c}{\partial t}, \]  

where \( c \) is the order parameter, \( \epsilon(c) \) is the anisotropic function, \( \lambda \) is the dimensionless coupling parameter, and \( U = c_p(T - T_M)/L \) is the dimensionless temperature field. Here \( c_p \) is the specific heat at constant pressure, \( T_M \) is the melting temperature, \( L \) is the latent heat of fusion, \( D = \alpha \tau_0/\epsilon_0^2 \), \( \alpha \) is the thermal diffusivity, \( \tau_0 \) is the characteristic time, and \( \epsilon_0 \) is the characteristic length. The order parameter is defined by \( c = 1 \) in the solid phase and \( c = -1 \) in the liquid phase. The interface is defined by \( c = 0 \) and \( \lambda \) is given as \( \lambda = D/a^2 \) with \( a = 0.6267 \) [246, 247]. We define a normal vector of \( c \) as \((c_x, c_y)\) and an angle between normal vector and \( x \)-axis as \( \phi \) that satisfies \( \tan \phi = c_y/c_x \).

Then by replacing \( \epsilon(c) \) with \( \epsilon(\phi) = \epsilon_0(1 + \epsilon_6 \cos(6\phi)) \), we can simplify the following terms in Equation (2.31):

\[ \left( |\nabla c|^2 \epsilon(\phi) \frac{\partial \epsilon(\phi)}{\partial c_x} \right)_x = \left( \epsilon_0^2 + c_y^2 \epsilon(\phi) \epsilon'(\phi) \left( -\frac{c_y}{c_x^2 + c_y^2} \right) \right)_x = -\left( \epsilon'(\phi) \epsilon(\phi) c_y \right)_x. \]

In a similar way, we get

\[ \left( |\nabla c|^2 \epsilon(\phi) \frac{\partial \epsilon(\phi)}{\partial c_y} \right)_y = (\epsilon'(\phi) \epsilon(\phi) c_x)_y. \]
Hence we can rewrite the governing equations of 6-fold symmetric crystal growth as following:

\[
\begin{align*}
\epsilon^2(\phi) \frac{\partial c}{\partial t} &= \nabla \cdot (\epsilon^2(\phi)\nabla c) + [c - \lambda U(1 - c^2)](1 - c^2) \\
&\quad - (\epsilon'(\phi)c_y) + (\epsilon'(\phi)c_x) y \\
\frac{\partial U}{\partial t} &= D\Delta U + \frac{1}{2} \frac{\partial c}{\partial t}.
\end{align*}
\]
2.8. Derivation of the thin film equation

We consider the dynamics of a thin layer of liquid of thickness \( h = h(x, y, t) \) on an inclined surface driven by thermally created surface tension gradients and influenced by gravity. The spatial variables \( x \) and \( y \) denote the direction of the flow and the direction normal to the flow, respectively. Let \( \alpha, \rho, g, \eta, \gamma, \) and \( \tau = d\gamma/dx \) denote the angle from horizontal of inclination of the plane, the density, the gravitational constant, the dynamic viscosity, the surface tension, and the surface tension gradient of the liquid [278].

We briefly review the derivation of the lubrication model [279]. We start from the three-dimensional Navier-Stokes equations.

\[
\begin{align*}
\rho(u_t + uu_x + vu_y + wu_z) &= -p_x + \eta(u_{xx} + u_{yy} + u_{zz}) - \rho g \sin \alpha, \\
\rho(v_t + uv_x + vv_y + wv_z) &= -p_y + \eta(v_{xx} + v_{yy} + v_{zz}), \\
\rho(w_t + uw_x + vw_y + ww_z) &= -p_z + \eta(w_{xx} + w_{yy} + w_{zz}) - \rho g \cos \alpha, \\
u_x + v_y + w_z &= 0,
\end{align*}
\]

where subscripts denote partial derivatives. In this system the dependent variables are \( u(x, y, z, t), v(x, y, z, t), w(x, y, z, t), \) and \( p(x, y, z, t) \).

We define characteristic variables: \( L, H, U, V, W \) and \( T \) are characteristic length (\( x \)- and \( y \)-direction), height (\( z \)-direction), velocity in \( x \)-direction, velocity in \( y \)-direction, velocity in \( z \)-direction, and time, respectively. The lubrication approximation takes advantage of the thickness of the film by employing a small parameter:

\[
\frac{H}{L} = \xi.
\]
The thin film equation is considered slow flow, so the Navier-Stokes equations reduce to the Stokes equations:

\[ \begin{align*}
0 &= -p_x + \eta(u_{xx} + u_{yy} + u_{zz}) - \rho g \sin \alpha, \\
0 &= -p_y + \eta(v_{xx} + v_{yy} + v_{zz}), \\
0 &= -p_z + \eta(w_{xx} + w_{yy} + w_{zz}) - \rho g \cos \alpha, \\
u_x + v_y + w_z &= 0.
\end{align*} \]

Before applying boundary conditions, we drop the terms containing \(u_{xx}, u_{yy}, v_{xx}, v_{yy}, w_{xx}, w_{yy}\), and \(w_{zz}\) by comparing their size (in terms of powers of \(\xi\)) to other terms in the Stokes equations. In particular, if

\[ x = L\hat{x}, \quad y = L\hat{y}, \quad z = H\hat{z}, \quad u = U\hat{u}, \quad \text{and} \quad t = T\hat{t}, \tag{2.34} \]

the incompressibility condition, \(u_x + v_y + w_z = 0\), gives us the following scaling for \(w\):

\[ w = \frac{HU}{L} \hat{w} = \xi U \hat{w} \]

and we see that

\[ \begin{align*}
u_{xx} &= \frac{U}{L^2} \hat{u}_{xx} \ll \frac{U}{H^2} \hat{u}_{zz} = u_{zz}, \quad u_{yy} = \frac{U}{L^2} \hat{u}_{yy} \ll \frac{U}{H^2} \hat{u}_{zz} = u_{zz}, \\
v_{xx} &= \frac{V}{L^2} \hat{v}_{xx} \ll \frac{V}{H^2} \hat{v}_{zz} = v_{zz}, \quad v_{yy} = \frac{V}{L^2} \hat{v}_{yy} \ll \frac{V}{H^2} \hat{v}_{zz} = v_{zz}, \\
w_{xx} &= \frac{\xi U}{L^2} \hat{w}_{xx} \ll \frac{\xi U}{H^2} \hat{w}_{zz} = u_{zz}, \\
w_{yy} &= \frac{\xi U}{L^2} \hat{w}_{yy} \ll \frac{\xi U}{H^2} \hat{w}_{zz} = u_{zz}, \quad \hat{w}_{zz} = \frac{\xi U}{H^2} \hat{w}_{zz} \ll \frac{U}{H^2} \hat{u}_{zz} = u_{zz}.
\end{align*} \]
Thus for a thin film we may reduce the Navier-Stokes equations to the following equations:

\[ p_x = \eta u_{zz} - \rho g \sin \alpha, \]  
\[ p_y = \eta v_{zz}, \]  
\[ p_z = -\rho g \cos \alpha, \]  
\[ u_x + v_y + w_z = 0. \]  

The normal stress boundary condition equates the force exerted by the interface at the liquid-air interface \( z = h(x, y, t) \) on the fluid below with the effect of surface tension, through the curvature of the free surface, \( \kappa \) [280]:

\[ p - p_{atm} = -\gamma \kappa = -\gamma \left( \frac{(1 + h_x^2)h_{xx} + (1 + h_y^2)h_{yy} - 2h_xh_yh_{xy}}{(1 + h_x^2 + h_y^2)^{3/2}} \right). \]

Here, surface tension coefficient is denoted by \( \gamma \). The expression for curvature can be simplified in the lubrication approximation since

\[ (h_x)^2 \approx \left( \frac{H}{L} \right)^2 = \xi^2 \ll 1 \quad \text{and} \quad (h_y)^2 \approx \left( \frac{H}{L} \right)^2 = \xi^2 \ll 1. \]

Therefore, the curvature is approximated by \( \Delta h \) and the normal stress boundary condition is

\[ p - p_{atm} = -\gamma \Delta h \quad \text{at} \quad z = h(x, y, t). \]  

We choose the following linear equation of state to relate the surface tension gradient to the temperature gradient:

\[ \Gamma(T_s) = \gamma_0 - \tau(T_s - T_\infty), \]
where $T_s$ is the temperature at the surface, $T_\infty$ is the ambient temperature, $\tau$ is the surface tension gradient, and $\gamma_0$ is a base value of surface tension. The tangential stress condition at the free surface $z = h(x, y, t)$ is based on the assumption that we drive the film with a constant surface stress, which couples the gradient in surface tension $\tau$ to the scalar component of stress $T_{zx}$ in the flow [281]:

$$\tau = T_{zx} = 2\eta D_{zx},$$

where subscript of $T$ and $D$ denotes partial derivative and component, respectively.

The strain rate tensor has the form

$$D = \frac{1}{2}(\nabla u + \nabla u^T) = \frac{1}{2} \begin{pmatrix} 2u_x & u_y + v_x & u_z + w_x \\ u_y + v_x & 2v_y & v_z + w_y \\ u_z + w_x & v_z + w_y & 2w_z \end{pmatrix}.$$ 

Therefore $D_{zx} = \frac{1}{2}u_z + O(\xi)$ and the tangential stress condition takes the form

$$\tau = \eta u_z \quad \text{at} \quad z = h(x, y, t). \quad (2.39)$$

The no slip boundary condition assumes that the liquid at the liquid-solid interface is not moving. This is expressed in the boundary condition:

$$u = v = 0 \quad \text{at} \quad z = 0.$$

We integrate Equation (2.37) with respect to $z$:

$$p(x, y, z, t) = -\rho g \cos \alpha z + c_1. \quad (2.40)$$

We determine $c_1$ using the normal stress boundary condition (2.38) which yields

$$c_1 = p_{atm} - \gamma \Delta h + \rho g \cos \alpha h.$$
We differentiate Equation (2.40) with respect to $x$ and $y$ and substitute the resulting expression for $p_x$ and $p_y$ into Equations (2.35) and (2.36), respectively:

\[
\eta u_{zz} = \rho g \cos \alpha h_x - \gamma \Delta h_x + \rho g \sin \alpha,
\]
\[
\eta v_{zz} = \rho g \cos \alpha h_y - \gamma \Delta h_y.
\]

We integrate twice with respect to $z$ to derive expressions for the velocity $u$ and $v$. After the first integration we have the following equation:

\[
\eta u_z = (\rho g \cos \alpha h_x - \gamma \Delta h_x + \rho g \sin \alpha)z + c_2,
\]
\[
\eta v_z = (\rho g \cos \alpha h_y - \gamma \Delta h_y)z + c_3.
\]

We determine $c_2$ and $c_3$ using the tangential stress boundary condition equation (2.39) which yields

\[
c_2 = \tau - (\rho g \cos \alpha h_x - \gamma \Delta h_x + \rho g \sin \alpha)h,
\]
\[
c_3 = -(\rho g \cos \alpha h_y - \gamma \Delta h_y)h.
\]

The second integration with respect to $z$ yields

\[
\eta u = (\rho g \cos \alpha h_x - \gamma \Delta h_x + \rho g \sin \alpha)\frac{z^2}{2} + c_2z + c_4,
\]
\[
\eta v = (\rho g \cos \alpha h_y - \gamma \Delta h_y)\frac{z^2}{2} + c_3z + c_5,
\]

where $c_4 = c_5 = 0$ due to the no slip boundary condition. This provides the final velocities $u(x, y, z, t)$ and $v(x, y, z, t)$:

\[
u = \frac{1}{\eta} \left[ (\rho g \cos \alpha h_y - \gamma \Delta h_y) \left( \frac{z^2}{2} - hz \right) \right],
\]
\[
v = \frac{1}{\eta} \left[ (\rho g \cos \alpha h_y - \gamma \Delta h_y) \left( \frac{z^2}{2} - hz \right) \right].
\]
We model the dynamics of the draining film using the lubrication approximation with a “depth averaged” velocity \( \bar{u} = (\bar{u}, \bar{v}) \) [277]:

\[
\bar{u} = \frac{1}{h} \int_0^h udz = \frac{\tau h}{2\eta} - \frac{\rho gh^2 \sin \alpha}{3\eta} + \frac{\gamma h^2 \Delta h}{3\eta} - \frac{\rho gh^2 h_x \cos \alpha}{3\eta},
\]

\[
\bar{v} = \frac{1}{h} \int_0^h vdz = \frac{\gamma h^2 \Delta h_y}{3\eta} - \frac{\rho gh^2 h_y \cos \alpha}{3\eta},
\]

\[
\bar{u} = \left( \frac{\tau h}{2\eta} - \frac{\rho gh^2 \sin \alpha}{3\eta} \right) \bar{e}_x + \frac{\gamma h^2 \nabla h}{3\eta} - \frac{\rho gh^2 h_x \cos \alpha}{3\eta},
\]

where \( \bar{e}_x = (1, 0) \). In Equation (2.41), the first term is due to surface tension gradient, the second term is due to the tangential component of gravity, the third term is due to curvature, and the fourth term is due to the normal component of gravity. Coupling Equation (2.41) with mass conservation, we obtain

\[
h_t + \nabla \cdot (h \bar{u}) = 0.
\]

To non-dimensionalize Equation (2.42), we employ the non-dimensional variables (denoted by hats) defined in Equation (2.34) and \( h = \hat{H} \hat{h} \) to obtain

\[
\frac{\hat{H}}{T} \hat{h}_t + \frac{1}{L} \hat{h} \cdot \left[ \left( \frac{\hat{H}^2 \tau h^2}{2\eta} - \frac{\hat{H}^3 \rho g h^3 \sin \alpha}{3\eta} \right) \bar{e}_x + \frac{\hat{h}^2 \nabla \hat{h}}{3L^3 \eta} - \frac{\rho g h^2 \nabla \hat{h} \cos \alpha}{3L \eta} \right] = 0.
\]

Now we define the characteristic variables by balancing terms. First we balance the tangential gravity and Marangoni terms to define \( \hat{H} \). Note that the independent and dependent variables are all order 1. We obtain the following equation for \( \hat{H} \)

\[
\frac{\hat{H}^2 \tau}{2\eta} = \frac{\hat{H}^3 \rho g \sin \alpha}{3\eta}, \text{ i.e., } \hat{H} = \frac{3\tau}{2\rho g \sin \alpha}.
\]

Next we define \( L \) such that the Marangoni and surface tension effects balance:

\[
\frac{\hat{H}^2 \tau}{2\eta} = \frac{\hat{H}^4 \gamma}{3L^3 \eta}, \text{ i.e., } L = \left( \frac{3\gamma \tau}{2\rho g^2 \sin^2 \alpha} \right)^{1/3}.
\]
Now choose the time scale $T$ so that gravity, Marangoni and surface tension forces balance:

$$\frac{H}{T} = \frac{H^2 \tau}{2L \eta}, \text{ i.e., } T = \frac{2\eta}{\tau^2} \left( \frac{4\gamma \rho g \sin \alpha}{9} \right)^{1/3}.$$

We substitute the expressions for $H$, $L$, and $T$ into Equation (2.43) and drop the ‘‘hat’’ to obtain the dimensionless equation thin film equation:

$$h_t + (h^2 - h^3)_x = D \nabla \cdot (h^3 \nabla h) - \nabla \cdot (h^3 \nabla \Delta h),$$

where $D = \frac{TH^3 \rho g \cos \alpha}{3\eta L^2}$. As in [278] we take $D = 0$. Thus we have derived the dimensionless thin film equation

$$h_t + (h^2 - h^3)_x = -\nabla \cdot (h^3 \nabla \Delta h). \quad (2.44)$$

We can think of this traveling wave as a viscous regularization of a shock wave, if we rescale the space and the time variables by $(x', y') = \epsilon (x, y)$ and $t' = \epsilon t$, then Equation (2.44) becomes after dropping the ‘’ notation

$$h_t + (h^2 - h^3)_x = -\epsilon^3 \nabla \cdot (h^3 \nabla \Delta h). \quad (2.45)$$

This equation is a fourth order nonlinear singular perturbation of the conservative law

$$h_t + (h^2 - h^3)_x = 0 \quad [278].$$

Here, $\epsilon$ parameter is related to the surface tension coefficient $\gamma \quad [282]$. 
Chapter 3

The multi-component Cahn–Hilliard system

The CH equation is the prototypical continuum model of phase separation in a binary alloy. It was originally derived by Cahn and Hilliard [117] to describe spinodal decomposition and has been widely adopted to model many other physical phenomena such as contact angle and wetting problem [148, 64, 149], gravity and capillary waves [150, 151], mixing [63], pinchoff of liquid-liquid jets [152, 153], Rayleigh-Taylor instability [11, 154, 155, 156], solid tumor growth [157, 158, 159], thermocapillary flow [160, 161], and vesicle dynamics [162].

Most of the technologically important alloys are multi-component systems exhibiting multiple phases in their microstructures. Moreover, one or more of these phases are formed as a result of phase transformations induced during processing. Since performance of these multi-component alloys depends crucially on the morphology of the phases, a fundamental understanding of the kinetics of phase transformations is important for controlling the microstructures of these multi-phase alloys [163].

