Hybrid phase-field and immersed boundary methods

By

Yibao Li

DISSERTATION

Submitted in partial satisfaction of the requirements for the degree of

Master of Science

in

Mathematics

in the

OFFICE OF GRADUATE STUDIES

of the

KOREA UNIVERSITY

Approved:

_____

Committee Member 1

_____

Committee Member 2

_____

Committee Member 3

Committee in Charge

2010

Contents

Abstract

This thesis describes various numerical methods for hybrid phase-field and immersed boundary methods. Hybrid method is a useful and powerful method in numerical computation. In the first part of this thesis, we present an unconditionally stable second-order hybrid numerical method for solving the Allen-Cahn equation representing a model for antiphase domain coarsening in a binary mixture. Then, we apply this hybrid method to the application of binary image segmentation, geometric image segmentation, multiphase image segmentation, and the simulation of crystal growth. In the second part, we consider the simulations of thin film based on Navier-Stokes flows by implicit ENO (essentially non-oscillatory) type scheme which has a good stability property. Secondly, an immersed boundary method (IBM) for two-phase fluid flows is considered. Since the interface between two fluids is moved in a discrete manner, this can result in a lack of volume conservation. We propose a volume correction scheme. The idea of area preserving correction scheme is to correct the interface location normally to the interface so that the area remains constant. Finally, we consider the cytokinesis of an animal cell using the IBM.

Acknowledgments

I am very grateful for my advisor, Prof. Junseok Kim, for his guidance, encouragement and assistance to my professional development. My appreciation also goes to Prof. Woonjae Hwang and Prof. Donggyun Kim for serving on my committee. I would also like to thank my colleagues and friends for their help and friendship in our research group, In particular, I thank Hyun Geun Lee, Darea Jeong, and Prof. Junseok Kim for making the research group pleasant and very productive environment for research. In addition, I would like to thank my parents and grandparents for their support and love. Finally, I want to thank my wife, Binhu Xia, for her love and understanding.

Chapter 1

Introduction

1.1. Motivation and Objectives

This dissertation consists of published and working papers

1. An unconditionally stable hybrid numerical method for solving the Allen-Cahn equation... Yibao Li, Hyun Geun Lee, Darae Jeong, and Junseok Kim, Computers and Mathematics with Applications, Vol. 60, No. 6, pp. 1591–1606, 2010.
2. Numerical studies of the fingering phenomena for the thin film equation ... Yibao Li, Hyun Geun Lee, Woonjae Hwang, Daeki Yoon, Suyeon Shin, Youngsoo Ha, and Junseok Kim, in press, International Journal for Numerical Methods in Fluids, 2010
3. A fast and accurate numerical method for medical image segmentation ... Yibao Li and Junseok Kim, J. KSIAM Vol 14, No. 4, pp. 201–210, 2010

1.2. Outline of the thesis

In Chapter 2, we present an unconditionally stable second-order hybrid numerical method for solving the Allen-Cahn equation representing a model for antiphase domain coarsening in a binary mixture. 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 a Crank-Nicolson scheme and the resulting discrete system of equations was solved by a fast solver such as a multigrid method. The nonlinear equation was then solved analytically due to the availability of a closed-form solution. Various numerical experiments are presented to confirm the accuracy, efficiency, and stability of the proposed method. In particular, we show that the scheme is unconditionally stable and second-order accurate in both time and space.

In Chapter 3, we propose a new unconditionally stable hybrid numerical method for minimizing the piecewise constant Mumford-Shah functional of image segmentation. The model is based on the Allen-Cahn equation and an operator splitting technique is used to solve the model numerically. We split its numerical solution algorithm into two linear equations and one nonlinear equation. One of the linear equations and the nonlinear equation are solved analytically due to the availability of closed-form solutions. The other linear equation is discretized using an implicit scheme and the resulting discrete system of equations is solved by a fast numerical method such as a multigrid method. We analyze and prove the unconditional stability of the scheme. Various numerical results on real and synthetic images with noises are presented to demonstrate the efficiency, robustness, and accuracy of the proposed method.

In Chapter 4, We propose a new robust and accurate method for the numerical solution of medical image segmentation. The modified Allen-Cahn equation is used to model the boundaries of the image regions. Its numerical algorithm is based on operator splitting techniques.

In the first step in the splitting scheme, we implicitly solve the heat equation with the variable diffusive coefficient and a source term. Then, in the second step, using a closed-form solution for the nonlinear equation, we get an analytic solution. We overcome the time step constraint associated with most numerical implementations of geometric active contours. We demonstrate performance of the proposed image segmentation algorithm on several artificial as well as real image examples.

In Chapter 5, we present a new unconditionally stable hybrid numerical method for minimizing the piecewise constant Mumford-Shah functional of image segmentation with a novel multiphase segmentation model. The proposed model outputs a single multiphase distribution from which each individual segment or phase can be easily extracted. Theoretical analysis is developed for the -convergence behavior of the proposed model and the existence of its minimizers. Since the model is neither quadratic nor convex, for computation we adopted the convex-concave procedure that has been developed in the literatures of both computational nonlinear PDEs and neural computation. Numerical details and experiments on both synthetic and natural images are presented.

In Chapter 6, we propose a fast, robust, and accurate operator splitting method for the crystal growth phase-field simulation of binary alloy solidification in both two- and three-dimensional space. The proposed method is based on operator splitting techniques. We split the governing phase-field equation into three parts. The first equation is calculated explicitly. The second one is a heat equation with source term and is solved by a fast solver such as a multigrid method. The third one is evaluated using a closed form solution. We also present a set of representative numerical experiments for crystal simulation to demonstrate the accuracy of the proposed method. Our simulation results are also consistent with previous numerical experiments.