We consider the evolution of the $N$-component CH system on a domain $\Omega \subset \mathbb{R}^d$, $d = 1, 2, 3$. Let $\mathbf{c} = (c_1, \ldots, c_N)$ be the phase variables (i.e., the mole fractions of different components). Clearly the total mole fractions must sum to 1, i.e.,

$$c_1 + \cdots + c_N = 1,$$
so that, admissible states will belong to the Gibbs $N$-simplex

$$G := \left\{ c \in \mathbb{R}^N \mid \sum_{i=1}^{N} c_i = 1, \ 0 \leq c_i \leq 1 \right\}. \quad (3.1)$$

Without loss of generality, we postulate that the free energy can be written as follows:

$$F(c) = \int_{\Omega} \left( F(c) + \frac{\epsilon^2}{2} \sum_{i=1}^{N} |\nabla c_i|^2 \right) dx, \quad (3.2)$$

where $F(c) = 0.25 \sum_{i=1}^{N} c_i^2 (1 - c_i)^2$ and $\epsilon > 0$ is the gradient energy coefficient. The natural boundary condition for the $N$-component CH system is the zero Neumann boundary condition:

$$\nabla c_i \cdot n = 0 \text{ on } \partial \Omega, \quad (3.3)$$

where $n$ is the unit normal vector to $\partial \Omega$.

The time evolution of $c$ is governed by the gradient of the energy with respect to the $\dot{H}^{-1}$ inner product under the additional constraint (3.1). This constraint has to hold everywhere at any time. In order to ensure this last constraint, we use a variable Lagrangian multiplier $\beta(c) [205]$. The time dependence of $c_i$ is given by the following CH equation:

$$\frac{\partial c_i}{\partial t} = M \Delta \mu_i, \quad (3.4)$$

$$\mu_i = f(c_i) - \epsilon^2 \Delta c_i + \beta(c), \text{ for } i = 1, \ldots, N; \quad (3.5)$$

where $M$ is a mobility, $f(c) = c(c - 0.5)(c - 1)$, and $\beta(c) = -\frac{1}{N} \sum_{i=1}^{N} f(c_i)$. We take $M \equiv 1$ for convenience.

The mass conserving boundary condition for the $N$-component CH system is

$$\nabla \mu_i \cdot n = 0 \text{ on } \partial \Omega. \quad (3.6)$$
We differentiate the energy $F$ and the total mass of each phase, $\int_\Omega c_i \, dx$, to get
\[
\frac{d}{dt} F(t) = -M \int_\Omega \sum_{i=1}^N |\nabla \mu_i|^2 \, dx \leq 0 \quad (3.7)
\]
and
\[
\frac{d}{dt} \int_\Omega c_i \, dx = 0, \quad (3.8)
\]
where we used the mass conserving boundary condition (3.6). For a more detailed description of derivations of (3.7) and (3.8), please refer to Ref. [195]. Therefore, the total energy is non-increasing in time and the total mass of each phase is conserved. That is
\[
F(t) \leq F(0) \quad \text{and} \quad \int_\Omega c_i(x, t) \, dx = \int_\Omega c_i(x, 0) \, dx \quad \text{for} \quad i = 1, \ldots, N.
\]

The generalization of the CH equation to multi-component systems first appeared in the literature of De Fontaine [164] and Morral and Cahn [165]. Hoyt [166] extended the CH continuum theory of nucleation to multi-component solutions. Elliott and Luckhaus [167] gave a global existence result under constant mobility and specific assumptions. Eyre [168] studied differences between multi-component and binary alloys and discussed the equilibrium and dynamic behavior of multi-component systems. Elliot and Garcke [169] proved a global existence for multi-component systems when the mobility matrix depends on a concentration. Maier-Paape et al. [170] explained the initial-stage phase separation process in multi-component CH systems through spinodal decomposition.

There are many numerical studies with the binary CH equation [149, 153, 156, 171, 172, 65, 173, 174, 175, 72, 176, 177, 178, 179, 180, 181], ternary [163, 171, 182, 183, 184, 185, 186, 187, 188, 189, 190, 191, 192, 193], and quaternary [194, 195, 196, 58, 197] CH systems. One of the main difficulties in solving the CH system numerically is that the
system is fourth-order in space which makes difference stencils very large and introduces a severe time step restriction for stability, i.e., $\Delta t \sim (\Delta x)^4$ for explicit methods. And there is a nonlinearity at the lower order spatial derivatives which can also contribute to numerical stability. Copetti [184] considered an explicit finite element approximation of a model for phase separation in a ternary mixture and took $\Delta x = 1.0$ and $\Delta t = 0.01$ in numerical experiments. In [163], Bhattacharyya and Abinandanan used a semi-implicit Fourier spectral method for solving the ternary CH equations. They treated the linear fourth-order terms implicitly and the non-linear terms explicitly and simulations were carried out using $\Delta x = 1.0$ and $\Delta t = 0.05$. Ma [185] performed numerical simulation of the phase separation kinetics in ternary mixtures with different interfacial properties by means of the cell dynamics approach and in the literature $\Delta x$ and $\Delta t$ were both set to be unity. In [182], $\Delta x$ and $\Delta t$ were also both set to be unity. Kim and Lowengrub [187] developed a stable, conservative, and second-order accurate fully implicit discretization of the ternary CH system. The authors used a nonlinear multigrid method to efficiently solve the discrete ternary CH system at the implicit time-level and a uniform time step $\Delta t \leq 0.25\Delta x$ for a uniform mesh size $\Delta x$. Boyer and Lapuerta [189] used the implicit Galerkin finite elements method with $\Delta x = 8.33 \times 10^{-4}$ and $\Delta t = 0.001$ to solve the ternary CH system.

In numerous papers there are stability restrictions on the time step which cause huge computational costs and make the calculation very inefficient. Therefore we need a scheme that allows the use of a sufficiently large time step without the technical limitations. But, though such an algorithm allows the use of a sufficiently large time step, it seems to be less attractive, because we need to invert an $(N - 1) \times (N - 1)$
matrix to obtain the solution of $N$-component CH system (except for the explicit method) and the matrix inversion becomes more and more complicated for increasing $N$. This problem was shown in previous papers. The authors in [195] developed a nonlinear multigrid method to efficiently solve the discrete $N$-component CH system at the implicit time level, but an iteration step for the nonlinear multigrid method consists of a $2(N - 1) \times 2(N - 1)$ matrix inversion.

In the literature [198, 199, 200, 201, 202], to overcome this problem, a variety of numerical approaches have been developed for a large number of components. These approaches can significantly reduce the required computer memory and CPU time, but such approaches are limited to the Allen–Cahn system. Adaptive mesh refinement (AMR) [159, 175, 179, 203, 73] has been used in the CH simulations to increase computational efficiency, but implementation of AMR could be difficult and, in general, AMR is employed only a small number of components.

In this thesis, we present a practically unconditionally gradient stable conservative numerical method for solving the CH system representing a model for phase separation in an $N$-component mixture. This method allows us to solve the $N$-component CH system in a decoupled way and significantly reduces the CPU time and memory requirements. We emphasize that while the method will allow us to take arbitrarily large time steps, the accuracy of the numerical solution depends on choosing a small enough time step to resolve the dynamics [204].
Chapter 4

The multi-component Allen–Cahn system

The Allen–Cahn (AC) equation [208] was originally introduced as a phenomenological model for antiphase domain coarsening in a binary alloy. The AC equation and its various modified forms have been applied to a wide range of problems, such as phase transitions [208], image analysis [209, 210], motion by mean curvature flows [211, 212, 213, 214, 215, 216], two-phase fluid flows [217], and crystal growth [218, 219, 220]. Therefore, an efficient and accurate numerical solution of this equation is needed to better understand its dynamics.

We consider the evolution of the $N$-component AC system on a domain $\Omega \subset \mathbb{R}^d$, $d = 1, 2, 3$. Let $\mathbf{c} = (c_1, c_2, \ldots, c_N)$ be a vector valued phase-field and $c_i = c_i(x, t)$ for $i = 1, 2, \ldots, N$ is the mole fraction of the $i$th component in the mixture as a function of space and time. Clearly the total mole fractions must sum to 1, i.e.,

$$c_1 + \cdots + c_N = 1,$$

so that, admissible states will belong to the Gibbs $N$-simplex

$$G := \left\{ \mathbf{c} \in \mathbb{R}^N \bigg| \sum_{i=1}^{N} c_i = 1, \ 0 \leq c_i \leq 1 \right\}.$$

Without loss of generality, we postulate that the free energy can be written as follows:

$$\mathcal{F}(\mathbf{c}) = \int_{\Omega} \left( \frac{F(c)}{\epsilon^2} + \frac{1}{2} \sum_{i=1}^{N} |\nabla c_i|^2 \right) dx,$$
where \( F(c) = 0.25 \sum_{i=1}^{N} c_i^2 (1 - c_i)^2 \) is the Helmholtz free energy density [221]. The small positive constant \( \epsilon \) is the gradient energy coefficient related to the interfacial energy. The natural boundary condition for the \( N \)-component AC system is the zero Neumann boundary condition:

\[
\nabla c_i \cdot n = 0 \quad \text{on} \quad \partial \Omega,
\]

(4.2)

where \( n \) is the unit normal vector to \( \partial \Omega \).

The time evolution of \( c \) is governed by the \( L^2 \)-gradient flow of the total free energy under the additional constraint (4.1), which has to hold everywhere at any time. In order to ensure this last constraint, we use a variable Lagrangian multiplier \( \beta(c) \) [222] and set \( \frac{\partial F}{\partial c} = \left( \frac{\partial F}{\partial c_1}, \frac{\partial F}{\partial c_2}, \ldots, \frac{\partial F}{\partial c_N} \right) = f(c) = (f(c_1), f(c_2), \ldots, f(c_N)) \), where \( f(c) = c(c - 0.5)(c - 1) \). Let \( \mathbf{1} = (1, 1, \ldots, 1) \in \mathbb{R}^N \). Using a general smooth vector valued function \( \xi \), we set

\[
\mathbf{d} = (d_1, d_2, \ldots, d_N) = \xi - \frac{1}{N} \sum_{i=1}^{N} \xi_i \mathbf{1}, \quad \text{then} \quad \sum_{i=1}^{N} d_i = 0.
\]

Let

\[
\beta(c) = -\frac{1}{N} \sum_{i=1}^{N} f(c_i).
\]

Then, the time dependence of \( c \) is given by the following \( N \)-component AC system:

\[
\frac{\partial c(x, t)}{\partial t} = -M \left( \frac{F'(c(x, t)) + \beta(c) \mathbf{1}}{c^2} - \Delta c(x, t) \right), \quad x \in \Omega, \ 0 < t \leq T,
\]

where the coefficient, \( M \), is a constant mobility and \( M \equiv 1 \) is taken for convenience.

Numerical simulations of the \( N \)-component AC system, using explicit methods, impose severe time step restrictions requiring the use of implicit type methods [118, 204, 223]. Restrictions on a time step cause huge computational costs and make the calculation very inefficient. So some authors [224, 225, 226, 227] used adaptive mesh
refinement (AMR) methods to increase computational efficiency in the AC simulations, but implementation of AMR could be difficult and, in general, AMR is employed only a small number of components.

The nonlinearly stabilized splitting scheme that involves a semi-implicit time and centered difference space discretizations of the binary AC equation is

$$\frac{c_i^{n+1} - c_i^n}{\Delta t} = \frac{0.25c_i^n - 0.75c_i^{n+1} + 1.5(c_i^{n+1})^2 - (c_i^{n+1})^3}{\epsilon^2} + \Delta h c_i^{n+1}$$

(4.3) for $i = 1, \ldots, N$ and $n = 0, \ldots, N_t - 1$. In Ref. [228], the authors reported the energy decreasing property for the corresponding discrete problem by using the eigenvalues of the Hessian matrix of the energy functional. And an implicit Euler’s scheme of the one-dimensional AC equation is

$$\frac{c_i^{n+1} - c_i^n}{\Delta t} = -0.5c_i^{n+1} + 1.5(c_i^{n+1})^2 - (c_i^{n+1})^3 + \Delta h c_i^{n+1}.$$ 

This scheme is much better than the simple explicit Euler’s scheme. However, the implicit Euler’s scheme suffers from instability if a large time step is used [118]. Equation (4.3) can be reformulated as

$$\frac{c_i^{n+1} - c_i^n}{\Delta t(2\epsilon)^2} = \frac{-0.5c_i^{n+1} + 1.5(c_i^{n+1})^2 - (c_i^{n+1})^3}{\epsilon^2} + \Delta h c_i^{n+1}.$$ 

This suggests that the unconditionally gradient stable scheme can be considered a time step rescaling of the implicit Euler’s scheme. In addition,

$$\Delta t_{\text{equiv}} = \frac{\Delta t(2\epsilon)^2}{\Delta t + (2\epsilon)^2} \leq \min(\Delta t, (2\epsilon)^2).$$

That is, the scaled equivalent time step $\Delta t_{\text{equiv}}$ is bounded by $(2\epsilon)^2$, which is a small value.

In this thesis, we propose an unconditionally stable and accurate numerical method for solving the $N$-component AC system. The proposed method is based on an operator
splitting technique, which is the method for treating the individual terms of a partial differential equation independently. The main benefit of operator splitting methods is that the stability depends on the minimum of each term rather than the stability of all terms combined. The proposed scheme for the $N$-component AC system involves two steps. First, $c_t = \Delta c - (\beta(c)1)/\epsilon^2$ is solved numerically using an implicit Euler’s method. Second, $c_t = -f(c)/\epsilon^2$ is solved analytically.
Chapter 5

Numerical solutions

5.1. Cahn–Hilliard solver

In this section, we present fully discrete schemes for the CH equation. We consider an unconditionally gradient stable scheme for time discretization introduced by Eyre [118, 204]. For simplicity of exposition, we shall discretize the CH equation in two-dimensional space, i.e., $\Omega = (a, b) \times (c, d)$. One and three-dimensional discretizations are defined analogously. Let $N_x$ and $N_y$ be positive even integers, $h = (b - a)/N_x$ be the uniform mesh size, and $\Omega_h = \{(x_i, y_j) : x_i = (i - 0.5)h, y_j = (j - 0.5)h, 1 \leq i \leq N_x, 1 \leq j \leq N_y\}$ be the set of cell-centers. Let $c_{ij}$ and $\mu_{ij}$ be approximations of $c(x_i, y_j)$ and $\mu(x_i, y_j)$. We define a discrete energy functional by

$$E^h(c^n) = \frac{h^2}{4} \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} \left((c_{ij}^n)^2 - 1\right)^2 + \frac{\epsilon^2 h^2}{2} \left( \sum_{i=0}^{N_x} \sum_{j=1}^{N_y} |\nabla_d c_{i+\frac{1}{2},j}^n|^2 + \sum_{i=1}^{N_x} \sum_{j=0}^{N_y} |\nabla_d c_{i,j+\frac{1}{2}}^n|^2 \right).$$

By using the linearly stabilized splitting scheme [204], we present an implicit time and centered difference space discretization of Equations (2.2) and (2.3) without flow:

$$\frac{c_{ij}^{n+1} - c_{ij}^n}{\Delta t} = \Delta_d \mu_{ij}^{n+1}, \quad (5.1)$$

$$\mu_{ij}^{n+1} = 2c_{ij}^{n+1} - \epsilon^2 \Delta_d c_{ij}^{n+1} + f(c_{ij}^n) - 2c_{ij}^n, \quad (5.2)$$

where $f(c) = F'(c)$. 

Eyre [118, 204] proved that if $c^{n+1}$ is the solution of Equations (5.1) and (5.2) with a given $c^n$, then

$$E^h(c^{n+1}) \leq E^h(c^n).$$

Furihata et al. [284] have examined the boundedness of the solution of a finite difference scheme [285] using discretized Lyapunov functional. Furihata [283] has shown that the decrease of the total energy implies boundedness of discretized Sobolev norm of the solution, independent $\Delta t$ and $\Delta x$, unconditionally.

We show that boundedness of the solution of finite difference scheme using the decrease of the discrete total energy functional. The decrease of the discrete total energy functional means that the numerical solution for the CH equation is pointwise bounded. If $c^n$ is a numerical solution for the discrete Equations (5.1) and (5.2), then there exists a constant $K$, independent of $n$, such that

$$\|c^n\|_\infty \leq K. \quad (5.3)$$

We prove Equation (5.3) by a contradiction. Assume on the contrary that there is an integer $n_K$, dependent on $K$, such that $\|c^{n_K}\|_\infty > K$ for all $K$. Then there is an index $i$ ($1 \leq i \leq N$) such that $|c_i^{n_K}| > K$. Let $K$ be the largest solution of $hF(K) = E^h(c^0)$, i.e., $K = \sqrt{1 + 2\sqrt{E^h(c^0)/h}}$. Note that $K \geq 1$. Then, $F(c)$ is a strictly increasing function on $(K, \infty)$ (see Figure 5.1). Since the total energy is non-increasing, we have $E^h(c^0) = hF(K) < hF(|c_i^{n_K}|) \leq E^h(c^{n_K}) \leq E^h(c^0)$ [286]. This contradiction implies that Equation (5.3) should be satisfied.
Figure 5.1. Graph of $h_F(c)$. 
5.2. Multigrid methods

In this section, we develop a multigrid method to solve the linear discrete system (5.1) and (5.2) at the implicit time level. A pointwise Gauss-Seidel relaxation scheme is used as the smoother in the multigrid method. See the reference text [71] for additional details. We use the same notations as this reference text. Let us rewrite Equations (5.1) and (5.2) as follows.

\[ L(c^{n+1}, \mu^{n+1}) = (\phi^n, \psi^n), \]

where the linear system operator \((L)\) is defined as

\[ L(c^{n+1}, \mu^{n+1}) = \left( \frac{c^{n+1}}{\Delta t} - \Delta_d \mu^{n+1}, \mu^{n+1} - 2c^{n+1} + \epsilon^2 \Delta_d c^{n+1} \right) \]

and the source term is \((\phi^n, \psi^n) = (c^n/\Delta t, f(c^n) - 2c^n)\). In the following description of one multigrid cycle, we assume a sequence of grids \(\Omega_k\) (\(\Omega_{k-1}\) is coarser than \(\Omega_k\) by a factor of 2). Given the numbers, \(\nu_1\) and \(\nu_2\), of pre- and post-smoothing relaxation sweeps, an iteration step for the multigrid method using the V-cycle is formally written as follows:

\[
\{c_k^{n+1,m+1}, \mu_k^{n+1,m+1}\} = MGCycle(k, c_k^{n+1,m}, \mu_k^{n+1,m}, L_k, \phi^n_k, \psi^n_k, \nu_1, \nu_2) \text{ on } \Omega_k \text{ grid.}
\]

That is, \(\{c_k^{n+1,m}, \mu_k^{n+1,m}\}\) and \(\{c_k^{n+1,m+1}, \mu_k^{n+1,m+1}\}\) are the approximations of \(\{c_k^{n+1}(x_i, y_j), \mu_k^{n+1}(x_i, y_j)\}\) before and after an MGcycle.