We present in Chapter 7, a new interpretation of the fingering phenomena of the thin liquid film layer through numerical experiments. The governing partial differential equation is $h_t + (h^2 - h^3)_x = -\epsilon^3 \nabla \cdot (h^3 \nabla \Delta h)$, which arises in the context of thin liquid films driven by a thermal gradient with a counteracting gravitational force, where $h = h(x, y, t)$ is the liquid film height. A robust and accurate finite difference method is developed for the thin liquid film equations. For the advection part $(h^2 - h^3)_x$, we use an implicit ENO (essentially non-oscillatory) type scheme and get a good stability property. The resulting nonlinear discrete system is solved by an efficient nonlinear multigrid method. Our numerical experiments indicate that higher the film thickness, the faster the film front evolves. The concave front has higher film thickness than the convex front. Combined these two effects cause the fingering phenomena.

In Chapter 8, we propose a simple area preserving correction scheme for the two-phase immiscible incompressible Navier-Stokes flows with an immersed boundary method (IBM). The IBM was originally developed to model blood flow in the heart and has been widely applied to biofluid dynamics problems with complex geometries and immersed elastic membranes. The main idea of IBM is to use a regular Eulerian computational grid for the fluid mechanics together with a Lagrangian representation of the immersed boundary. Using the discrete Dirac delta function and the indicator function, we can include surface tension force, variable viscosity and mass density, and gravitational force effects. The principal advantage of IBM for two-phase fluid flows is its inherent accuracy due in part to the ability to use a large number of

the interfacial marker points on the interface. However, because the interface between two fluids is moved in a discrete manner, this can result in a lack of volume conservation. The idea of area preserving correction scheme is to correct the interface location normally to the interface so that the area remains constant. Various numerical experiments are presented to illustrate the efficiency and accuracy of the proposed conservative IBM for two-phase fluid flows.

In Chapter 9, cytokinesis is the process of cell division which has been extensively studied. The site of cleavage and the functional response of cytoplasm have been controversial issues. We present more realistic model for the site of cleavage using the aster relaxation theory. We use an immersed boundary model of the axisymmetric eukaryotic cell division.

Finally, conclusions are drawn in Chapter 10.

Chapter 2

An unconditionally stable hybrid numerical method for the Allen-Cahn equation

2.1. Introduction

This Chapter presents an unconditionally stable second-order hybrid numerical method for the Allen-Cahn (AC) equation

$$\frac{\partial c(\mathbf{x},t)}{\partial t} = -M\left(\frac{F'(c(\mathbf{x},t))}{\epsilon^2} - \Delta c(\mathbf{x},t)\right),\ \mathbf{x}\in\Omega,\ 0<t\le T, \tag{2.1}$$

where $\Omega \subset \mathbf{R}^d$ $(d = 1, 2, 3)$ is a domain [1]. The quantity $c(\mathbf{x}, t)$ is defined as the difference between the concentrations of the two components in a mixture. The coefficient, M, is a constant mobility and $M \equiv 1$ is taken for convenience. The function, $F(c) = 0.25(c^2 - 1)^2$, is the Helmholtz free-energy density, as shown in Fig. 2.1 [113]. The small positive constant ϵ is the gradient energy coefficient related to the interfacial energy.

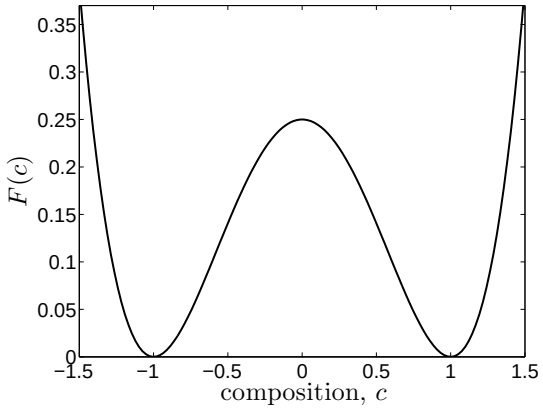


FIGURE 2.1. A double well potential, $F(c) = 0.25(c^2 - 1)^2$.

The boundary condition is

$$\frac{\partial c}{\partial \mathbf{n}} = 0 \text{ on } \partial\Omega, \tag{2.2}$$

where $\frac{\partial}{\partial \mathbf{n}}$ denotes the normal derivative on $\partial\Omega$. Let us define the Ginzburg-Landau free energy,

$$\mathcal{E}(c) := \int_\Omega \left(\frac{F(c)}{\epsilon^2} + \frac{|\nabla c|^2}{2}\right) d\mathbf{x}. \tag{2.3}$$

The AC equation can then be derived as the L^2-gradient flow of the total free energy, $\mathcal{E}(c)$ [29]. The AC equation was introduced originally as a phenomenological model for antiphase domain coarsening in a binary alloy [1]. The AC equation and its various modified forms have been applied to a range of problems, such as phase transitions [1], image analysis [7, 41], motion by mean curvature [15, 49, 52, 74, 87, 122], two-phase fluid flows [164], and crystal growth [38, 158]. Therefore, an efficient and accurate numerical solution of this equation is needed to better understand its dynamics.

In [7], the authors proposed the two-level semi-implicit finite-difference scheme. They observed that their scheme is dominated by the second-order difference operator. In [23], the authors proposed an unconditionally gradient stable scheme, which means that the discrete energy of Eq. (2.3) is decreasing regardless of time step sizes. Decreasing discrete energy implies the pointwise boundedness of the numerical solution for the AC equation. They showed that their scheme is indeed second-order accurate in space and first-order accurate in time. In [55], the authors proposed an adaptive finite element approximations of the AC equation. In [56], the authors proposed a new numerical scheme that combines the adaptive moving mesh method with the semi-implicit Fourier spectral algorithm. By maintaining a similar accuracy, the proposed method was shown to be far more efficient than the existing methods for microstructures with small ratios of interfacial widths to the domain size. In [116], the authors used an adaptive mesh refinement with a second-order L_0-stable scheme. In [46], they used the exponential time differencing method [28]. Eyre reported that the AC equation becomes an unconditionally gradient stable algorithm if the free energy functional is split appropriately into contractive and expansive parts,

$$\begin{aligned}\mathcal{E}(c) &= \int_a^b \left(\frac{F(c)}{\epsilon^2}+\frac{c_x^2}{2}\right)dx \\ &= \int_a^b \left(\frac{c^4+1}{4\epsilon^2}+\frac{c_x^2}{2}\right)dx - \int_a^b \frac{c^2}{2\epsilon^2}dx = \mathcal{E}_c(c)-\mathcal{E}_e(c) \end{aligned} \tag{2.4}$$

and the contractive part $\mathcal{E}_c(c)$ and expansive part $-\mathcal{E}_e(c)$ are treated implicitly and explicitly, respectively [47]. The nonlinearly stabilized splitting scheme that involves a semi-implicit time and centered difference space discretizations of Eq. (2.1) is

$$\frac{c_i^{n+1}-c_i^n}{\Delta t} = \frac{c_i^n-(c_i^{n+1})^3}{\epsilon^2}+\Delta_h c_i^{n+1} \text{ for } i=1,\cdots,N;\ n=0,\cdots,N_t-1, \tag{2.5}$$

where the discrete notations are defined in the following section. In [23], the authors reported the energy decreasing property for the corresponding discrete problem by using the eigenvalues of the Hessian matrix of the energy functional. An implicit Euler's scheme is

$$\frac{c_i^{n+1}-c_i^n}{\Delta t} = \frac{c_i^{n+1}-(c_i^{n+1})^3}{\epsilon^2}+\Delta_h c_i^{n+1}. \tag{2.6}$$

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 [48]. Eq. (2.5) can be reformulated as

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

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

$$\Delta t_{equiv} = \frac{\Delta t \epsilon^2}{\Delta t + \epsilon^2} \leq \min(\Delta t, \epsilon^2). \tag{2.7}$$

That is, the scaled equivalent time step Δt_{equiv} is bounded by ϵ^2, which is a small value. In [31, 108], authors addressed unconditional stability of AC algorithms given by Eyre [48] and showed that the equivalent time step is bounded.

As Eyre noticed, the explicit Euler's scheme is not gradient stable and it was found that the stable time steps are severely restricted. The implicit Euler's scheme is conditionally gradient stable and it suffers from uniqueness problem of solution with large time steps. It turns out that the allowed time step sizes are just less than twice the explicit Euler's step. The Crank-Nicolson scheme also suffers from the solvability restriction. The semi-implicit Euler's scheme is much better compared to the explicit Euler's scheme, but still not gradient stable [48]. Eyre's scheme Eq. (2.5) has also the equivalent time step limitation. In this paper, we propose an unconditionally stable and accurate numerical method for solving the AC equation. The proposed method is based on operator splitting techniques. The AC equation was divided into a linear and a nonlinear equation. The linear equation was solved using a Crank-Nicolson scheme and the nonlinear equation was then solved analytically due to the availability of a closed-form solution. 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 AC equation involves two steps. First, $c_t = \Delta c$ is solved numerically using a Crank-Nicolson method. Second, $c_t = (c - c^3)/\epsilon^2$ is solved analytically.

This Chapter is organized as follows. In section 2.2, we propose an unconditionally stable second-order hybrid numerical method for the AC equation. The numerical results showing the accuracy, efficiency, and stability of the proposed method are presented in section 2.3. Finally, conclusions are drawn in section 2.4.

2.2. Proposed operator splitting algorithm

In this section, an unconditionally stable second-order hybrid numerical method is proposed for the AC equation. The unconditional stability means that arbitrarily large time steps can be used in the numerical algorithm. For simplicity, we shall discretize the AC equation in one-dimensional space, i.e., $\Omega = (a, b)$. Two and three-dimensional discretizations are defined analogously. Let N be a positive even integer, $h = (b - a)/N$ be a uniform grid size, and $\Omega_h = \{x_i = (i - 0.5)h,\ 1 \leq i \leq N\}$ be the set of cell-centers. Let c_i^n be the approximations of $c(x_i, n\Delta t)$, 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 zero Neumann boundary condition, Eq. (2.2), is first implemented by requiring that for each n,

$$\nabla_h c^n_{\frac{1}{2}} = \nabla_h c^n_{N+\frac{1}{2}} = 0,$$

where the discrete differentiation operator is $\nabla_h c^n_{i+\frac{1}{2}} = (c^n_{i+1} - c^n_i)/h$. We then define a discrete Laplacian operator as $\Delta_h c_i = (\nabla_h c_{i+\frac{1}{2}} - \nabla_h c_{i-\frac{1}{2}})/h$ and a discrete l_2 inner product

as

$$\langle \mathbf{c}, \mathbf{d} \rangle_h = h \sum_{i=1}^{N} c_i d_i,$$

where $\mathbf{c} = (c_1, c_2, \cdots, c_N)$. We also define the discrete l_2 and maximum norms, respectively, as $||\mathbf{c}||_h = \sqrt{\langle \mathbf{c}, \mathbf{c} \rangle_h}$ and $\|\mathbf{c}\|_\infty = \max_{1 \le i \le N} |c_i|$. We propose the following operator splitting scheme, which is an unconditionally stable second-order accurate hybrid scheme.