**Step 1) Presmoothing**

\[
\{c_k^{n+1,m}, \mu_k^{n+1,m}\} = SMOOTH^{\nu_1}(c_k^{n+1,m}, \mu_k^{n+1,m}, L_k, \phi^n_k, \psi^n_k) \text{ on } \Omega_k \text{ grid.}
\]
This means performing $\nu_1$ smoothing steps. Let us discretize Equations (5.1) and (5.2) as a Gauss-Seidel type.

$$\frac{\vec{c}_{k,ij}^{n+1,m} - \vec{c}_{k,ij}^{n+1,m}}{\Delta t} + 4\hat{\mu}_{k,ij}^{n+1,m} h^2 = \phi_{k,ij}^{n+1,m} + \frac{\hat{\mu}_{k,i-1,j}^{n+1,m} + \hat{\mu}_{k,i+1,j}^{n+1,m} + \hat{\mu}_{k,i,j-1}^{n+1,m} + \hat{\mu}_{k,i,j+1}^{n+1,m}}{h^2},$$  

(5.4)

$$- \left(2 + \frac{4\epsilon^2}{h^2}\right)\vec{c}_{k,ij}^{n+1,m} + \vec{\mu}_{k,ij}^{n+1,m} = \psi_{k,ij}^{n+1,m} - \epsilon^2 e_{k,i-1,j}^{n+1,m} + e_{k,i+1,j}^{n+1,m} + e_{k,i,j-1}^{n+1,m} + e_{k,i,j+1}^{n+1,m},$$  

(5.5)

One SMOOTH relaxation operator step consists of solving the system (5.4) and (5.5) by a $2 \times 2$ matrix inversion for each $ij$.

**Step 2) Coarse grid correction**

- Compute the defect: $(\vec{d}_{1k}^m, \vec{d}_{2k}^m) = (\phi_{k}^{n+1,m}, \psi_{k}^{n+1,m}) - L_k(\vec{c}_{k}^{n+1,m}, \vec{\mu}_{k}^{n+1,m})$.

- Restrict the defect: $(\vec{d}_{1,k-1}^m, \vec{d}_{2,k-1}^m) = I_{k-1}(\vec{d}_{1k}^m, \vec{d}_{2k}^m)$.

- Compute an approximate solution $\{\vec{c}_{k}^{m,n+1,m}, \vec{\mu}_{k}^{m,n+1,m}\}$ of the coarse grid equation on $\Omega_{k-1}$:

$$L_{k-1}(\vec{c}_{k-1}^{n+1,m}, \vec{\mu}_{k-1}^{n+1,m}) = (\vec{d}_{1,k-1}^m, \vec{d}_{2,k-1}^m).$$  

(5.6)

If $k = 1$, we apply the smoothing procedure in **Step 1)** to obtain the approximate solution. If $k > 1$, we solve (5.6) by performing a $k$-grid cycle using the zero grid function as an initial approximation:

$$\{\vec{v}_{1,k-1}^{n+1,m}, \vec{v}_{2,k-1}^{n+1,m}\} = MGcycle(k - 1, 0, 0, L_{k-1}, \vec{d}_{1,k-1}^m, \vec{d}_{2,k-1}^m, \nu_1, \nu_2)$$.  

- Interpolate the correction: $(\vec{v}_{1k}^{n+1,m}, \vec{v}_{2k}^{n+1,m}) = I_{k-1}^k(\vec{v}_{1,k-1}^{n+1,m}, \vec{v}_{2,k-1}^{n+1,m})$.

- Compute the corrected approximation on $\Omega_k$:

$$c_{k}^m, after\ CGC = c_{k}^{n+1,m} + \hat{v}_{1k}^{n+1,m}, \mu_{k}^m, after\ CGC = \hat{\mu}_{k}^{n+1,m} + \hat{v}_{2k}^{n+1,m}$$.
5.2. MULTIGRID METHODS

Step 3) Postsmoothing:

\[
\{ c_k^{n+1,m+1}, \mu_k^{n+1,m+1} \} = SMOOTH^{\nu_2}(c_k^m, \mu_k^m, \phi_k^n, \psi_k^n) \text{ on } \Omega_k \text{ grid.}
\]

This completes the description of an MGcycle. Figure 5.2 shows the description of an MGcycle.

Describing an algorithm for a discrete system is one thing and actual implementation is another. Especially, using recursive function for a multigrid method is non-trivial task for the beginners. We made the code as simple as possible. Therefore, it needs
a modification for accounting other features of the equations such as different boundary conditions, variable mobilities, three dimensional extension, and more than two components. These modifications can be done in a straightforward manner.
5.3. Navier–Stokes solver

The simultaneous solution of the large number of discrete equations arising from (2.6) and (2.7) is very costly, especially in three spatial dimensions [69]. An efficient approximation can be obtained by decoupling the solution of the momentum equations from the solution of the continuity equation by a projection method [67, 68]. We will focus on describing the idea in two-dimensions. A staggered marker-and-cell (MAC) mesh of Harlow and Welch [70] is used in which pressure and phase fields are stored at cell centers and velocities at cell interfaces (see Figure 5.3).

\[
\begin{array}{cc}
  u_{i-1/2,j+1} & u_{i+1/2,j+1} \\
  \downarrow & \downarrow \\
  v_{i,j+1/2} & v_{i+1,j+1/2} \\
  \phi_{i,j} P_{i,j} & u_{i+1/2,j} \\
  v_{i,j-1/2} & v_{i+1,j-1/2}
\end{array}
\]

**Figure 5.3.** Velocities are defined at cell boundaries while the pressure and phase field are defined at the cell centers.

Let a computational domain be partitioned in Cartesian geometry into a uniform mesh with mesh spacing $h$. The center of each cell, $\Omega_{ij}$, is located at $(x_i, y_j) = ((i - 0.5)h, (j - 0.5)h)$ for $i = 1, \cdots, N_x$ and $j = 1, \cdots, N_y$. $N_x$ and $N_y$ are the numbers of cells in $x$ and $y$-directions, respectively. The cell vertices are located at $(x_{i+\frac{1}{2}}, y_{j+\frac{1}{2}}) = (ih, jh)$. 
At the beginning of each time step, given \( u^n, \phi^n, \) and \( p^n \), we want to find \( u^{n+1}, \phi^{n+1}, \) and \( p^{n+1} \) which solve the following temporal discretization of dimensionless form of the Equations (2.18) - (2.21) of motion:

\[
\rho^n \frac{u^{n+1} - u^n}{\Delta t} = -\rho^n (u \cdot \nabla_d u) + \nabla_d p^{n+1} + \frac{1}{Re} \Delta_d u^n + \frac{1}{We} SF^n + \frac{\rho^n}{Fr} g,
\]

\[
\nabla_d \cdot u^{n+1} = 0,
\]

\[
\phi^{n+1} - \phi^n = \frac{1}{Pe} \Delta_d \phi^n - \nabla_d \cdot (\phi u^n),
\]

\[
\nu^{n+1} = (\phi^{n+1})^3 - \epsilon^2 \Delta_d \phi^{n+1},
\]

where \( \rho^n = \rho(\phi^n) \) and \( g = (0, -1) \).

The outline of the main procedures in one time step is:

**Step 1.** Initialize \( u^0 \) to be the divergence-free velocity field and \( \phi^0 \).

**Step 2.** Solve an intermediate velocity field, \( \tilde{u} \), which generally does not satisfy the incompressible condition, without the pressure gradient term,

\[
\frac{\tilde{u} - u^n}{\Delta t} = -u^n \cdot \nabla_d u^n + \frac{1}{\rho^n Re} \Delta_d u^n + \frac{1}{\rho^n We} SF^n + \frac{1}{Fr} g.
\]

The resulting finite difference equations are written out explicitly. They take the form

\[
\tilde{u}_{i+\frac{1}{2},j} = u^n_{i+\frac{1}{2},j} - \Delta t (u x + v y)_i^n + \frac{\Delta t}{\rho^n_{i+\frac{1}{2},j} We} SF^{x-edge}_{i+\frac{1}{2},j} + \frac{\Delta t}{h^2 \rho^n_{i+\frac{1}{2},j} Re} (u^n_{i+\frac{1}{2},j} - v^n_{i+\frac{1}{2},j} - 4u^n_{i+\frac{1}{2},j} + u^n_{i+\frac{1}{2},j+1} + u^n_{i+\frac{1}{2},j-1}),
\]

\[
\tilde{v}_{i,j+\frac{1}{2}} = v^n_{i,j+\frac{1}{2}} - \Delta t (u x + v y)_{i,j+\frac{1}{2}} + \frac{\Delta t}{\rho^n_{i,j+\frac{1}{2}} We} SF^{y-edge}_{i,j+\frac{1}{2}} - \frac{\Delta t}{Fr},
\]

\[
\tilde{v}_{i,j+\frac{1}{2}} = v^n_{i,j+\frac{1}{2}} - \Delta t (u x + v y)_{i,j+\frac{1}{2}} + \frac{\Delta t}{\rho^n_{i,j+\frac{1}{2}} We} SF^{y-edge}_{i,j+\frac{1}{2}} - \frac{\Delta t}{Fr}.
\]
where the advection terms, \((uu_x + vu_y)^n_{i+\frac{1}{2},j}\) and \((uv_x + vv_y)^n_{i,j+\frac{1}{2}}\), are defined by

\[
(uu_x + vu_y)^n_{i+\frac{1}{2},j} = u^n_{i+\frac{1}{2},j} \tilde{u}^n_{x,i+\frac{1}{2},j} + \frac{u^n_{i,j-\frac{1}{2}} + u^n_{i+1,j-\frac{1}{2}} + u^n_{i,j+\frac{1}{2}} + u^n_{i+1,j+\frac{1}{2}}}{4} \tilde{u}^n_{y,i+\frac{1}{2},j},
\]

\[
(uv_x + vv_y)^n_{i,j+\frac{1}{2}} = \frac{u^n_{i-\frac{1}{2},j} + u^n_{i-\frac{1}{2},j+1} + u^n_{i+\frac{1}{2},j} + u^n_{i+\frac{1}{2},j+1}}{4} \tilde{v}^n_{x,i,j+\frac{1}{2}} + v^n_{i,j+\frac{1}{2}} \tilde{v}^n_{y,i+\frac{1}{2},j}.
\]

The values \(\tilde{u}^n_{x,i+\frac{1}{2},j}\) and \(\tilde{u}^n_{y,i+\frac{1}{2},j}\) are computed using the upwind procedure. The procedure is

\[
\tilde{u}^n_{x,i+\frac{1}{2},j} = \begin{cases} 
\frac{u^n_{i+\frac{1}{2},j} - u^n_{i-\frac{1}{2},j}}{h} & \text{if } u^n_{i+\frac{1}{2},j} > 0 \\
\frac{u^n_{i+\frac{1}{2},j} - u^n_{i-\frac{1}{2},j}}{h} & \text{otherwise}
\end{cases}
\]

and

\[
\tilde{v}^n_{y,i+\frac{1}{2},j} = \begin{cases} 
\frac{v^n_{i,j+\frac{1}{2}} - v^n_{i,j-\frac{1}{2}}}{h} & \text{if } v^n_{i,j+\frac{1}{2}} > 0 \\
\frac{v^n_{i,j+\frac{1}{2}} - v^n_{i,j-\frac{1}{2}}}{h} & \text{otherwise}.
\end{cases}
\]

The quantities \(\tilde{v}^n_{x,i,j+\frac{1}{2}}\) and \(\tilde{v}^n_{y,i,j+\frac{1}{2}}\) are computed in a similar manner. The surface tension force terms, \(SF_{x-edge}^{i+\frac{1}{2},j}\) and \(SF_{y-edge}^{i,j+\frac{1}{2}}\), are computed using the procedure derived in Ref. [190].

Then, we solve the following equations for the advanced pressure field at \((n + 1)\) time step.

\[
\frac{u^{n+1} - \tilde{u}}{\Delta t} = -\frac{1}{\rho^n} \nabla d p^{n+1}, \quad (5.9)
\]

\[
\nabla d \cdot u^{n+1} = 0. \quad (5.10)
\]

With application of the divergence operator to Equation (5.9), we find that the Poisson equation for the pressure at the advanced time \((n + 1)\).

\[
\nabla d \cdot \left( \frac{1}{\rho^n} \nabla d p^{n+1} \right) = \frac{1}{\Delta t} \nabla d \cdot \tilde{u}, \quad (5.11)
\]
where we have made use of the Equation (5.10) and the terms are defined as in the following.

\[
\nabla_d \cdot \left( \frac{1}{\rho^n} \nabla \rho_i^{n+1} \right) = \frac{1}{h^2} \left( \rho_{i+\frac{1}{2},j} p_{i+1,j}^{n+1} + \rho_{i-\frac{1}{2},j} p_{i-1,j}^{n+1} + \rho_{i,j+\frac{1}{2}} p_{i,j+1}^{n+1} + \rho_{i,j-\frac{1}{2}} p_{i,j-1}^{n+1} \right) \\
\frac{1}{h^2} \left( \rho_{i+\frac{1}{2},j} + \rho_{i-\frac{1}{2},j} + \rho_{i,j+\frac{1}{2}} + \rho_{i,j-\frac{1}{2}} \right) p_i^{n+1} + 1,
\]

\[
\nabla_d \cdot \tilde{u}_{ij} = \frac{\tilde{u}_{i+\frac{1}{2},j} - \tilde{u}_{i-\frac{1}{2},j}}{h} + \frac{\tilde{v}_{i,j+\frac{1}{2}} - \tilde{v}_{i,j-\frac{1}{2}}}{h},
\]

where \(\rho_i^{n+\frac{1}{2},j} = (\rho_{i+\frac{1}{2},j} + \rho_{i-\frac{1}{2},j})/2\) and the other terms are similarly defined.

The boundary condition for the pressure is

\[
n \cdot \nabla_d p_i^{n+1} = n \cdot \left( -\rho^n \frac{u_i^{n+1} - u_i^n}{\Delta t} - \rho^n (u \cdot \nabla_d u)^n + \frac{1}{Re} \Delta_d u^n + \frac{1}{W_e} \mathbf{SF}^n + \rho^n \frac{\mathbf{g}}{Fr^2} \right),
\]

where \(n\) is the unit normal vector to the domain boundary.

In our application of the phase-field to the Rayleigh-Taylor instability, we will use a periodic boundary condition to vertical boundaries and no slip boundary condition to the top and bottom domain. Therefore,

\[
n \cdot \nabla_d p_i^{n+1} = n \cdot \left( \frac{1}{W_e} \mathbf{SF}^n + \frac{\rho^n}{Fr^2} \mathbf{g} \right).
\]

The resulting linear system of Equation (5.11) is solved using a multigrid method [71], specifically, V-cycles with a Gauss-Seidel relaxation. Then the divergence-free normal velocities \(u_i^{n+1}\) and \(v_i^{n+1}\) are defined by

\[
u_i^{n+1} = \tilde{u}_{i+\frac{1}{2},j} - \frac{\Delta t}{\rho_i^{n+\frac{1}{2},j} h} (p_{i+1,j} - p_{i,j}), \quad v_i^{n+1} = \tilde{v}_{i,j+\frac{1}{2}} - \frac{\Delta t}{\rho_i^{n+\frac{1}{2},j} h} (p_{i,j+1} - p_{i,j}).
\]

We implement the unconditionally gradient stable scheme in Equations (5.7)-(5.8) with a nonlinear multigrid method. For a detailed description of the numerical method used in solving these equations, please refer to Refs. [72, 73].
Since we are interested in long time simulations, mass conservation is an important factor. Therefore, we use a conservative discretization of the convective part of the phase-field equation (5.7).

\[
((\phi u)_x + (\phi v)_y)_{ij}^n = \frac{u_{i+\frac{1}{2},j}^n (\phi_{ij}^n + \phi_{i+1,j}^n) - u_{i-\frac{1}{2},j}^n (\phi_{ij}^n + \phi_{i-1,j}^n)}{2h} + \frac{v_{i,j+\frac{1}{2}}^n (\phi_{i,j+1}^n + \phi_{ij}^n) - v_{i,j-\frac{1}{2}}^n (\phi_{ij}^n + \phi_{i,j-1}^n)}{2h}.
\]

These complete the one time step. Other projection type methods [67] can be used.
5.4. Modified Allen–Cahn solver

In this section, we propose a robust hybrid numerical method for crystal growth simulation. For simplicity of exposition we shall discretize Equations (2.32) and (2.33) in two-dimensional space, i.e., \( \Omega = (-l_1, l_1) \times (-l_2, l_2) \). Let \( N_x \) and \( N_y \) be positive even integers, \( h = 2l_1/N_x \) be the uniform mesh size, and \( \Omega_h = \{(x_i, y_j) : x_i = (i - 0.5)h, y_j = (j - 0.5)h, 1 \leq i \leq N_x, 1 \leq j \leq N_y\} \) be the set of cell-centers. Let \( c_{ij}^n \) be approximations of \( c(x_i, y_j, n\Delta t) \), where \( \Delta t = T/N_t \) is the time step, \( T \) is the final time, and \( N_t \) is the total number of time steps. The discrete differentiation operator is \( \nabla_d c_{ij} = (c_{i+1,j} - c_{i-1,j}, c_{i,j+1} - c_{i,j-1})/(2h) \). We then define the discrete Laplacian by \( \Delta_d c_{ij} = (c_{i+1,j} + c_{i-1,j} - 4c_{ij} + c_{i,j+1} + c_{i,j-1})/h^2 \). We discretize Equations (2.32) and (2.33):

\[
\epsilon^2(\phi^n) \frac{c_{ij}^{n+1} - c_{ij}^n}{\Delta t} = \epsilon^2(\phi^n) \Delta_d c_{ij}^{n+1,2} + 2\epsilon(\phi^n) \nabla_d \epsilon(\phi^n) \cdot \nabla_d c^n - F'(c_{ij}^{n+1}) - 4\lambda U^n F(c_{ij}^{n+1,1})
\]

\[
- \left( \epsilon'(\phi) \cdot \epsilon(\phi) c_y \right)_x^n + \left( \epsilon'(\phi) \cdot \epsilon(\phi) c_x \right)_y^n,
\]

\[
\frac{U^{n+1} - U^n}{\Delta t} = D \Delta_d U^{n+1} + \frac{c_{ij}^{n+1} - c_{ij}^n}{2\Delta t},
\]

where \( F(c) = 0.25(c^2 - 1)^2 \) and \( F'(c) = c(c^2 - 1) \). Here \( c_{ij}^{n+1,k} \) for \( k = 1, 2 \) are defined in the operator splitting scheme. We propose the following operator splitting scheme:

\[
\frac{\epsilon^2(\phi^n) c_{ij}^{n+1,1} - c_{ij}^n}{\Delta t} = 2\epsilon(\phi^n) \nabla_d \epsilon(\phi^n) \cdot \nabla_d c^n
\]

\[
- \left( \epsilon'(\phi) \cdot \epsilon(\phi) c_y \right)_x^n + \left( \epsilon'(\phi) \cdot \epsilon(\phi) c_x \right)_y^n,
\]

\[
\frac{\epsilon^2(\phi^n) c_{ij}^{n+1,2} - c_{ij}^{n+1,1}}{\Delta t} = \epsilon^2(\phi^n) \Delta_d c_{ij}^{n+1,2} - 4\lambda U^n F(c_{ij}^{n+1,1}),
\]

\[
\frac{\epsilon^2(\phi^n) c_{ij}^{n+1} - c_{ij}^{n+1,2}}{\Delta t} = -F'(c_{ij}^{n+1}).
\]

(5.12)
We can solve Equation (5.12) analytically by the method of separation of variables [252, 253]. The solution is given as follows:

\[
  c^{n+1} = \frac{c^{n+1,2}}{\sqrt{e^{-\frac{2\Delta t}{\epsilon^2(\phi^n)}} + (c^{n+1,2})^2 \left(1 - e^{-\frac{2\Delta t}{\epsilon^2(\phi^n)}}\right)}}.
\]

Finally, the proposed scheme can be written as follows:

\[
  \epsilon(\phi^n) \frac{c^{n+1,1} - c^n}{\Delta t} = 2\epsilon(\phi^n)x c^n_x + 2\epsilon(\phi^n)y c^n_y - (\epsilon'(\phi) \cdot c_y)_x + (\epsilon'(\phi) \cdot c_x)_y, \\
  \epsilon^2(\phi^n) \frac{c^{n+1,2} - c^{n+1,1}}{\Delta t} = \epsilon^2(\phi^n) \Delta_d c^{n+1,2} - 4\lambda U^n F(c^{n+1,1}), \tag{5.13}
\]

\[
  c^{n+1} = \frac{c^{n+1,2}}{\sqrt{e^{-\frac{2\Delta t}{\epsilon^2(\phi^n)}} + (c^{n+1,2})^2 \left(1 - e^{-\frac{2\Delta t}{\epsilon^2(\phi^n)}}\right)}}.
\]

\[
  \frac{U^{n+1} - U^n}{\Delta t} = D\Delta_d U^{n+1} + \frac{c^{n+1} - c^n}{2\Delta t}. \tag{5.14}
\]

Equations (5.13) and (5.14) can be solved by a multigrid method [233, 71].
5.5. The multi-component Cahn–Hilliard solver

The numerical solution of the $N$-component CH system uses a second-order accurate spatial discretization and a nonlinear splitting time stepping method. For simplicity and clarity of exposition, we will present the numerical method in 2D, but the extension to 3D is straightforward. Note that we only need to solve equations with $c_1, c_2, \ldots, c_{N-1}$ since $c_N = 1 - c_1 - c_2 - \cdots - c_{N-1}$ for the $N$-component CH system.

Let $c = (c_1, c_2, \ldots, c_{N-1})$ and $\mu = (\mu_1, \mu_2, \ldots, \mu_{N-1})$.

5.5.1. Discretization. Let $\Omega = (a, b) \times (c, d)$ be the computational domain in 2D, $N_x$ and $N_y$ be positive even integers, $h = (b - a)/N_x$ be the uniform mesh size, and $\Omega_h = \{(x_i, y_j) : x_i = (i - 0.5)h, \ y_j = (j - 0.5)h, 1 \leq i \leq N_x, 1 \leq j \leq N_y\}$ be the set of cell-centers.

Let $c_{ij}$ and $\mu_{ij}$ be approximations of $c(x_i, y_j)$ and $\mu(x_i, y_j)$. We first implement the zero Neumann boundary condition (3.3) and (3.6) by requiring that

$$D_x c_{\frac{i}{2},j} = D_x c_{N_x+\frac{1}{2},j} = D_y c_{i,\frac{1}{2}} = D_y c_{i,N_y+\frac{1}{2}} = 0,$$

$$D_x \mu_{\frac{i}{2},j} = D_x \mu_{N_x+\frac{1}{2},j} = D_y \mu_{i,\frac{1}{2}} = D_y \mu_{i,N_y+\frac{1}{2}} = 0,$$

where the discrete differentiation operators are

$$D_x c_{i+\frac{1}{2},j} = (c_{i+1,j} - c_{ij})/h \quad \text{and} \quad D_y c_{i,j+\frac{1}{2}} = (c_{ij+1} - c_{ij})/h.$$

We then define the discrete Laplacian by

$$\Delta_h c_{ij} = (D_x c_{i+\frac{1}{2},j} - D_x c_{i-\frac{1}{2},j}) + (D_y c_{i,j+\frac{1}{2}} - D_y c_{i,j-\frac{1}{2}})/h$$

and the discrete $L^2$ inner product by

$$(c, d)_h = h^2 \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} (c_{1ij} d_{1ij} + c_{2ij} d_{2ij} + \cdots + c_{N-1ij} d_{N-1ij}). \quad (5.15)$$
We also define a discrete norm associated with (5.15) as
\[ \|c\|^2 = (c, c)_h. \]

We define \( f(c) \) and \( 1 \) to \( f(c) = (f(c_1), f(c_2), \ldots, f(c_{N-1})) \) and \( 1 = (1, 1, \ldots, 1) \in \mathbb{R}^{N-1} \). We discretize Equations (3.4) and (3.5) in time by a nonlinear splitting algorithm:

\[
\frac{c_{ij}^{n+1} - c_{ij}^n}{\Delta t} = \Delta h \nu_{ij}^{n+1} + \Delta h \left( \beta(c_{ij}^n)1 - \frac{1}{4}c_{ij}^n \right), \\
\nu_{ij}^{n+1} = \varphi(c_{ij}^{n+1}) - \epsilon^2 \Delta h c_{ij}^{n+1},
\]

where the nonlinear function \( \varphi(c) = (\varphi_1(c), \varphi_2(c), \ldots, \varphi_{N-1}(c)) = f(c) + c/4 \).

In Equation (5.16), the variable Lagrangian multiplier \( \beta(c) \) is determined by the solutions at time level \( n \). By treating \( \beta(c) \) explicitly, there is no relation between the solutions at time level \( n+1 \). Thus the \( N \)-component \( \text{CH} \) system can be solved in a decoupled way, i.e.,

\[
\frac{c_{k,ij}^{n+1} - c_{k,ij}^n}{\Delta t} = \Delta h \nu_{k,ij}^{n+1} + \Delta h \left( \beta(c_{ij}^n) - \frac{1}{4}c_{k,ij}^n \right), \\
\nu_{k,ij}^{n+1} = \varphi(c_{k,ij}^{n+1}) - \epsilon^2 \Delta h c_{k,ij}^{n+1}, \text{ for } k = 1, 2, \ldots, N - 1.
\]

This means that we only solve the binary \( \text{CH} \) equation \( N - 1 \) times to solve the \( N \)-component \( \text{CH} \) system. The above discrete system is solved by a nonlinear multigrid method [71].
5.6. The multi-component Allen–Cahn solver

In this section, an unconditionally stable hybrid method is proposed for the $N$-component AC system. The unconditional stability means that arbitrarily large time steps can be used in the numerical algorithm. For simplicity and clarity of exposition, we will present the numerical method in 2D, but the extension to 3D is straightforward. Note that we only need to solve equations with $c_1, c_2, \ldots, c_{N-1}$ since $c_N = 1 - c_1 - c_2 - \cdots - c_{N-1}$ for the $N$-component AC system. Let $c = (c_1, c_2, \ldots, c_{N-1})$.

Let $\Omega = (a, b) \times (c, d)$ be the computational domain in 2D, $N_x$ and $N_y$ be positive even integers, $h = (b - a)/N_x$ be the uniform mesh size, and $\Omega_h = \{ (x_i, y_j) : x_i = (i - 0.5)h, \; y_j = (j - 0.5)h, \; 1 \leq i \leq N_x, \; 1 \leq j \leq N_y \}$ be the set of cell-centers. Let $c_{ij}$ be an approximation of $c(x_i, y_j)$. We first implement the zero Neumann boundary condition (4.2) by requiring that

$$D_x c_{ij} \frac{1}{2} = D_x c_{i, \frac{1}{2}j} = D_x c_{i, N_y} = D_y c_{i, N_y} = 0,$$

where the discrete differentiation operators are

$$D_x c_{i+\frac{1}{2}, j} = (c_{i+1, j} - c_{ij})/h \quad \text{and} \quad D_y c_{i, j+\frac{1}{2}} = (c_{ij+1} - c_{ij})/h.$$

We then define the discrete Laplacian by

$$\Delta_h c_{ij} = (D_x c_{i+\frac{1}{2}, j} - D_x c_{i-\frac{1}{2}, j}) + D_y c_{i, j+\frac{1}{2}} - D_y c_{i, j-\frac{1}{2}})/h$$

and the discrete $L^2$ inner product by

$$(c, d)_h = h^2 \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} (c_{1ij}d_{1ij} + c_{2ij}d_{2ij} + \cdots + c_{N-1ij}d_{N-1ij}). \quad (5.17)$$

We also define a discrete norm associated with (5.17) as

$$\|c\|^2 = (c, c)_h.$$
We define \( f(c) \) and \( 1 \) to \( f(c) = (f(c_1), f(c_2), \ldots, f(c_{N-1})) \) and \( 1 = (1, 1, \ldots, 1) \in \mathbb{R}^{N-1} \). We propose the following operator splitting scheme, which is an unconditionally stable hybrid scheme:

\[
\begin{align*}
\frac{c_{ij}^* - c_{ij}^0}{\Delta t} &= \Delta_h c_{ij}^* - \frac{\beta(c_{ij}^n)1}{\epsilon^2}, \quad (5.18) \\
\frac{c_{ij}^{n+1} - c_{ij}^*}{\Delta t} &= -\frac{f(c_{ij}^{n+1})}{\epsilon^2}. \quad (5.19)
\end{align*}
\]

Equation (5.18) is an implicit Euler’s scheme for \( c_t = \Delta c - (\beta(c)1)/\epsilon^2 \) with an initial condition \( c^n \). Using a von Neumann stability analysis [229], it can be seen that this scheme is unconditionally stable. The resulting implicit discrete system can be solved by a fast solver such as a multigrid method [230, 71]. Equation (5.19) can be considered as an approximation of the equation

\[
c_t = -\frac{f(c)}{\epsilon^2} \quad (5.20)
\]

by an implicit Euler’s method with the initial condition \( c^* \). We can solve Equation (5.20) analytically by the method of separation of variables [231]. The solution is given as follows:

\[
c_{ij}^{n+1} = 0.5 + \frac{c_{ij}^* - 0.5}{\sqrt{e^{\frac{-\Delta t}{2\epsilon}} + (2c_{ij}^* - 1)^2(1 - e^{\frac{-\Delta t}{2\epsilon}})}}.
\]

The proposed operator splitting algorithm is shown schematically in Fig. 5.4.

**Figure 5.4.** A hybrid numerical method for the \( N \)-component AC system.
In Equation (5.18), the variable Lagrangian multiplier $\beta(c)$ is determined by the solutions at time level $n$. By treating $\beta(c)$ explicitly, there is no relation between the solutions at time level $\ast$. Thus the $N$-component AC system can be solved in a decoupled way, i.e.,

$$\frac{c_{k,ij}^* - c_{k,ij}^n}{\Delta t} = \Delta_h c_{k,ij}^* - \frac{\beta(c_{ij}^n)}{\epsilon^2},$$

$$c_{k,ij}^{n+1} = 0.5 + \frac{c_{k,ij}^* - 0.5}{\sqrt{e^{-\Delta t/2\epsilon^2} + (2c_{k,ij}^* - 1)^2(1 - e^{-\Delta t/2\epsilon^2})}}, \text{ for } k = 1, 2, \ldots, N - 1.$$

This means that we only solve the binary AC equation $N - 1$ times to solve the $N$-component AC system.
Chapter 6

Numerical results

6.1. Numerical experiments for regularized Dirac delta functions

We now present numerical results to show the performance of regularized Dirac delta functions as (1) a postprocessing of the phase-field solution; length of line and circle, Rayleigh-Taylor instability, deformation of a circle by a single vortex, a three-dimensional deformation field, and triply periodic minimal surfaces, and (2) a representation of the surface tension force; reduction of spurious velocities and pressure jump across the drop. And we also demonstrate relation between the interfacial width and grid size. Across the interfacial region, the concentration field varies from 0.1 to 0.9 over a distance of approximately $4\sqrt{2}\epsilon \tanh^{-1}(0.8)$. Therefore, if we want this value to be approximately $m (> 0)$ grid points, the $\epsilon$ value needs to be taken as $\epsilon_m = hm/[4\sqrt{2}\tanh^{-1}(0.8)]$. Figure 6.1 shows the concentration $c(x) = 0.5(1 + \tanh(x/(2\sqrt{2}\epsilon_m)))$ with $m = 4, 8, 12, \text{and} 16$. We will use various $\epsilon$ that is suitable for each problem.

6.1.1. The Dirac delta function as a postprocessing of the phase-field solution. In this section, we investigate the performance of each delta function as an interface length or a surface area calculation tool. The interface length $L$ and surface
6.1. NUMERICAL EXPERIMENTS FOR REGULARIZED DIRAC DELTA FUNCTIONS

Figure 6.1. The concentration $c(x) = 0.5(1 + \tanh(x/(2\sqrt{2}\epsilon_m)))$ with $m = 4, 8, 12,$ and $16$.

area $\mathcal{A}$ are defined as:

$$\mathcal{L}(c) = \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} \delta(c_{ij})h^2 \quad \text{and} \quad \mathcal{A}(c) = \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} \sum_{k=1}^{N_z} \delta(c_{ijk})h^3,$$

where $N_x$, $N_y$, and $N_z$ are the number of grid points in the $x$, $y$, and $z$ directions, respectively. The discrete composition field $c_{ij}$ (or $c_{ijk}$) is located at cell-centers and $h$ is the uniform mesh size.

When there is no theoretical value of an interface length or a surface area, we need a reasonable value corresponding to the theoretical value. To obtain a reasonable value, we take the initial condition as

$$c(x, y, 0) = \frac{1}{2} \left( 1 + \tanh \left( \frac{0.25 - \sqrt{(x - 0.5)^2 + (y - 0.5)^2}}{2\sqrt{2}\epsilon_4} \right) \right)$$

on the computational domain $\Omega = (0, 1) \times (0, 1)$, which represents a circle with a radius 0.25. We calculate the length of a 0.5-level contour using MATLAB. The results with increasingly finer grids are given in Table 6.1.

<table>
<thead>
<tr>
<th>Mesh</th>
<th>16$^2$</th>
<th>32$^2$</th>
<th>64$^2$</th>
<th>128$^2$</th>
<th>Theoretical value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface length</td>
<td>1.565632</td>
<td>1.569770</td>
<td>1.570572</td>
<td>1.570778</td>
<td>$2\pi r \approx 1.570796$</td>
</tr>
</tbody>
</table>
Next, the initial condition is

\[
c(x, y, z, 0) = \frac{1}{2} \left(1 + \tanh \left(\frac{0.25 - \sqrt{(x - 0.5)^2 + (y - 0.5)^2 + (z - 0.5)^2}}{2\sqrt{2}\epsilon}\right)\right)
\]

on \(\Omega = (0, 1) \times (0, 1) \times (0, 1)\), which represents a sphere with a radius 0.25. We compute the area of an isosurface by summation of the areas of all triangle tiles in the isosurface using MATLAB. The results with increasingly finer grids are given in Table 6.2. The results in Tables 6.1 and 6.2 suggest that the length of contour and the area of isosurface agree well with the theoretical value.