$$\frac{c_i^* - c_i^n}{\Delta t} = \frac{1}{2}(\Delta_h c_i^* + \Delta_h c_i^n), \tag{2.8}$$

$$\frac{c_i^{n+1} - c_i^*}{\Delta t} = \frac{c_i^{n+1} - (c_i^{n+1})^3}{\epsilon^2}. \tag{2.9}$$

Eq. (2.8) is a Crank-Nicolson scheme for $c_t = \Delta c$ with an initial condition c^n. Using a von Neumann stability analysis [45], it can be seen that this scheme is unconditionally stable. The resulting implicit discrete system of equations can be solved by a fast solver such as a multigrid method [11, 153]. Eq. (2.9) can be considered as an approximation of the equation

$$c_t = \frac{c - c^3}{\epsilon^2} \tag{2.10}$$

by an implicit Euler's method with the initial condition c^*. We can solve Eq. 2.10 analytically by the method of separation of variables [143]. The solution is given as follows:

$$c_i^{n+1} = \frac{c_i^* e^{\frac{\Delta t}{\epsilon^2}}}{\sqrt{1 + (c_i^*)^2 (e^{\frac{2\Delta t}{\epsilon^2}} - 1)}} = \frac{c_i^*}{\sqrt{e^{-\frac{2\Delta t}{\epsilon^2}} + (c_i^*)^2 (1 - e^{-\frac{2\Delta t}{\epsilon^2}})}}.$$

The proposed operator splitting algorithm is shown schematically in Fig. 2.2.

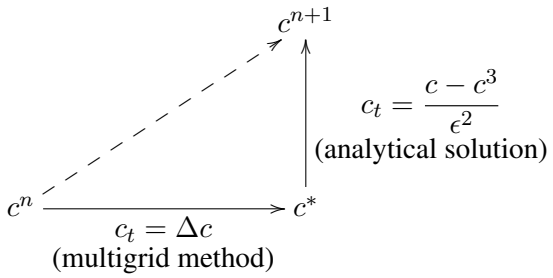


FIGURE 2.2. A hybrid numerical method for the AC equation.

Finally, the proposed scheme can be written as follows:

$$\frac{c_i^* - c_i^n}{\Delta t} = \frac{1}{2}(\Delta_h c_i^* + \Delta_h c_i^n) \text{ for } i = 1, \cdots, N \text{ and } n = 0, \cdots, N_t - 1$$

$$c_i^{n+1} = \frac{c_i^*}{\sqrt{e^{-\frac{2\Delta t}{\epsilon^2}} + (c_i^*)^2 (1 - e^{-\frac{2\Delta t}{\epsilon^2}})}}.$$

2.3. Numerical experiments

In this section, the following numerical tests were performed: finding the relationship between the ϵ value and the width of the transition layer, traveling wave solutions, convergence test, stability and accuracy test, motion by the mean curvature, and the AC equation with a logarithmic free energy. From the results of these numerical tests, it was confirmed that the proposed scheme is second-order accurate in space and time and has unconditional stability.

2.3.1. The relationship between the ϵ value and the width of the transition layer. Across the interfacial regions, the concentration field varies from -0.9 to 0.9 over a distance of approximately $2\sqrt{2}\epsilon \tanh^{-1}(0.9)$. Therefore, if we want this value to be approximately $m(> 0)$ grid points, the ϵ value needs to be taken as follows:

$$\epsilon_m = \frac{hm}{2\sqrt{2}\tanh^{-1}(0.9)}.$$

To confirm this, a simulation was run with the initial condition $c(x, 0) = 0.01\text{rand}(x)$ on the unit domain $\Omega = (0, 1)$ with $h = 1/128,\ \Delta t =$ E-5, and ϵ_4. Here, rand(x) is a random number between -1 and 1. In Fig. 2.3, the transition layer (from $c = -0.9$ to $c = 0.9$) is approximately 4 grid points at time $t =$3E-3 (circled line).

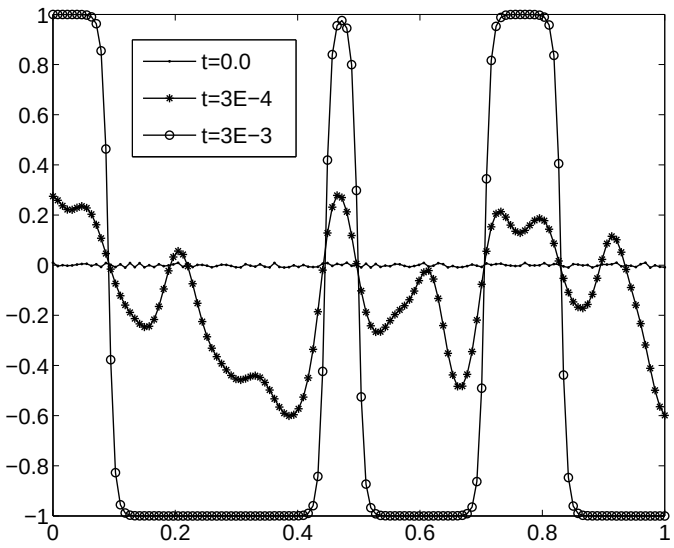


FIGURE 2.3. The evolution of an initial random distribution of concentration, $c(x, 0) = 0.01\text{rand}(x)$. The concentration profile is shown at $t = 0,$ 3E-4, and 3E-3.