**Table 6.2.** Comparison between the theoretical value and the area of isosurface.

<table>
<thead>
<tr>
<th>Mesh</th>
<th>32^3</th>
<th>64^3</th>
<th>128^3</th>
<th>256^3</th>
<th>Theoretical value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Surface area</td>
<td>0.781695</td>
<td>0.784493</td>
<td>0.785185</td>
<td>0.785364</td>
<td>(4\pi r^2 \approx 0.785398)</td>
</tr>
</tbody>
</table>

6.1.1. Test 1: length of line and circle. In Section 2.3, we explored the performance of delta functions by the interface profile using Mathematica. To numerically explore the performance, we consider two initial conditions on \(\Omega = (0, 1) \times (0, 1)\):

\[
c(x, y, 0) = \frac{1}{2} \left(1 + \tanh \left(\frac{0.5 - x}{2\sqrt{2}\epsilon}\right)\right) \quad \text{and} \quad c(x, y, 0) = \frac{1}{2} \left(1 + \tanh \left(\frac{0.25 - \sqrt{(x - 0.5)^2 + (y - 0.5)^2}}{2\sqrt{2}\epsilon}\right)\right)
\]

for \(a = 0.5\epsilon_8, \epsilon_8, \text{ and } 2\epsilon_8\). We choose \(h = 1/128\) and \(\Delta t = 10h\).

In Figures 6.2 (case of line) and 6.3 (case of circle), (a), (b), and (c) show the contour lines of concentration with \(a = 0.5\epsilon_8, \epsilon_8, \text{ and } 2\epsilon_8\), respectively. The length of line and circle for each delta function is given in Table 6.3. From Table 6.3, it is observed that \(\delta_6\) accurately calculates the length of line and circle regardless of whether an interface transition is compressed \((a = 0.5\epsilon_8)\) or stretched \((a = 2\epsilon_8)\). Figures 6.2 (d) and 6.3 (d) show the percentage error of each delta function of cases of line and circle by representing them as histograms of various sizes, respectively.
6.1. NUMERICAL EXPERIMENTS FOR REGULARIZED DIRAC DELTA FUNCTIONS

6.1.1.2. Test 2: Rayleigh–Taylor instability. When a heavy fluid is superposed over a light fluid in a gravitational field, the fluid interface is unstable. Any perturbation of this interface tends to grow with time, producing the phenomenon known as the Rayleigh–Taylor instability. This phenomenon represents the penetration of both heavy and light fluids into each other. The Rayleigh–Taylor instability for a fluid in a gravitational field was originally introduced by Rayleigh [134] and later applied to

Figure 6.2. The length of line. (a), (b), and (c) show the contour lines of concentration with $a = 0.5\epsilon_8$, $\epsilon_8$, and $2\epsilon_8$, respectively. Contour levels are 0.1, 0.2, \ldots, 0.9. (d) the percentage error of each delta function.

Table 6.3. The length of line and circle for each delta function.

<table>
<thead>
<tr>
<th>Case</th>
<th>$a = 0.5\epsilon_8$</th>
<th>$\epsilon_8$</th>
<th>$2\epsilon_8$</th>
<th>$a = \epsilon_8$</th>
<th>$2\epsilon_8$</th>
<th>$3\epsilon_8$</th>
<th>$4\epsilon_8$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contour value</td>
<td>$0.5\epsilon_8$</td>
<td>$\epsilon_8$</td>
<td>$2\epsilon_8$</td>
<td>$0.5\epsilon_8$</td>
<td>$\epsilon_8$</td>
<td>$2\epsilon_8$</td>
<td>$3\epsilon_8$</td>
</tr>
<tr>
<td>$\delta_1$</td>
<td>1.8556</td>
<td>0.9804</td>
<td>0.4975</td>
<td>2.9677</td>
<td>1.5473</td>
<td>0.7823</td>
<td></td>
</tr>
<tr>
<td>$\delta_2$</td>
<td>0.5000</td>
<td>1.0000</td>
<td>2.0000</td>
<td>0.7854</td>
<td>1.5708</td>
<td>3.1416</td>
<td></td>
</tr>
<tr>
<td>$\delta_3$</td>
<td>1.1778</td>
<td>0.9902</td>
<td>1.2488</td>
<td>1.8766</td>
<td>1.5590</td>
<td>1.9619</td>
<td></td>
</tr>
<tr>
<td>$\delta_4$</td>
<td>1.7929</td>
<td>0.9720</td>
<td>0.4964</td>
<td>2.8927</td>
<td>1.5373</td>
<td>0.7810</td>
<td></td>
</tr>
<tr>
<td>$\delta_5$</td>
<td>0.5000</td>
<td>1.0000</td>
<td>2.0000</td>
<td>0.7854</td>
<td>1.5708</td>
<td>3.1413</td>
<td></td>
</tr>
<tr>
<td>$\delta_6$</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.5708</td>
<td>1.5706</td>
<td>1.5704</td>
<td></td>
</tr>
<tr>
<td>$\delta_7$</td>
<td>0.7500</td>
<td>1.0000</td>
<td>1.5000</td>
<td>1.1781</td>
<td>1.5707</td>
<td>2.3558</td>
<td></td>
</tr>
<tr>
<td>$\delta_8$</td>
<td>1.7353</td>
<td>0.9645</td>
<td>0.4955</td>
<td>2.8278</td>
<td>1.5284</td>
<td>0.7799</td>
<td></td>
</tr>
</tbody>
</table>
all accelerated fluids by Taylor [135]. In order to simulate the Rayleigh–Taylor instability, the Navier–Stokes–Cahn–Hilliard equations (NSCH) are preferred. The NSCH
6.1. NUMERICAL EXPERIMENTS FOR REGULARIZED DIRAC DELTA FUNCTIONS

Equations can be written in a dimensionless form

\[ \rho(c)(u_t + u \cdot \nabla u) = -\nabla p + \frac{1}{Re} \Delta u + \frac{\rho(c)}{Fr} g, \]

\[ \nabla \cdot u = 0, \]

\[ c_t + \nabla \cdot (cu) = \frac{1}{Pe} \Delta \mu, \]

\[ \mu = f(c) - C \Delta c, \]

where \( u \) is the velocity, \( p \) is the pressure, \( \rho(c) = \rho_1 c + \rho_2 (1 - c) \) is the variable density (\( \rho_1 \) and \( \rho_2 \) are the densities of the heavier and lighter fluid, respectively), and \( g = (0, -1) \). The dimensionless parameters are the Reynolds number, \( Re = \rho_s U_s L_s / \eta \), Froude number, \( Fr = U_s^2 / (gL_s) \), Peclet number, \( Pe = U_s L_s / (\mu_s) \), and Cahn number, \( C = \epsilon^2 / (\mu_s L_s^2) \). The values with lower * are characteristic values of corresponding ones, \( \eta \) is the viscosity, and \( g \) is the acceleration due to gravity. Here, the effect of the surface tension is negligible. For a detailed description of the numerical method used in solving the NSCH equations, please refer to Ref. [136].

We calculate the length of the interface with three different Peclet numbers. The initial condition is

\[ c(x, y, 0) = \frac{1}{2} \left( 1 + \tanh \left( \frac{y - 2 - 0.1 \cos(2\pi x)}{2\sqrt{2}\epsilon} \right) \right) \]

on \( \Omega = (0, 1) \times (0, 4) \), which represents a planar interface superimposed by a perturbation of wave number \( k = 1 \) and amplitude 0.1. The density ratio is \( \rho_1 : \rho_2 = 3 : 1 \) and we use the simulation parameters such as the uniform grids \( h = 1/128 \), \( \Delta t = 0.00125 \), \( \epsilon = 0.01 \), and \( Re = 3000 \).

Figure 6.4 (a) shows the evolution of the interface with \( Pe = 1/\epsilon \) at times \( t = 0 \), 1, and 2. The results are given in Table 6.4. In the numerical simulations of the
Rayleigh–Taylor instability, as the $Pe$ number increases, the width of an interface transition becomes non-uniform (see Figures 6.4 (b), (c), and (d)). As a result, the percentage error of almost all delta functions is high (see Figure 6.5). But the result obtained using $\delta_6$ is in better agreement with the contour value, regardless of the $Pe$ number.

![Figure 6.4](image)

**Figure 6.4.** (a) the evolution of the interface with $Pe = 1/\epsilon$ at times $t = 0, 1, \text{and} 2$. The effect of the Pelet number on the temporal evolution of the interface at time $t = 2$: (b) $Pe = 0.1/\epsilon$, (c) $Pe = 1/\epsilon$, and (d) $Pe = 10/\epsilon$. Contour levels are 0.1, 0.2, ..., 0.9.

<table>
<thead>
<tr>
<th>$Pe$</th>
<th>Contour value</th>
<th>$\delta_1$</th>
<th>$\delta_2$</th>
<th>$\delta_3$</th>
<th>$\delta_4$</th>
<th>$\delta_5$</th>
<th>$\delta_6$</th>
<th>$\delta_7$</th>
<th>$\delta_8$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1/\epsilon</td>
<td>5.169</td>
<td>4.955</td>
<td>5.478</td>
<td>5.217</td>
<td>4.953</td>
<td>5.798</td>
<td>5.097</td>
<td>5.447</td>
<td>4.950</td>
</tr>
<tr>
<td>1/\epsilon</td>
<td>5.832</td>
<td>6.385</td>
<td>5.532</td>
<td>5.958</td>
<td>6.222</td>
<td>4.611</td>
<td>5.924</td>
<td>5.267</td>
<td>6.150</td>
</tr>
</tbody>
</table>

6.1.1.3. **Test 3: deformation of a circle by a single vortex.** In this test, a circle is deformed with a velocity field defined by $u = k \sin^2(\pi x) \sin(2\pi y)$ and $v = -k \sin(2\pi x) \sin^2(\pi y)$. In order to demonstrate the capability of each delta function...
in an extreme velocity field, we choose \( k = 100 \), which is much larger than values used in the previous experiments. This flow satisfies \( u = v = 0 \) on the boundaries of the unit square domain. As shown in Figure 6.6 (a) by the dotted line, initially the circle has a radius of 0.15 and is centered at \((0.50, 0.75)\) in the unit square domain. The phase-field is initialized to \( c = 1 \) and \( c = 0 \) inside and outside the circle, respectively. The advection by the vorticity field causes the circle to evolve into a filament that spirals toward the vortex center at \((0.5, 0.5)\). The numerical solutions are computed on the uniform grids \( h = 1/128 \) and the calculation is run with \( \Delta t = 0.00125/k \) and \( \epsilon = 0.01 \). Figure 6.6 (a) shows the evolution of the interface at times \( t = 0, 0.0025, \) and \( 0.005 \). The results are given in Table 6.5. For most of delta functions, except \( \delta_6 \), the error is large because of the compressed interfacial transition (see Figure 6.6 (b)). But, as seen in Section 2.3, \( \delta_6 \) yields a good result even though the interfacial transition is compressed (see Figure 6.6 (c)).
6.1. NUMERICAL EXPERIMENTS FOR REGULARIZED DIRAC DELTA FUNCTIONS

Figure 6.6. (a) the evolution of the interface at times $t = 0$, 0.0025, and 0.005. (b) Contour lines of concentration. Contour levels are $0.1, 0.2, \ldots, 0.9$. (c) the percentage error of each delta function.

Table 6.5. Deformed circle: the interface length for each delta function.

<table>
<thead>
<tr>
<th>Contour value</th>
<th>$\delta_1$</th>
<th>$\delta_2$</th>
<th>$\delta_3$</th>
<th>$\delta_4$</th>
<th>$\delta_5$</th>
<th>$\delta_6$</th>
<th>$\delta_7$</th>
<th>$\delta_8$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.968</td>
<td>3.192</td>
<td>2.219</td>
<td>3.117</td>
<td>1.218</td>
<td>1.919</td>
<td>1.568</td>
<td>3.060</td>
<td></td>
</tr>
</tbody>
</table>

6.1.1.4. Test 4: three-dimensional deformation field. We consider the problem of a deforming sphere in a velocity field given by $u = 2\sin^2(\pi x)\sin(2\pi y)\sin(2\pi z)$, $v = -\sin(2\pi x)\sin^2(\pi y)\sin(2\pi z)$, and $w = -\sin(2\pi x)\sin(2\pi y)\sin^2(\pi z)$ [137]. A simulation of this problem has previously been performed by Enright et al. using the level-set method [138]. This flow satisfies $u = v = w = 0$ on the boundaries of the unit cube domain. A sphere of radius 0.15 is placed within a unit computational domain at $(0.35, 0.35, 0.35)$. We take $h = 1/256$, $\Delta t = 2.5e^{-4}$, $\epsilon_3$, and $Pe = 0.1/\epsilon_3$.

Figure 6.7 shows the evolution of the interface at times $t = 0$, 0.2, 0.4, and 0.6. Each figure can be compared with the figures from the level-set computation in [138]. The sphere is entrained by two rotating vortices and then is compressed into a pancake-like shape. The surface of the pancake-like shape becomes stretched out. Parts of the
interface thin out to about a few grid and almost all delta functions have difficulty to calculate exactly the surface area of this thin interface (see the results in Table 6.6). But $\delta_6$ has very little error even if the surface is deformed. Figure 6.8 shows the percentage error of each delta function by representing them as histograms of various sizes.

![Figure 6.7. Deformation test: the evolution of the interface.](image)

**Table 6.6.** Deformed sphere: the surface area for each delta function.

<table>
<thead>
<tr>
<th>Isosurface value</th>
<th>$\delta_1$</th>
<th>$\delta_2$</th>
<th>$\delta_3$</th>
<th>$\delta_4$</th>
<th>$\delta_5$</th>
<th>$\delta_6$</th>
<th>$\delta_7$</th>
<th>$\delta_8$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.586</td>
<td>0.565</td>
<td>0.567</td>
<td>0.566</td>
<td>0.545</td>
<td>0.634</td>
<td>0.596</td>
<td>0.615</td>
<td>0.528</td>
</tr>
</tbody>
</table>
6.1.1.5. Test 5: triply periodic minimal surfaces. Triply periodic minimal surfaces (TPMS, see Figure 6.9) are of special interest because they appear in a variety of real structures such as silicates, bicontinuous mixtures, lyotropic colloids, detergent films, lipid bilayers, and biological formations [139]. One important application of TPMS is tissue scaffolds. Tissue scaffolds should have an optimal surface area and pore size to restore function or regenerate tissue more efficiently. The TPMS morphology has been successfully adapted to tissue scaffolds [140]. To calculate surface areas of TPMS, we take the periodic nodal surface approximations of the P, D, and G TPMS [141]:

\[
P(x, y, z) = \cos 2\pi x + \cos 2\pi y + \cos 2\pi z + 0.5,
\]
\[
D(x, y, z) = \cos 2\pi x \cos 2\pi y \cos 2\pi z - \sin 2\pi x \sin 2\pi y \sin 2\pi z + 0.5,
\]
\[
G(x, y, z) = \sin 2\pi x \cos 2\pi y + \sin 2\pi z \cos 2\pi x + \sin 2\pi y \cos 2\pi z + 0.5.
\]

For these calculations, we employ the computational domain \( \Omega = (0, 1) \times (0, 1) \times (0, 1) \) with \( h = 1/256, \Delta t = 0.5h, \) and \( \epsilon = 0.01. \) We stop the numerical computations when
the discrete $l_2$-norm of the difference between $(n + 1)$th and $n$th time step solutions becomes less than $10^{-6}$. That is $||c^{n+1} - c^n|| \leq 10^{-6}$. In Table 6.7, we compare surface areas obtained using each delta function with those obtained by Jung et al. [142]. As we can see from Figure 6.10, the error of $\delta_6$ is small.

![Figure 6.9](image URL)

**Figure 6.9.** Triply periodic minimal surfaces: (a) Schwarz P (primitive), (b) Schwarz D (diamond), and (c) Schoen G (gyroid).

<table>
<thead>
<tr>
<th>TPMS (the results in [142])</th>
<th>$\delta_1$</th>
<th>$\delta_2$</th>
<th>$\delta_3$</th>
<th>$\delta_4$</th>
<th>$\delta_5$</th>
<th>$\delta_6$</th>
<th>$\delta_7$</th>
<th>$\delta_8$</th>
</tr>
</thead>
<tbody>
<tr>
<td>P surface (2.34)</td>
<td>2.21</td>
<td>2.24</td>
<td>2.22</td>
<td>2.11</td>
<td>2.29</td>
<td>2.35</td>
<td>2.32</td>
<td>2.02</td>
</tr>
<tr>
<td>D surface (3.84)</td>
<td>3.57</td>
<td>3.63</td>
<td>3.60</td>
<td>3.39</td>
<td>3.80</td>
<td>3.82</td>
<td>3.81</td>
<td>3.22</td>
</tr>
<tr>
<td>G surface (3.10)</td>
<td>2.88</td>
<td>2.93</td>
<td>2.91</td>
<td>2.74</td>
<td>3.14</td>
<td>3.08</td>
<td>3.11</td>
<td>2.60</td>
</tr>
</tbody>
</table>

### 6.1.2. The Dirac delta function in the surface tension force formulation.

An accurate approximation of the surface tension force is essential for solving two-phase incompressible fluid flows. The continuum surface force (CSF) model of Brackbill et al. [120] is employed extensively to model the surface tension force of two-phase incompressible fluid flows in volume-of-fluid [99, 121, 124, 143], level-set [113, 39], and phase-field [72, 65, 55, 63] methods. In the CSF model, surface tension forces acting on the interface are transformed to volume forces in regions near the interface via a delta function, $\mathbf{SF} = \sigma \kappa \delta \hat{\mathbf{n}}$, where $\sigma$ is the surface tension coefficient, $\kappa$ is the curvature, $\delta$
6.1. NUMERICAL EXPERIMENTS FOR REGULARIZED DIRAC DELTA FUNCTIONS

Figure 6.10. TPMS: the percentage error of each delta function.

is a delta function concentrated on the interface, and $\mathbf{n}$ is the unit outward normal to the surface and defined by $\mathbf{n} = \nabla c/|\nabla c|$. We note the identity

$$\nabla \cdot \mathbf{n} = -\kappa. \quad (6.1)$$

For a derivation of Equation (6.1), see Ref. [120]. The governing equations for two-phase incompressible fluid flows can be written as [72, 65]

$$\rho(u_t + u \cdot \nabla u) = -\nabla p + \eta \Delta u - \sigma \nabla \cdot \left( \frac{\nabla c}{|\nabla c|} \right) \delta(c) \frac{\nabla c}{|\nabla c|}, \quad (6.2)$$

$$\nabla \cdot u = 0,$$

$$c_t + \nabla \cdot (cu) = M \Delta \mu,$$

$$\mu = f(c) - \epsilon^2 \Delta c.$$ 

For details of the numerical solution, we refer to Ref. [65]. In this section, we present two tests for the surface tension force.
6.1. NUMERICAL EXPERIMENTS FOR REGULARIZED DIRAC DELTA FUNCTIONS

6.1.2.1. Reduction of spurious velocities. Spurious or parasitic velocities are unphysical currents that arise from a slight imbalance between stresses in the interfacial region. There are a number of papers for spurious velocities in incompressible flow problems [102, 144, 145, 146, 147]. Approximating and choosing the surface tension force formulation accurately is important since an improper formulation will lead to spurious velocities. In this section, we compare the performance of each delta function with the similar test problem in [124]. The computational domain is $\Omega = (0, 1) \times (0, 1)$ and the time step is $\Delta t = 10^{-5}$. The boundary conditions are zero velocity at the top and bottom walls, and periodicity in $x$-direction. Initially, a circular drop is centered at $(0.5, 0.5)$, with radius $a = 0.1$ and surface tension coefficient $\sigma = 0.357$. Both fluids have equal density, 4, and viscosity, 1. The initial velocity field is zero. The exact solution is zero velocity for all time. In dimensionless terms, the relevant parameter is the Ohnesorge number $Oh = \frac{\eta}{\sqrt{\sigma \rho a}} \sim 2.6463$.

Table 6.8 shows the convergence of spurious velocities as we refine the mesh size for each delta function. Most delta functions show minor decreases as we refine the mesh. But $\delta_1$, $\delta_6$, and $\delta_7$ show a linear convergence of spurious velocities, and $\delta_6$ effectively eliminates spurious velocities than $\delta_1$ and $\delta_7$. In Figure 6.11, scaled velocity vector plots at 200th time step with $\Delta t = 10^{-5}$ are shown. These show the locations of spurious velocities with mesh refinement, (a) $h = 1/64$, (b) $1/128$, and (c) $1/256$ with $\epsilon = 0.64h$. Spurious velocities are mainly concentrated in the neighborhood of the interface. The convergence of spurious velocities is evident as we refine the mesh size $h$ and interface parameter $\epsilon$. 
Table 6.8. \(l_2\)-norm of velocity at 200th time step with \(\epsilon = 0.64h\) and \(\Delta t = 10^{-5}\).

<table>
<thead>
<tr>
<th>Case</th>
<th>(\delta_1)</th>
<th>(\delta_2)</th>
<th>(\delta_3)</th>
<th>(\delta_4)</th>
<th>(\delta_5)</th>
<th>(\delta_6)</th>
<th>(\delta_7)</th>
<th>(\delta_8)</th>
</tr>
</thead>
<tbody>
<tr>
<td>64(^2)</td>
<td>2.064e-5</td>
<td>4.460e-5</td>
<td>3.219e-5</td>
<td>3.010e-5</td>
<td>3.019e-5</td>
<td>2.232e-5</td>
<td>2.599e-5</td>
<td>4.463e-5</td>
</tr>
<tr>
<td>128(^2)</td>
<td>1.112e-5</td>
<td>4.345e-5</td>
<td>2.663e-5</td>
<td>1.901e-5</td>
<td>1.820e-5</td>
<td>1.018e-5</td>
<td>1.389e-5</td>
<td>2.667e-5</td>
</tr>
<tr>
<td>256(^2)</td>
<td>5.880e-6</td>
<td>4.264e-5</td>
<td>2.350e-5</td>
<td>1.325e-5</td>
<td>1.318e-5</td>
<td>4.653e-6</td>
<td>8.362e-6</td>
<td>2.137e-5</td>
</tr>
</tbody>
</table>

Figure 6.11. Scaled velocity vector plots at 200th time step with \(\Delta t = 10^{-5}\). These show the locations of spurious velocities with mesh refinement, (a) \(h = 1/64\), (b) 1/128, and (c) 1/256 with \(\epsilon = 0.64h\). The solid line in each figure represents the interface. The length of the arrow is proportional to the magnitude of the velocity vector.

6.1.2.2. Pressure jump across the drop. Let us consider the equilibrium of a drop placed within another fluid. Let the drop composition be defined as

\[
c(x, y) = \frac{1}{2} \left(1 + \tanh \left(0.1 - \sqrt{(x - 0.5)^2 + (y - 0.5)^2} \sqrt{2\epsilon}\right)\right).
\]

In the equilibrium state of a droplet, the velocity vanishes \((u \equiv 0)\) and therefore Equation (6.2) reduces to Equation (6.3) and therefore pressure gradient should balance surface tension force.

\[
\nabla p = -\sigma \nabla \cdot \left(\frac{\nabla c}{|\nabla c|}\right) \delta(c) \frac{\nabla c}{|\nabla c|}.
\]

We solve Equation (6.4) numerically by taking the divergence operator to Equation (6.3) with \(\sigma = 20\), 128 \(\times\) 128 mesh, computational domain \(\Omega = (0, 1) \times (0, 1)\), and
\[ \Delta p = -\sigma \nabla \cdot \left( \nabla \cdot \left( \frac{\nabla c}{|\nabla c|} \right) \delta(c) \frac{\nabla c}{|\nabla c|} \right). \] (6.4)

From Laplace’s formulation, we can obtain the theoretical prediction of the pressure jump inside an infinite cylinder as \( \Delta p_{\text{theo}} = \sigma/R \), where \( R \) is the drop radius. In this test, the pressure jump \( \Delta p_{\text{theo}} \) is 200. This value is compared with the difference between the maximum and minimum computed drop pressures obtained with each delta function, \([p]\), defined as

\[ [p] = \max_{i,j} p_{ij} - \min_{i,j} p_{ij}. \]

As shown in Table 6.9, the numerical pressure jump \([p]\) obtained using \( \delta_6 \) is in excellent agreement with the theoretical prediction.

<table>
<thead>
<tr>
<th>Case</th>
<th>( \delta_1 )</th>
<th>( \delta_2 )</th>
<th>( \delta_3 )</th>
<th>( \delta_4 )</th>
<th>( \delta_5 )</th>
<th>( \delta_6 )</th>
<th>( \delta_7 )</th>
<th>( \delta_8 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>([p])</td>
<td>190.40</td>
<td>199.08</td>
<td>194.58</td>
<td>182.76</td>
<td>197.31</td>
<td>199.70</td>
<td>198.48</td>
<td>176.90</td>
</tr>
</tbody>
</table>

### 6.1.3. Relation between the interfacial width and grid size.

With the mass conserving boundary condition \( (\nabla \mu \cdot \mathbf{n} = 0 \text{ on } \partial\Omega) \), the differentiation of the total mass yields

\[ \frac{d}{dt} \int_{\Omega} \phi \, d\mathbf{x} = \int_{\Omega} \phi_t \, d\mathbf{x} = \int_{\Omega} \Delta \mu \, d\mathbf{x} = \int_{\partial\Omega} \nabla \mu \cdot \mathbf{n} \, ds = 0, \] (6.5)

where \( \phi \) is the phase-field variable and we redefine \( \phi \) as the difference between the concentration of the two components. Equation (6.5) means that the solution of the Cahn–Hilliard equation conserves mass over the entire domain. Even though the phase-field variable is conserved globally, the mass of a drop is liable to variations as the \( \phi \) evolves. Theoretically, such variations vanish as the interfacial thickness approaches
zero. But, in practice, the interfacial thickness is finite. This was pointed out by Yue et al. [133]. To minimize the variations of mass, Yue et al. provided the guidelines on how to pick the interfacial width and grid size relative to the radius of curvature. And the authors calculated the shift of the phase-field variable and found that the shift $\delta \phi$ is proportional to $\epsilon/r_0$, where $r_0$ is the initial drop radius.

In this section, we perform numerical simulations of the spontaneous shrinking of a drop. Drop with initial radius $r_0 = 0.75$ and interfacial width $\epsilon_{8.3281}$ (this value corresponds to the value in Section 2.2 in Ref. [133]) and $\epsilon_{30}$ are simulated. The initial drop is centered at the center of the computational domain $\Omega = (-2, 2) \times (-2, 2)$ and a $128 \times 128$ mesh is used. Figures 6.12 (a) and (b) show the phase-field $\phi$ at $y = 0$ for $\epsilon_{8.3281}$ and $\epsilon_{30}$, respectively. In the case of $\epsilon_{8.3281}$, a grid size $h$ is $\epsilon_{8.3281}/2$ and a $128 \times 128$ mesh is sufficient to guarantee numerical accuracy. Thus, we can see that $\phi$ at the equilibrium state shifts slightly away from the initial state. But, in the case of $\epsilon_{30}$, a $128 \times 128$ mesh is too coarse compared with $\epsilon_{30}$. As a result, the phase-field disperses (see Figure 6.12 (b)). In Figure 6.12 (c), we use a $512 \times 512$ mesh with $\epsilon_{30}$ and obtain more accurate result than result in Figure 6.12 (b). Therefore, to guarantee numerical accuracy, we need to choose properly an interfacial width and grid size.
6.1. NUMERICAL EXPERIMENTS FOR REGULARIZED DIRAC DELTA FUNCTIONS

Figure 6.12. Each figure shows the phase-field at $y = 0$. The interfacial width and grid size for each case are given below each figure.
6.2. Numerical experiments for snow crystal growth

In this section we perform numerical experiments for two-dimensional solidification to validate that our proposed scheme is accurate, efficient, and robust. Unless otherwise specified, we take the initial state as

\[ c(x, y, 0) = \tanh \left( \frac{R_0 - \sqrt{x^2 + y^2}}{\sqrt{2}} \right) \quad \text{and} \quad U(x, y, 0) = \begin{cases} 0 & \text{if } c > 0 \\ \Delta & \text{else.} \end{cases} \]

The zero level set \((c = 0)\) represents a circle of radius \(R_0\). From the dimensionless variable definition the value \(U = 0\) corresponds to the melting temperature of the pure material, while \(U = \Delta\) is the initial undercooling. The capillary length, \(d_0\), is defined as \(d_0 = a_1/\lambda\) [235, 250, 262] with \(a_1 = 0.8839\) [246, 247, 262] and \(\lambda = 3.1913\) [262].

6.2.1. Convergence test. To obtain an estimate of the convergence rate, we perform a number of simulations for 6-fold crystal growth problem on a set of increasingly finer grids. The computational domain is \(\Omega = (-100, 100)^2\) and we take \(R_0 = 15d_0\), \(\epsilon_6 = 0.02\), and \(\Delta = -0.55\). The numerical solutions are computed on the uniform grids \(h = 200/2^n\) and with corresponding time steps \(\Delta t = 0.6/2^{n-8}\) for \(n = 8, 9, 10,\) and 11. The calculations are run up to time \(T = 150\). We define the error to be the discrete \(l_2\)-norm of the difference between that grid and the average of the next finer grid cells covering it:

\[ e_{h/h}^{ij} = c_{h}^{ij} - \left( c_h^{2i-1,2j-1} + c_h^{2i-1,2j} + c_h^{2i,2j-1} + c_h^{2i,2j} \right)/4. \]

The rate of convergence is defined as:

\[ \log_2(\| e_{h/2}^{ij} \|_2 / \| e_{h/4}^{ij} \|_2). \]
The errors and rates of convergence are given in Table 6.10. The results suggest that the scheme is indeed second order accurate in space. Figure 6.13 shows the convergence of numerical results under mesh refinement.

Table 6.10. Error and $l_2$ convergence result.

<table>
<thead>
<tr>
<th>mesh size</th>
<th>rate</th>
<th>256 $-\times$ 512</th>
<th>512 $-\times$ 1024</th>
<th>1024 $-\times$ 2048</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.477E$-4$</td>
<td>1.96</td>
<td>1.405E$-4$</td>
<td>2.01</td>
<td>3.487E$-5$</td>
</tr>
</tbody>
</table>

Figure 6.13. Convergence of numerical results under mesh refinement.

Next, we consider the evolution of the interface with different time steps in order to investigate the effect of time step. A 1024 $\times$ 1024 mesh is used on the domain $\Omega = (-200,200)^2$ with $R_0 = 50d_0$, $\epsilon_6 = 0.02$, and $\Delta = -0.55$. Figure 6.14 (a) shows the interfaces at time $T = 1200$ with different time steps $\Delta t = 0.6$, 0.3, and 0.15. Figure 6.14 (b) shows the velocity of the tip versus time. For the calculation of the crystal tip velocity, refer to Ref. [252]. The velocity $V$ of the tip at time $T = 1200$ versus time step is shown in Figure 6.14 (c). Here, we define the error between the fitting
velocity $\tilde{V}$ and $V$ as $E_i = |\tilde{V}_i - V_i|/V_i$. In Figure 6.14 (c), the linear fit $\tilde{V}$ is done using the MATLAB function “polyfit” and the errors on the index $i$ are calculated by the MATLAB function “polyval” on the results of the linear fit. In this test, the $l_2$ error is 0.54%. Therefore the results suggest that the convergence rate of the tip velocity is linear with respect to the time step.

Figure 6.14. (a) The interfaces at $T = 1200$ for different time steps. (b) shows the velocity of the tip versus time. (c) The numerical experimental and linear fitting velocities versus time step.

6.2.2. Stability test. In this section, we perform a number of simulations on a set of increasingly finer grids to show that our proposed method is more stable than
the previous methods which suffer from time restrictions \( \Delta t \leq O(h^2) \) for stability. The computational domain is \( \Omega = (-200, 200)^2 \) and we take \( R_0 = 15d_0, \epsilon_6 = 0.02, \) and \( \Delta = -0.55. \) The numerical solutions are computed on the uniform grids \( h = 400/2^n \) with corresponding time steps \( \Delta t = 3h \) for \( n = 8, 9, \) and \( 10. \) Figure 6.15 shows the crystal growth with different time steps at \( T = 70.31. \) In general, large time steps may cause large truncation errors. However, as can be seen in Figure 6.15, we obtain stable solutions with large time steps.

\[ \text{Figure 6.15. The stability of crystal growth with different mesh sizes:} \]
\[ \begin{align*}
\text{(a) 256} \times \text{256 mesh (} \Delta t = 4.68 \text{),} & \quad \text{(b) 512} \times \text{512 mesh (} \Delta t = 2.34 \text{),} & \quad \text{(c) 1024} \times \text{1024 mesh (} \Delta t = 1.17 \text{).}
\end{align*} \]

\[ \text{6.2.3. Effect of } \epsilon_6. \] To investigate the effect of \( \epsilon_6, \) we consider the evolution of the interface with different \( \epsilon_6 = 0.002, 0.02, \) and \( 0.05. \) A \( 1024 \times 1024 \) mesh is used on the domain \( \Omega = (-100, 100)^2 \) and we take \( R_0 = 50d_0, \Delta = -0.55, \Delta t = 0.3, \) and \( T = 1200. \) Figures 6.16 (a), (b), and (c) are the evolution of crystal growth with \( \epsilon_6 = 0.002, 0.02, \) and \( 0.05, \) respectively. As advised in the previous paper, If \( \epsilon_6 < \frac{1}{33}, \) all of tangent planes lie outside and all orientations appear on the equilibrium shape. Detail view is drawn in Figure 6.16(a). Otherwise, there is missing orientations shown in Figure 6.16(c). While if \( \epsilon_6 \) is not more smaller than \( \frac{1}{33}, \) the crystal also works well shown in Figure 6.16(b). Thus the Wulff construction is not strictly correlated with \( \epsilon_6 \) in crystal growth, but provide guidelines for parameter selection.
6.2. NUMERICAL EXPERIMENTS FOR SNOW CRYSTAL GROWTH

Figure 6.16. The effect of $\epsilon_6$. (a), (b), and (c) are the evolution of crystal growth with $\epsilon_6 = 0.002$, 0.02, and 0.05, respectively. The times are $t = 0, 120, 240, 360, 480, 600, 720, 840, 960, 1080$, and 1200.

6.2.4. Effect of undercooling. Now we investigate the effects of undercooling of the initial solid seed. For each test, a $1024 \times 1024$ mesh is used on the domain $\Omega = (-200, 200)^2$ and we choose $R_0 = 15d_0$, $\epsilon_6 = 0.02$, $\Delta t = 0.3$, and $T = 1080$. Figure 6.17 shows sequences of interfaces with different undercooling sizes $\Delta = -0.45$, $\Delta = -0.55$, and $\Delta = -0.65$. We observe that the large initial undercooling causes the dendrite to grow faster.

Figure 6.17. Sequences of interfaces with different undercooling sizes $\Delta = -0.45$, $\Delta = -0.55$, and $\Delta = -0.65$.

6.2.5. $k$-fold symmetric crystal growth. If we set the energy function by $\epsilon(\phi) = \epsilon_0(1 + \epsilon_k \cos(k\phi))$, then our proposed method can simulate the $k$-fold crystal
growth in general. To show this, we simulate sequences of computational experiments of $k$-fold symmetric crystal growth for $k = 4, \ldots, 9$. A $1024 \times 1024$ mesh is used on the domain $\Omega = (-200, -200)^2$ and we take $R_0 = 15d_0$, $\Delta = -0.55$, and $\Delta t = 0.3$. Note that we use $\epsilon_k = 1/(k^2 - 1)$ to respond to the Wulff’s algorithm. The evolutions for each $k$ are shown in Figure 6.18.