2.3.2. Traveling wave solutions. Traveling wave solutions of Eq. (2.1) were obtained in the following form:

$$c(x, t) = \frac{1}{2}\left(1 - \tanh\frac{x - st}{2\sqrt{2}\epsilon}\right), \tag{2.11}$$

where $s = 3/(\sqrt{2}\epsilon)$ is the speed of the traveling wave [23]. In Fig. 2.4, the numerical traveling wave solutions (where the lines with symbols '$*$', '+', and '$\cdot$' represent 128, 256, and 512 grids, respectively) with an initial profile (dashed line), $c(x,0) = \frac{1}{2}(1 - \tanh\frac{x}{2\sqrt{2}\epsilon})$, on a computational domain, $\Omega = (-0.5, 5.5)$. The following are selected: final time $T = 2/s$, time step $\Delta t = T/16$, and $\epsilon = 0.0197$. The analytic final profile is $c(x,T) = \frac{1}{2}(1 - \tanh\frac{x-2}{2\sqrt{2}\epsilon})$. The convergence of the results with grid refinement is qualitatively evident.

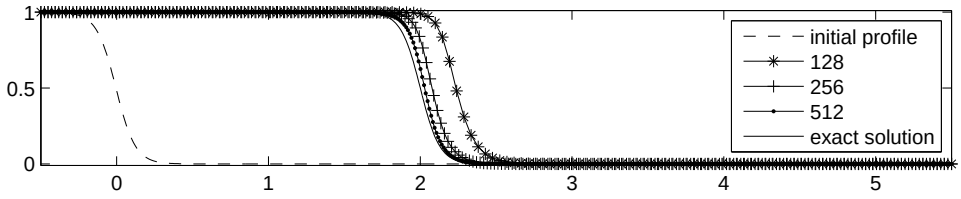


FIGURE 2.4. Numerical traveling wave solutions with an initial profile (dashed line), $c(x,0) = \frac{1}{2}(1 - \tanh\frac{x}{2\sqrt{2}\epsilon})$. The final time is $T = 2/s$. The analytic solution at T is a solid line. Lines with symbols, '$*$', '$+$', and '$\cdot$' represent 128, 256, and 512 grids, respectively.

2.3.3. Convergence test. The rate of convergence of the scheme is difficult to prove analytically. However, numerical experimentation suggests that the scheme is second order accurate in space and time. A quantitative estimate of the rate of convergence was obtained by performing a number of simulations for the same initial profile on a set of increasingly finer space grids and time steps. The initial conditions are

$$c(x,0) = \frac{1}{2}\left(1 - \tanh\frac{x}{2\sqrt{2}\epsilon}\right), \tag{2.12}$$

$$c(x,y,0) = \frac{1}{2}\left(1 - \tanh\frac{x}{2\sqrt{2}\epsilon}\right), \text{ and} \tag{2.13}$$

$$c(x,y,z,0) = \frac{1}{2}\left(1 - \tanh\frac{x}{2\sqrt{2}\epsilon}\right) \tag{2.14}$$

for one-, two-, and three-dimensional tests, respectively. The numerical solutions with initial condition Eqs. (2.12)-(2.14) were calculated on the computational domain $\Omega = (-0.5, 1.5)$, $\Omega = (-0.5, 1.5) \times (-0.5, 1.5)$, and $\Omega = (-0.5, 1.5) \times (-0.5/8, 1.5/8) \times (-0.5/8, 1.5/8)$, respectively. For one- and two- dimensional simulations, the parameters are $h = 2^{1-n}$, $\epsilon = 0.015$, and $s = 3/(\sqrt{2}\epsilon)$ for $n = 7, 8, 9,$ and 10. For each grid we integrate to time $T = 1/s$ with $\Delta t = h/(16s)$. Note that as we refine the space step we also refine the time step. The error of the numerical solution was defined as $\mathbf{e}^{N_t} = (e_1^{N_t}, e_2^{N_t}, \cdots, e_N^{N_t})$, where $e_i^{N_t} = c_i^{N_t} - c(x_i, T)$ for $i = 1, \cdots, N$. Tables 2.1, 2.2, and 2.3 show the discrete l_2 and maximum norms of the errors and rates of convergence in one-, two-, and three- dimensions, respectively. These results suggest that the scheme is indeed second-order accurate in space and time.

TABLE 2.1. Convergence results in one dimension.

Case	128	Rate	256	Rate	512	Rate	1024
$\|\mathbf{e}^{N_t}\|_2$	3.444E-2	1.973	8.775E-3	1.962	2.252E-3	1.924	5.937E-4
$\|\mathbf{e}^{N_t}\|_\infty$	1.424E-1	1.969	3.637E-2	1.957	9.365E-3	1.915	2.483E-3

TABLE 2.2. Convergence results in two dimensions.

Case	128×128	Rate	256×256	Rate	512×512	Rate	1024×1024
$\|\mathbf{e}^{N_t}\|_2$	4.872E-2	1.973	1.241E-2	1.962	3.185E-3	1.929	8.367E-4
$\|\mathbf{e}^{N_t}\|_\infty$	1.425E-1	1.969	3.639E-2	1.960	9.351E-3	1.924	2.465E-3

TABLE 2.3. Convergence results in three dimensions.