![Images of crystal growth](image.png)

**Figure 6.18.** The evolutions of $k$-fold crystal growth after time: (a) $T = 720$, (b) $T = 1200$, (c) $T = 1680$, (d) $T = 2160$, (e) $T = 2520$, and (f) $T = 2880$. 
6.3. Numerical experiments for the multi-component Cahn–Hilliard system

6.3.1. The stability of the proposed scheme. We investigate the stability of five different schemes for the quaternary CH system: the explicit Euler’s (EE), semi-implicit Euler’s (SIE), implicit Euler’s (IE), Crank-Nicolson (CN), and nonlinear splitting (NS). For dissipative dynamics such as the CH system, a discrete time stepping algorithm is defined to be gradient stable if the free energy is nonincreasing, $\mathcal{F}(c^{n+1}) \leq \mathcal{F}(c^n)$, for each $n$. Define $\Delta t_{\text{max}}$ as the largest possible time step which allows stable numerical computation. In other words, if the time step is larger than $\Delta t_{\text{max}}$, then the algorithm is not gradient stable. To measure $\Delta t_{\text{max}}$ for each scheme, we perform a number of simulations for a sample initial problem on a set of increasingly finer grids. The initial conditions are

$$c_1(x, y, 0) = 0.25 + 0.1\text{rand}( ),$$
$$c_2(x, y, 0) = 0.25 + 0.1\text{rand}( ),$$
$$c_3(x, y, 0) = 0.25 + 0.1\text{rand}( )$$

on a domain $\Omega = (0, 1) \times (0, 1)$. Here, rand( ) is a random number between $-1$ and 1. The numerical solutions are computed on the uniform grids, $h = 1/2^n$ for $n = 5, 6, 7, \text{ and } 8$. For each case, $\epsilon = 0.64h$ is used. The values of $\Delta t_{\text{max}}$ with different schemes are listed in Table 6.11. From the results, we observe that EE, SIE, IE, and CN schemes are not gradient stable when we use the time step larger than $\Delta t_{\text{max}}$ (see Figure 6.19). However, our proposed scheme (NS) is gradient stable for time steps of any size, i.e., the scheme is practically unconditionally gradient stable.
6.3. NUMERICAL EXPERIMENTS FOR THE MULTI-COMPONENT CAHN–HILLIARD SYSTEM

Table 6.11. The values of the maximum time step guaranteeing the stability of each scheme.

<table>
<thead>
<tr>
<th>Case</th>
<th>32 × 32</th>
<th>64 × 64</th>
<th>128 × 128</th>
<th>256 × 256</th>
</tr>
</thead>
<tbody>
<tr>
<td>EE</td>
<td>7.6 × 10⁻⁵</td>
<td>1.9 × 10⁻³</td>
<td>4.7 × 10⁻⁶</td>
<td>1.1 × 10⁻⁶</td>
</tr>
<tr>
<td>SIE</td>
<td>2.9 × 10⁻²</td>
<td>7.9 × 10⁻₃</td>
<td>1.3 × 10⁻³</td>
<td>4.3 × 10⁻⁴</td>
</tr>
<tr>
<td>IE</td>
<td>3.2 × 10⁻²</td>
<td>7.8 × 10⁻₃</td>
<td>1.7 × 10⁻³</td>
<td>4.4 × 10⁻⁴</td>
</tr>
<tr>
<td>CN</td>
<td>2.0 × 10⁻²</td>
<td>5.0 × 10⁻₃</td>
<td>1.1 × 10⁻³</td>
<td>2.8 × 10⁻⁴</td>
</tr>
<tr>
<td>NS</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
</tr>
</tbody>
</table>

Figure 6.19. The maximum time step (Δt_{max}) guaranteeing the stability of each scheme.

6.3.2. Linear stability analysis. In this section, we study the short-time behavior of a quaternary mixture. The partial differential Equations (3.4) and (3.5) we wish to solve may be written as

\[
\frac{\partial c(x, t)}{\partial t} = \Delta \left( \psi(c) - \epsilon^2 \Delta c \right), \quad \text{for } (x, t) \in \Omega \times (0, T),
\]

(6.6)

where \( \psi(c) = f(c) + \beta(c)1 \). Let the mean concentration take the form \( m = (m_1, m_2, m_3) \). We seek a solution of the form

\[
c(x, t) = m + \sum_{k=1}^{\infty} \cos(k \pi x) (\alpha_k(t), \beta_k(t), \gamma_k(t)),
\]

(6.7)
6.3. NUMERICAL EXPERIMENTS FOR THE MULTI-COMPONENT CAHN–HILLIARD SYSTEM

where \(|\alpha_k(t)|, |\beta_k(t)|, \) and \(|\gamma_k(t)| \ll 1\). After linearizing \(\psi(\mathbf{c})\) about \(\mathbf{m}\), we have

\[
\psi(\mathbf{c}) \approx \psi(\mathbf{m}) + (\mathbf{c} - \mathbf{m}) \begin{pmatrix}
\partial_{\alpha_k} \psi_1(\mathbf{m}) & \partial_{\alpha_k} \psi_2(\mathbf{m}) & \partial_{\alpha_k} \psi_3(\mathbf{m}) \\
\partial_{\beta_k} \psi_1(\mathbf{m}) & \partial_{\beta_k} \psi_2(\mathbf{m}) & \partial_{\beta_k} \psi_3(\mathbf{m}) \\
\partial_{\gamma_k} \psi_1(\mathbf{m}) & \partial_{\gamma_k} \psi_2(\mathbf{m}) & \partial_{\gamma_k} \psi_3(\mathbf{m})
\end{pmatrix}.
\]

(6.8)

Substituting (6.8) into (6.6) and letting \(m_1 = m_2 = m_3 = m\) for simplicity, then, up to first order, we have

\[
\frac{\partial \mathbf{c}}{\partial t} = \Delta \mathbf{c} \begin{pmatrix}
\frac{18m^2 - 9m + 1}{3m^2(4m - 1)} & \frac{3m(4m - 1)}{3m^2(4m - 1)} & \frac{3m(4m - 1)}{18m^2 - 9m + 1} \\
\frac{3m^2(4m - 1)}{3m^2(4m - 1)} & \frac{18m^2 - 9m + 1}{3m^2(4m - 1)} & \frac{3m(4m - 1)}{18m^2 - 9m + 1} \\
\frac{3m(4m - 1)}{18m^2 - 9m + 1} & \frac{3m(4m - 1)}{18m^2 - 9m + 1} & \frac{18m^2 - 9m + 1}{3m^2(4m - 1)}
\end{pmatrix} - \epsilon^2 \Delta^2 \mathbf{c}.
\]

(6.9)

After substituting \(\mathbf{c}(x, t)\) from Equation (6.7) into (6.9), we get

\[
\begin{pmatrix}
\alpha_k(t) \\
\beta_k(t) \\
\gamma_k(t)
\end{pmatrix}' = \mathbf{A} \begin{pmatrix}
\alpha_k(t) \\
\beta_k(t) \\
\gamma_k(t)
\end{pmatrix}, \quad \mathbf{A} = \begin{pmatrix}
a & b & b \\
b & a & b \\
b & b & a
\end{pmatrix},
\]

(6.10)

where \('\) indicates the time derivative and

\[
a = \frac{-k^2\pi^2}{2}(18m^2 - 9m + 1) - \epsilon^2 k^4\pi^4, \quad b = \frac{-3k^2\pi^2m(4m - 1)}{2}.
\]

The eigenvalues of \(\mathbf{A}\) are

\[
\lambda_1 = \frac{-k^2\pi^2}{2}(42m^2 - 15m + 1 + 2\epsilon^2 k^2\pi^2),
\]

\[
\lambda_2 = \lambda_3 = \frac{-k^2\pi^2}{2}(6m^2 - 6m + 1 + 2\epsilon^2 k^2\pi^2).
\]

The solution to the system of ODEs (6.10) is given by

\[
\begin{pmatrix}
\alpha_k(t) \\
\beta_k(t) \\
\gamma_k(t)
\end{pmatrix} = \begin{pmatrix}
\alpha_k(0) + \beta_k(0) + \gamma_k(0) \\
\beta_k(0) + \gamma_k(0) \\
\gamma_k(0)
\end{pmatrix} \begin{pmatrix}
1 & 1 & e^{\lambda_1 t} \\
1 & 1 & e^{\lambda_2 t} + \frac{-\alpha_k(0) - \beta_k(0) + 2\gamma_k(0)}{3} \\
1 & 0 & e^{\lambda_3 t}
\end{pmatrix} \begin{pmatrix}
-1 \\
0 \\
1
\end{pmatrix}
\]

In Figure 6.20, we plot the evolution of the amplitudes as a function of time. The symbols ‘-o-’, ‘-c-’, and ‘-d-’ are numerical results that are compared with the theoretical values \(\alpha_k(t)\) (point), \(\beta_k(t)\) (star), and \(\gamma_k(t)\) (plus), respectively, with the
6.3. NUMERICAL EXPERIMENTS FOR THE MULTI-COMPONENT Cahn–Hilliard System

Figure 6.20. The symbols ‘-○-’, ‘-◆-’, and ‘-△-’ are numerical results that are compared with the theoretical values $\alpha_k(t)$ (point), $\beta_k(t)$ (star), and $\gamma_k(t)$ (plus), respectively, with the initial conditions of Equations (6.11)-(6.13).

initial conditions:

\[
\begin{align*}
    c_1(x, 0) &= 0.25 + 0.001 \cos(3\pi x), \\
    c_2(x, 0) &= 0.25 + 0.002 \cos(3\pi x), \\
    c_3(x, 0) &= 0.25 + 0.003 \cos(3\pi x).
\end{align*}
\]  

Here, we used $k = 3$, $m = 0.25$, $\epsilon = 0.005$, $h = 1/256$, $\Delta t = 0.1h$, and $T = 200\Delta t$. The numerical amplitudes are defined by

\[
\begin{align*}
    \alpha_k^n &= \left( \max_{1 \leq i \leq N_x} c_1^n(x_i) - \min_{1 \leq i \leq N_x} c_1^n(x_i) \right) / 2, \\
    \beta_k^n &= \left( \max_{1 \leq i \leq N_x} c_2^n(x_i) - \min_{1 \leq i \leq N_x} c_2^n(x_i) \right) / 2, \\
    \gamma_k^n &= \left( \max_{1 \leq i \leq N_x} c_3^n(x_i) - \min_{1 \leq i \leq N_x} c_3^n(x_i) \right) / 2.
\end{align*}
\]
6.3. NUMERICAL EXPERIMENTS FOR THE MULTI-COMPONENT CAHN–HILLIARD SYSTEM

The results in Figure 6.20 show that the linear stability analysis and numerical solutions are in good agreement in a linear regime.

6.3.3. The efficiency of the proposed scheme. As mentioned in Section 5.5.1, we can solve the $N$-component CH system in a decoupled way by using our scheme. In order to show the efficiency of the proposed scheme, we consider phase separation of $N = 3, 4, \ldots, 10$ components in the unit square domain $\Omega = (0, 1) \times (0, 1)$. For each number of components, the initial condition is a randomly chosen superposition of circles. We choose $h = 1/128$, $\Delta t = 10h$, and $\epsilon = 0.0047$ and perform 4000 time steps. The evolution of the interface is shown in Figure 6.27. Rows 1 and 2 correspond to $t = 10\Delta t$ and $4000\Delta t$, respectively. Table 6.13 provides the average CPU time (in seconds) during 4000 time steps for each number of components. The average CPU time versus number of components is shown in Figure 6.28. The results suggest that the convergence rate of average CPU time is linear with respect to number of components.

Table 6.12. Average CPU times (sec.) for different numbers of components.

<table>
<thead>
<tr>
<th>$N$</th>
<th>Average CPU time</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>2.060</td>
</tr>
<tr>
<td>4</td>
<td>2.990</td>
</tr>
<tr>
<td>5</td>
<td>4.070</td>
</tr>
<tr>
<td>6</td>
<td>5.028</td>
</tr>
<tr>
<td>7</td>
<td>5.791</td>
</tr>
<tr>
<td>8</td>
<td>6.686</td>
</tr>
<tr>
<td>9</td>
<td>7.492</td>
</tr>
<tr>
<td>10</td>
<td>8.533</td>
</tr>
</tbody>
</table>

6.3.4. Spinodal decomposition - phase separation of a ten-component mixture. We consider phase separation of a ten-component mixture by spinodal decomposition. The initial condition is a randomly chosen superposition of circles. A $128 \times 128$ mesh is used on the domain $\Omega = (0, 1) \times (0, 1)$ and we take $\Delta t = 10h$ and $\epsilon = 0.0038$. We compute until the solution becomes numerically stationary. Figure 6.23 shows the evolution of the interface at different times. We observe that three phases meet at one point and the angles between them approach $120^\circ$ as they approach local
6.3. NUMERICAL EXPERIMENTS FOR THE MULTI-COMPONENT CAHN–HILLIARD SYSTEM

Figure 6.21. Phase separation of $N = 3, 5, 8,$ and 10 components. Rows 1 and 2 correspond to $t = 10\Delta t$ and $4000\Delta t$, respectively. Numbers in row 2 indicate the number of components.

Figure 6.22. Average CPU time versus number of components.

equilibrium states. This is due to the fact that in the total energy functional equation (3.2), $F(c(x,t))$ is symmetric and the interaction parameter $\epsilon$ is constant. This result is in good agreement with the theory in [205].
6.4. Numerical experiments for the multi-component Navier–Stokes–Cahn–Hilliard system

6.4.1. Phase separation of a five-component mixture in a gravitational field. We consider multi-component incompressible viscous fluid flow. The fluid dynamics is described by the Navier–Stokes–Cahn–Hilliard (NSCH) equations [64, 63, 153, 11, 154, 155, 156, 65, 187, 189, 58, 55, 16, 56]:

\[ \rho(c) \left( \frac{\partial u}{\partial t} + u \cdot \nabla u \right) = -\nabla p + \eta \Delta u + \rho(c)g, \]  

(6.14)

\[ \nabla \cdot u = 0, \]  

(6.15)

\[ \frac{\partial c}{\partial t} + u \cdot \nabla c = M \Delta \mu, \]  

(6.16)

\[ \mu = f(c) - \epsilon^2 \Delta c + \beta(c), \]  

(6.17)
where $u$ is the velocity, $p$ is the pressure, $\rho(c)$ is the density, $\eta$ is the viscosity, $g = (0, -g)$ is the gravity, $M$ is the mobility, and $\mu$ is the generalized chemical potential. In this paper the effect of the surface tension is negligible. We note that even though our phase-field model can deal with a variable viscosity case straightforwardly, we focus on the viscosity matched case.

To make Equations (6.14)-(6.17) dimensionless, we choose the following definitions:

$$x' = \frac{x}{L_c}, \quad u' = \frac{u}{U_c}, \quad t' = \frac{tU_c}{L_c}, \quad \rho' = \frac{\rho}{\rho_c}, \quad p' = \frac{p}{\rho_c U_c^2}, \quad g' = \frac{g}{g}, \quad \mu' = \frac{\mu}{\mu_c},$$

where the primed quantities are dimensionless and $L_c$ is the characteristic length, $U_c$ is the characteristic velocity, $\rho_c$ is the characteristic density and is defined as that of fluid $1$, $g$ is the gravitational acceleration, and $\mu_c$ is the characteristic chemical potential. Substituting these variables into Equations (6.14)-(6.17) and dropping the primes, we have

$$\rho(c) \left( \frac{\partial u}{\partial t} + u \cdot \nabla u \right) = -\nabla p + \frac{1}{Re} \Delta u + \frac{\rho(c)}{Fr^2} g, \quad (6.18)$$

$$\nabla \cdot u = 0, \quad (6.19)$$

$$\frac{\partial c}{\partial t} + u \cdot \nabla c = \frac{1}{Pe} \Delta \mu, \quad (6.20)$$

$$\mu = f(c) - \epsilon^2 \Delta c + \beta(c), \quad (6.21)$$

where $g = (0, -1)$ and $\epsilon$ is redefined according to the scaling. The dimensionless parameters are the Reynolds number, $Re = \rho_c U_c L_c/\eta$, Froude number, $Fr = U_c/\sqrt{gL_c}$, and Peclet number, $Pe = U_c L_c/(M \mu_c)$. Using $U_c = \sqrt{gL_c}$, we have $Re = \rho_c U_c L_c/\eta = \rho_c g^{1/2} L_c^{3/2}/\eta$ and $Fr = U_c/\sqrt{gL_c} = \sqrt{gL_c}/\sqrt{gL_c} = 1$. By applying our scheme, we can solve the multi-component advective CH system (6.20) and (6.21) in a decoupled way and solving the multi-component NSCH system (6.18)-(6.21) becomes solving the
6.4. NUMERICAL EXPERIMENTS FOR THE MULTI-COMPONENT NAVIER-STOKES-CAHN-HILLIARD SYSTEM

For a detailed description of the numerical method used in solving the binary NSCH system, please refer to Ref. [156].

To model phase separation of a five-component mixture in a gravitational field, we take an initial velocity field as zero, $u = 0$, and the initial conditions for $c$ are randomly distributed between 0 and 1. $\rho(c) = \sum_{i=1}^{5} \rho_i c_i$ ($c_5 = 1 - c_1 - c_2 - c_3 - c_4$ and $\rho_i$ is the $i$th fluid density) and $\rho_i = 6 - i$ for $i = 1, \ldots, 5$. A mesh size 128 $\times$ 128 is used on the unit square domain and we choose $\Delta t = 2.0 \times 10^{-3}$, $\epsilon = 0.0047$, $Re = 3000$, and $Pe = 0.1/\epsilon$. Figure 6.24 shows the time evolution of the five-component mixture system in a gravitational field. Fluid 1 is represented by the black region, fluid 2 by the dark gray region, fluid 3 by the gray region, fluid 4 by the light gray region, and fluid 5 by the white region. From Figure 6.24 we see that the gravity affects a multi-component simulation by pulling the heavy fluid to the bottom of the computational domain.

Figure 6.24. Phase separation of a five-component mixture in a gravitational field. Fluid 1 is represented by the black region, fluid 2 by the dark gray region, fluid 3 by the gray region, fluid 4 by the light gray region, and fluid 5 by the white region. Times are shown below each figure.

6.4.2. The Rayleigh-Taylor instability of a five-component mixture. When a heavy fluid is superposed over a light fluid in a gravitational field, the fluid interface is unstable. Any perturbation of this interface tends to grow with time, producing the
phenomena known as the Rayleigh-Taylor instability. The phenomena are the penetration of both heavy and light fluids into each other. The Rayleigh-Taylor instability for a fluid in a gravitational field was originally introduced by Rayleigh [206] and later applied to all accelerated fluids by Taylor [207].

In this section, we study the Rayleigh-Taylor instability of a five-component mixture. In the simulations, we have two initial states as shown in Figure 6.25 (a) and Figure 6.26 (a). The initial velocity is zero. \( \rho(c) = \sum_{i=1}^{5} \rho_i c_i \) \( (c_5 = 1 - c_1 - c_2 - c_3 - c_4 \) and \( \rho_i \) is the \( i \)th fluid density) and \( \rho_i = 6 - i \) for \( i = 1, \ldots, 5 \). A mesh size \( 128 \times 512 \) is used on a domain \( \Omega = (0, 1) \times (0, 4) \) and we choose \( \Delta t = 2.0 \times 10^{-3} \), \( \epsilon = 0.0047 \), \( Re = 3000 \), and \( Pe = 0.1/\epsilon \). The results are presented in Figures 6.25 (b)-(g) and Figures 6.26 (b)-(g). The area shown by black indicates the fluid 1 region, while the dark gray, gray, light gray, and white color regions stand for the fluid 2, 3, 4, and 5 domains, respectively. We observe that our proposed method is a powerful tool to simulate the Rayleigh-Taylor instability between multi-component fluids.

**Figure 6.25.** The Rayleigh-Taylor instability of a five-component mixture. Times are \( t = 0, 3, 6, 10, 20, 40, \) and 160 (left to right).
Figure 6.26. The Rayleigh-Taylor instability of a five-component mixture. Times are \( t = 0, 3, 6, 10, 30, 50, \) and 200 (left to right).
6.5. Numerical experiments for the multi-component Allen–Cahn system

6.5.1. The efficiency of the proposed scheme. As mentioned in Section 5.6, we can solve the $N$-component AC system in a decoupled way by using our scheme. In order to show the efficiency of the proposed scheme, we consider phase separation of $N = 3, 4, \ldots, 10$ components in the unit square domain $\Omega = (0, 1) \times (0, 1)$. For each number of components, the initial condition is a randomly chosen superposition of circles. We choose $h = 1/128$, $\Delta t = 10h$, and $\epsilon = 0.0047$ and perform 30000 time steps. The evolution of the interface is shown in Figure 6.27. Rows 1 and 2 correspond to $t = 3000\Delta t$ and $30000\Delta t$, respectively. Table 6.13 provides the average CPU time (in seconds) during 30000 time steps for each number of components. The average CPU time versus number of components is shown in Figure 6.28. The results suggest that the convergence rate of average CPU time is linear with respect to number of components.

<table>
<thead>
<tr>
<th>$N$</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average CPU time</td>
<td>0.0498</td>
<td>0.0742</td>
<td>0.0998</td>
<td>0.1254</td>
<td>0.1526</td>
<td>0.1779</td>
<td>0.2056</td>
<td>0.2329</td>
</tr>
</tbody>
</table>

Table 6.13. Average CPU times (sec.) for different numbers of components.
6.5. NUMERICAL EXPERIMENTS FOR THE MULTI-COMPONENT ALLEN–CAHN SYSTEM

**Figure 6.27.** Phase separation of $N = 3, 5,$ and 8 components. Rows 1 and 2 correspond to $t = 3000\Delta t$ and $30000\Delta t$, respectively.

**Figure 6.28.** Average CPU time versus number of components.
Chapter 7

Conclusion

To model multi-component fluid flows, we considered the vector valued phase-field, velocity, and pressure which are governed by the $N$-component advective Cahn–Hilliard and modified Navier–Stokes equations. For the $N$-component Cahn–Hilliard equation, we presented a practically unconditionally gradient stable conservative nonlinear numerical scheme. The scheme was based on a nonlinear splitting method and was solved by an efficient and accurate nonlinear multigrid method. And the scheme allows us to convert the $N$-component Cahn–Hilliard equation into a system of $N-1$ binary Cahn–Hilliard equations and significantly reduces the required computer memory and CPU time. The Cahn–Hilliard–Navier–Stokes equations can be applied to any number of fluid components. And, to investigate the buoyancy driven interpenetration of fluids with different densities, we implemented a time-dependent pressure boundary condition through a time-dependent density field at the boundary. Due to the pressure boundary treatment, we can perform long time evolutions resulting in an equilibrium state.

We presented an unconditionally stable second-order hybrid numerical method for solving the Allen–Cahn equation. The proposed method was based on operator splitting techniques. The Allen–Cahn equation was divided into a linear and a nonlinear equation. First, the linear equation was discretized using an implicit Euler’s scheme and the resulting discrete system of equations was solved by a multigrid method. The
nonlinear equation was then solved analytically due to the availability of a closed-form solution. In particular, we applied this technique to dendritic growth simulation. A great challenge in the simulation with various supercoolings is the large difference in time and length scales. This introduces a severe time step restriction for stability. The proposed scheme allows the use of a sufficiently large time step without the technical limitations.
Appendix

In this Appendix, we present a source code written in C language for the Cahn-Hilliard equation.

```c
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#define nx 128
#define ny 128
#define iloop for(i=1;i<=nx;i++)
#define jloop for(j=1;j<=ny;j++)
#define ijloop int i, j; iloop jloop
#define iloopt for(i=1;i<=nxt;i++)
#define jloopt for(j=1;j<=nyt;j++)
#define ijloopt int i, j; iloopt jloopt
void initialization(double **cn);
void Cahn_Hilliard(double **cn, double **cnp);
void source(double **cn, double **src_c, double **src_mu);
void vcycle(double **cnp, double **mu, double **sor_c, double **sor_mu,
            int nxf, int nyf, int ilevel);
void relax(double **cnp, double **mu, double **sor_c, double **sor_mu,
          int ilevel, int nxt, int nyt);
void defect(double **def_c, double **def_mu, double **cn, double **cnp,
            double **sor_c, double **sor_mu, int nxf, int nyt);
void LS(double **LSc, double **LSmu, double **cnp, double **mu,
        int nxf, int nyt);
void laplace(double **a, double **lap_a, int nxt, int nyt);
void restrict(double **cf, double **cc, double **muf, double **muc,
              int nxt, int nyt);
void prolong(double **cc, double **cf, double **muc, double **muf,
             int nxt, int nyt);
double **dmatrix(long nrl, long nrh, long ncl, long nch);
void zero_matrix(double **a, int nxt, int nyt);
void free_dmatrix(double **m, long nrl, long nrh, long ncl, long nch);
void mat_add(double **a, double **b, double **c, int nxt, int nyt);
void mat_sub(double **a, double **b, double **c, int nxt, int nyt);
void mat_copy(double **a, double **b, int nxt, int nyt);
double error(double **ct, double **cnp);
```
void print_data(double **cnp);
int n_level, c_relax;
double **tmp1, **tmp2, **ct, **mu, **sor_c, **sor_mu, xleft, xright,
yleft, yright, h, h2, dt, epsilon, Cahn;
int main()
{
    extern int n_level, c_relax;
    extern double **tmp1, **tmp2, **ct, **mu, **sor_c, **sor_mu,
xleft, xright, yleft, yright, h, h2, dt, epsilon, Cahn;
    int it, max_it, print_interval, count=1;
    double **cn, **cnp;
    FILE *fphi;
    if (nx<ny) n_level=(int)(log(nx)/log(2)+0.1);
    else n_level=(int)(log(ny)/log(2)+0.1);
    c_relax=5, max_it=50, print_interval=max_it/2,
xleft=0.0, xright=1.0, yleft=0.0, yright=1.0;
    h=xright/(double)nx, h2=pow(h,2), dt=h, epsilon=h,
    Cahn=pow(epsilon,2);
    tmp1=dmatrix(1, nx, 1, ny); tmp2=dmatrix(1, nx, 1, ny);
    ct=dmatrix(1, nx, 1, ny); mu=dmatrix(1, nx, 1, ny);
    sor_c=dmatrix(1, nx, 1, ny); sor_mu=dmatrix(1, nx, 1, ny);
    cn=dmatrix(1, nx, 1, ny); cnp=dmatrix(1, nx, 1, ny);
    initialization(cnp);
    fphi=fopen("phi.m","w");
    fprintf(fphi,"A=[ 
    fclose(fphi);
    print_data(cnp);
    for (it=1; it<=max_it; it++) {
        mat_copy(cn, cnp, nx, ny);
        Cahn_Hilliard(cn, cnp);
        if (it % print_interval==0) {
            print_data(cnp);
            printf("print out counts %d \n", count++);
        }
        printf("Iteration is %d \n", it);
    }
    fphi = fopen("phi.m","a");
    fprintf(fphi,"]; surf(A'); shading interp \n");
    fclose(fphi);
    return 0;
}
void initialization(double **cn)
{
    ijloop {
cn[i][j]=0.3*(0.5-rand()/(double)RAND_MAX);
mu[i][j]=0.0; }
void Cahn_Hilliard(double **cn, double **cnp)
{
    int max_it=300, iter=1;
double tol=1.0e-5, resid=1.0;
    source(cn, sor_c, sor_mu);
    mat_copy(ct, cn, nx, ny);
    while (iter <= max_it && resid > tol) {
        vcycle(cnp, mu, sor_c, sor_mu, nx, ny, 1);
        resid=error(ct, cnp);
        mat_copy(ct, cnp, nx, ny);
        iter++; }
    printf("Error is %12.10f %d",resid,iter-1);
}
void source(double **cn, double **src_c, double **src_mu)
{
    ijloop {
        src_c[i][j]=cn[i][j]/dt;
        src_mu[i][j]=pow(cn[i][j],3)-3.0*cn[i][j];}
}
void vcycle(double **cnp, double **mu, double **sor_c, double **sor_mu,
            int nxf, int nyf, int ilevel)
{
    relax(cnp, mu, sor_c, sor_mu, ilevel, nxf, nyf);
    if (ilevel<n_level) {
        double **def_c, **def_mu, **codef_c, **codef_mu,
             **fidef_c, **fidef_mu;
        def_c=dmatrix(1, nxf/2, 1, nyf/2);
        def_mu=dmatrix(1, nxf/2, 1, nyf/2);
        fidef_c=dmatrix(1, nxf, 1, nyf);
        fidef_mu=dmatrix(1, nxf, 1, nyf);
        codef_c=dmatrix(1, nxf/2, 1, nyf/2);
        codef_mu=dmatrix(1, nxf/2, 1, nyf/2);
        defect(def_c, def_mu, cnp, mu, sor_c, sor_mu, nx, nyf);
        zero_matrix(codef_c, nx, nyf/2);
        vcycle(codef_c, codef_mu, def_c, def_mu, nx, nyf/2, ilevel+1);
        prolong(codef_c, fidef_c, codef_mu, fidef_mu, nx, nyf/2);
        mat_add(cnp, cnp, fidef_c, nx, nyf);
        mat_add(mu, mu, fidef_mu, nx, nyf);
        relax(cnp, mu, sor_c, sor_mu, ilevel, nx, nyf);
        free_dmatrix(def_c, 1, nxf/2, 1, nyf/2);
void restrict(double **cf, double **cc, double **muf, double **muc,  
              int nxt, int nyt)
{
  ijloopt {
    cc[i][j]=0.25*(cf[2*i][2*j]+cf[2*i-1][2*j]
         +cf[2*i][2*j-1]+cf[2*i-1][2*j-1]);
    muc[i][j]=0.25*(muf[2*i][2*j]+muf[2*i-1][2*j]
         +muf[2*i][2*j-1]+muf[2*i-1][2*j-1]);
  }
}
void prolong(double **cc, double **cf, double **muc, double **muf,  
              int nxt, int nyt)
{
  ijloopt {
    cf[2*i][2*j]=cf[2*i-1][2*j]
    =cf[2*i][2*j-1]=cf[2*i-1][2*j-1]=cc[i][j];
    muc[2*i][2*j]=muf[2*i-1][2*j]
    =muf[2*i][2*j-1]=muf[2*i-1][2*j-1]=muc[i][j];
  }
}
void LS(double **LSc, double **LSmu, double **cnp, double **mu,  
         int nxt, int nyt)
{
  double **lap_mu, **lap_c;
  lap_mu=dmatrix(1, nxt, 1, nyt);
  lap_c=dmatrix(1, nxt, 1, nyt);
  laplace(cnp, lap_c, nxt, nyt);
  laplace(mu, lap_mu, nxt, nyt);
  ijloopt {
    LSc[i][j]=cnp[i][j]/dt-lap_mu[i][j];
    LSmu[i][j]= - 2.0*cnp[i][j]+Cahn*lap_c[i][j]+mu[i][j];
  }
  free_dmatrix(lap_mu, 1, nxt, 1, nyt);
  free_dmatrix(lap_c, 1, nxt, 1, nyt);
}
void defect(double **def_c, double **def_mu, double **cn, double **cnp,  
            double **sor_c, double **sor_mu, int nxf, int nyf)
{
  LS(tmp1, tmp2, cn, cnp, nxf, nyf);
  mat_sub(tmp1, sor_c, tmp1, nxf, nyf);
  mat_sub(tmp2, sor_mu, tmp2, nxf, nyf);
restrict(tmp1, def_c, tmp2, def_mu, nxf/2, nyf/2);
}
void error(double **ct, double **cnp)
{
    double value=0.0;
    ijloop {
      if (fabs(ct[i][j]-cnp[i][j]) > value)
        value=fabs(ct[i][j]-cnp[i][j]);
    }
    return value;
}
void relax(double **cnp, double **mu, double **sor_c, double **sor_mu,
          int ilevel, int nxt, int nyt)
{
    int iter;
    double ht2, a[4], f[2], det;
    ht2 = pow(xright/(double) nxt,2);
    for (iter=1; iter<=c_relax; iter++) {
        ijloopt {
            a[0]=1.0/dt, a[1]=0.0, a[2]=-2.0, a[3]=1.0;
            f[0]=sor_c[i][j], f[1]=sor_mu[i][j];
            if (i>1) {a[1]+=1.0/ht2,a[2]=-Cahn/ht2;
                f[0]+=mu[i-1][j]/ht2,f[1]-=Cahn*cnp[i-1][j]/ht2;}
            if (i<nxt) {a[1]+=1.0/ht2,a[2]=-Cahn/ht2;
                f[0]+=mu[i+1][j]/ht2,f[1]-=Cahn*cnp[i+1][j]/ht2;}
            if (j>1) {a[1]+=1.0/ht2,a[2]=-Cahn/ht2;
                f[0]+=mu[i][j-1]/ht2,f[1]-=Cahn*cnp[i][j-1]/ht2;}
            if (j<nyt) {a[1]+=1.0/ht2,a[2]=-Cahn/ht2;
                f[0]+=mu[i][j+1]/ht2,f[1]-=Cahn*cnp[i][j+1]/ht2;}
            det = a[0]*a[3] - a[1]*a[2];
            cnp[i][j] = (a[3]*f[0] - a[1]*f[1])/det;
            mu[i][j] = (-a[2]*f[0] + a[0]*f[1])/det;
        }
    }
void laplace(double **a, double **lap_a, int nxt, int nyt)
{
    double ht2, value;
    ht2 = pow(xright / (double) nxt, 2);
    ijloopt {
        value=0.0;
        if (i<nxt) value += a[i+1][j]-a[i][j];
        if (i>1) value -= a[i][j]-a[i-1][j];
        if (j<nyt) value += a[i][j+1]-a[i][j];
        if (j>1) value -= a[i][j]-a[i][j-1];
        lap_a[i][j]=value/ht2;
```c
} void mat_add(double **a, double **b, double **c, int nxt, int nyt)
{   ijloopt a[i][j] = b[i][j]+c[i][j];
} void zero_matrix(double **a, int nxt, int nyt)
{   ijloopt a[i][j]=0.0;
} void mat_copy(double **a, double **b, int nxt, int nyt)
{   ijloopt a[i][j]=b[i][j];
} double **dmatrix(long nrl, long nrh, long ncl, long nch)
{   double ***m;
   long i, nrow=nrh-nrl+1+1, ncol=nch-ncl+1+1;
   m=(double **) malloc((nrow)*sizeof(double*));
   m+=1;
   m-=nrl;
   m[nrl]=(double *) malloc((nrow*ncol)*sizeof(double));
   m[nrl]+=1;
   m[nrl]-=ncl;
   for (i=nrl+1; i<=nrh; i++) m[i]=m[i-1]+ncol;
   return m;
} void free_dmatrix(double **m, long nrl, long nrh, long ncl, long nch)
{   free(m[nrl]+ncl-1);
   free(m+nrl-1);
} void mat_sub(double **a, double **b, double **c, int nxt, int nyt)
{   ijloopt a[i][j]=b[i][j]-c[i][j];
} void print_data(double **cnp)
{   int i, j;
   FILE *fp;
   fp = fopen("phi.m","a");
   iloop {
     jloop { fprintf(fp, " %f", cnp[i][j]);
         fprintf(fp, "\n");}
   fclose(fp);
}
Bibliography


[216] T. Ohtsuka, Motion of interfaces by an Allen–Cahn type equation with multiple-well potentials, Asymptotic Anal. 56 (2008) 87-123.

References