Case	64×8^2	Rate	128×16^2	Rate	256×32^2	Rate	512×64^2
$\|\mathbf{e}^{N_t}\|_2$	3.218E-2	1.902	8.612E-3	1.973	2.193E-3	1.969	5.601E-4
$\|\mathbf{e}^{N_t}\|_\infty$	4.909E-1	1.785	1.424E-1	1.969	3.638E-2	1.967	9.304E-3

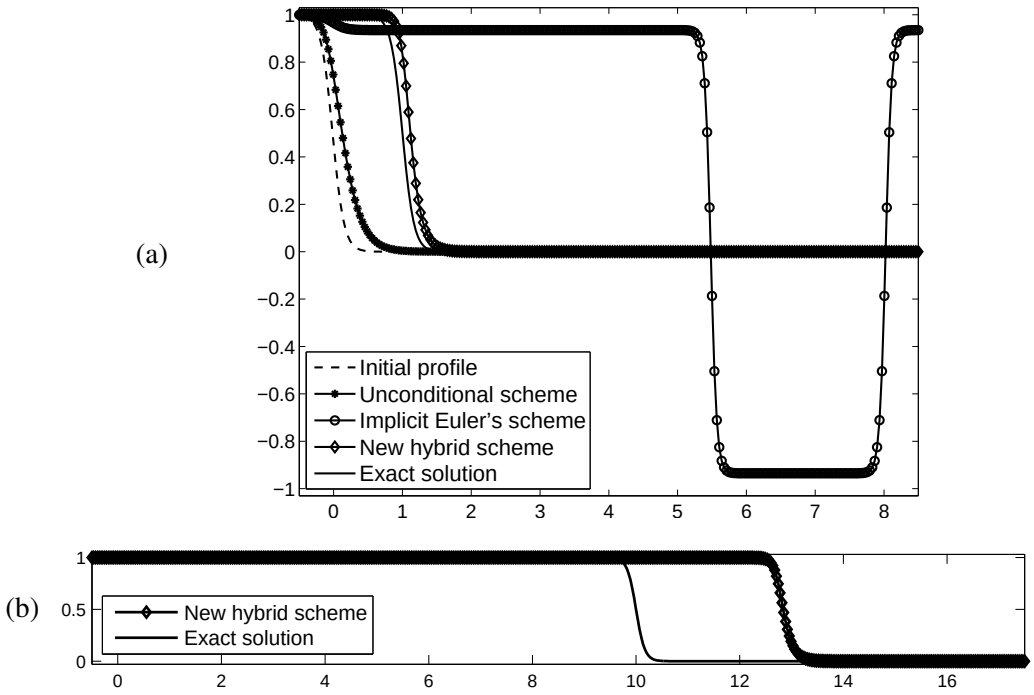
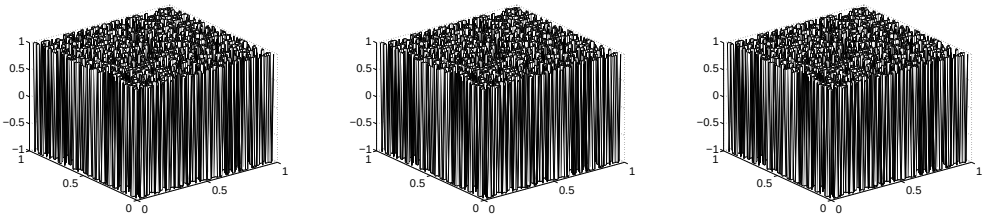


FIGURE 2.5. (a) One step evolutions with three different schemes with a time step $\Delta t = 1/s$. (b) A numerical solution with new hybrid scheme at $t = 10\Delta t$.

2.3.4. Stability and accuracy tests. The stability of the proposed numerical algorithm was compared with other methods. Fig. 2.5 (a) shows one step evolutions with three different



(a) $\Delta t = 100$ (b) $\Delta t = 10000$ (c) $\Delta t = 1000000$

FIGURE 2.6. Snapshots after ten time step iterations with three different time steps.

schemes with a time step $\Delta t = 1/s$ and ϵ_7, on the computational domain, $\Omega = (-0.5, 8.5)$, with a 256 grid. The dashed line is the initial configuration, $c(x,0) = \frac{1}{2}(1 - \tanh\frac{x}{2\sqrt{2}\epsilon_7})$ and the solid line is the exact solution $c(x,\Delta t) = \frac{1}{2}(1 - \tanh\frac{x-1}{2\sqrt{2}\epsilon_7})$ at time $t = \Delta t$. The lines denoted with the star, circle, and diamond symbols are the numerical solutions with the unconditional scheme, implicit Euler's scheme and new hybrid scheme, respectively. The result with the unconditional scheme traveled with the wrong speed, as was expected from the time step rescaling property, Eq. (2.7). In this case, the equivalent time step of the unconditional scheme is $\Delta t_{equiv} \approx 0.1115\Delta t$. The implicit Euler's scheme is unstable and reaches a wrong profile. The new hybrid scheme is closest to the exact solution and is stable with respect to a relatively large time step, $\Delta t = 1/s$. Fig. 2.5 (b) shows a numerical solution with new hybrid scheme at $t = 10\Delta t$. We can see that the new hybrid scheme is stable with a long time evolution.

Next, we perform a numerical experiment with an example of spinodal decomposition of a binary mixture in order to demonstrate the unconditionally stability of the scheme. In this simulation, the initial condition was random perturbation with the maximum amplitude 0.02.

$$c(x, y, 0) = 0.02\text{rand}(x, y).$$

A 64×64 mesh was used on the computational domain $\Omega = (0, 1) \times (0, 1)$ and different time steps, $\Delta t = 100$, $\Delta t = 10000$, and $\Delta t = 1000000$ were employed for the time integration. We took the simulation parameters, $h = 1/64$ and ϵ_7. In Fig. 6.2, we display snapshots after ten time step iterations with three different time steps. These results suggest that the scheme is indeed unconditionally stable.

The effect of the equivalent time step was compared with the proposed numerical algorithm. Fig. 2.7 shows 200-step evolutions using two different schemes with a time step $\Delta t = \epsilon_7^2$, a 1024 grid and ϵ_7 on the computational domain, $\Omega = (-0.5, 8.5)$. The dashed line is the initial configuration, $c(x,0) = \frac{1}{2}(1 - \tanh\frac{x}{2\sqrt{2}\epsilon_7})$ and the solid line is the exact solution calculated by Eq. (2.11) at time $t = 200\Delta t$. The symbols denoted with the star and circle are the numerical solutions with the unconditional scheme Eq. (2.5) and new hybrid scheme, respectively. The result shows that while the numerical solution with the new hybrid scheme is close to the exact solution, the one with the unconditional scheme is far from the exact solution since the equivalent time step is $\Delta t_{equiv} = \Delta t\epsilon_7^2/(\Delta t + \epsilon_7^2) = 0.5\Delta t$.

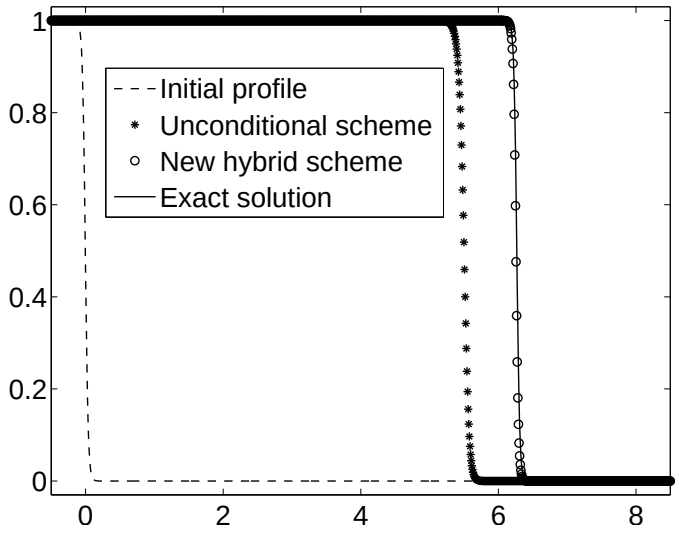


FIGURE 2.7. The comparison with equivalent unconditional scheme with $\Delta t = \epsilon_7^2$ at time $t = 200\Delta t$.

2.3.5. Mean curvature flow. Numerical simulations of surfaces evolving according to their mean curvature are presented. Eq. (2.1) can be rewritten in the following form:

$$c_t = \frac{c - c^3}{\epsilon^2} + \Delta c.$$

It was formally shown that, as $\epsilon \to 0$, the zero level set of c, which is denoted by $\Gamma_t^\epsilon := \{\mathbf{x} \in \Omega : c(\mathbf{x}, t) = 0\}$, approaches a surface Γ_t that evolves according to the geometric law

$$V = -\kappa = -\left(\frac{1}{R_1} + \frac{1}{R_2}\right), \tag{2.15}$$

where V is the normal velocity of the surface, Γ_t, at each point, κ is its mean curvature, and R_1 and R_2 are the principal radii of curvatures at the point of the surface [1, 5, 51]. In d-dimensions, with the same radii of curvature, Eq. (2.15) becomes

$$V = \frac{1 - d}{R}. \tag{2.16}$$

If the initial radius of the circular region is set to R_0 and the radius at time t is denoted as $R(t)$, then Eq. (2.16) becomes $dR(t)/dt = (1 - d)/R(t)$. Its solution is given as follows:

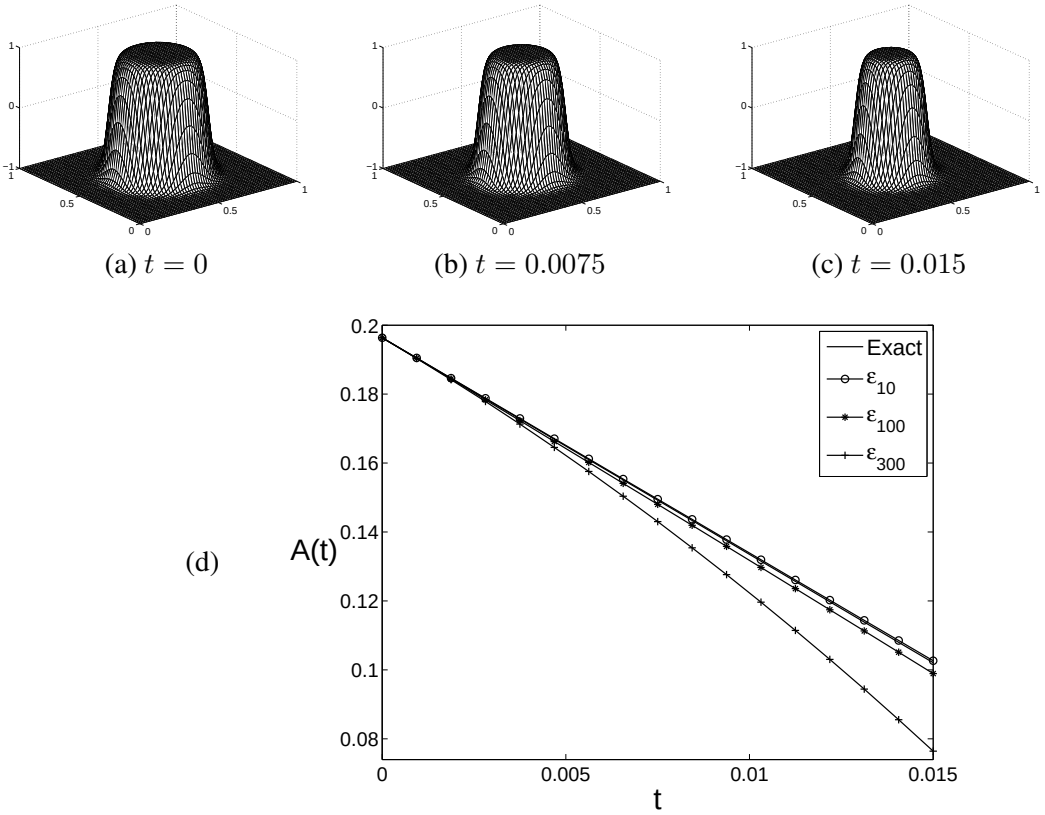
$$R(t) = \sqrt{R_0^2 + 2(1 - d)t}. \tag{2.17}$$

Hence the analytic area $A(t)$ at time t is

$$A(t) = \pi(R_0^2 + 2(1 - d)t).$$

First, a two-dimensional test was performed with the following initial condition

$$c(x, y, 0) = \tanh \frac{0.25 - \sqrt{(x - 0.5)^2 + (y - 0.5)^2}}{\sqrt{2}\epsilon} \tag{2.18}$$



(a) $t = 0$ (b) $t = 0.0075$ (c) $t = 0.015$

(d)

FIGURE 2.8. (a), (b), and (c) show the evolution of the initial concentration $c(x, y, 0)$ given by Eq. (2.18). The times are shown below each figure. (d) illustrates evolution of the radius with three different ϵ. Lines with the '$+$', '$*$', and '$\circ$' symbols denote ϵ_{300}, ϵ_{100}, and ϵ_{10}, respectively.

on the computational domain, $\Omega = (0, 1) \times (0, 1)$ with a 512×512 mesh and a time step, $\Delta t = 0.0015h$. Fig. 2.8 (a), (b), and (c) show the evolution of the initial concentration $c(x, y, 0)$ given by Eq. (2.18). The times are shown below each figure. Fig. 2.8 (d) shows that as the ϵ size decreases from ϵ_{300} ('$+$') to ϵ_{100} ('$*$') and ϵ_{10} ('$\circ$'), the numerical area $A(t)$ of the circle with time approaches the asymptotic value (solid line) given by Eq. (2.17).

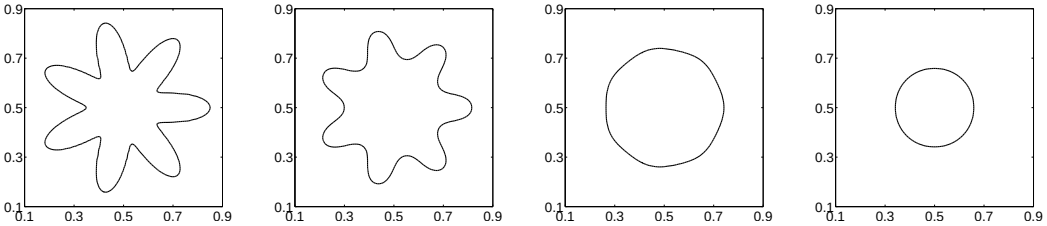
Next, a three-dimensional test was performed with the following initial condition

$$c(x, y, z, 0) = \tanh \frac{0.4 - \sqrt{(x - 0.5)^2 + (y - 0.5)^2 + (z - 0.5)^2}}{\sqrt{2}\epsilon}$$

on the computational domain, $\Omega = (0, 1) \times (0, 1) \times (0, 1)$ with a $256 \times 256 \times 256$ mesh and a time step $\Delta t =$ 7E-6. Fig. 2.9 presents snapshots of the zero isosurface of the solution in three-dimensional space. The times are shown below each figure. At time $t = 0.0175$, the numerical radii of the sphere are 0.2999 (ϵ_{20}), 0.2991 (ϵ_{40}), and 0.2990 (ϵ_{60}), respectively. The analytic radius from Eq. (2.17) is 0.3. From these two- and three-dimensional numerical tests, it was confirmed that $\Gamma_t^\epsilon \to \Gamma_t$ as $\epsilon \to 0$.

(a) $t = 0$ (b) $t = 0.0175$

FIGURE 2.9. (a) and (b) are initial and final isosurfaces.



(a) $t = 0$ (b) $t = 0.001$ (c) $t = 0.005$ (d) $t = 0.020$

FIGURE 2.10. Evolution of a star-shaped interface in a curvature-driven flow. The tips of the star move inward, while the gaps between the tips move outward.

Fig. 2.10 shows the evolution of a star-shaped interface in a curvature-driven flow on the computational domain, $\Omega = (0,1) \times (0,1)$, with a 128×128 mesh and $\Delta t =$ 5E-5. The initial configuration is defined as follows:

$$c(x,y,0) = \tanh \frac{0.25 + 0.1\cos(7\theta) - \sqrt{(x-0.5)^2 + (y-0.5)^2}}{\sqrt{2}\epsilon_4},$$

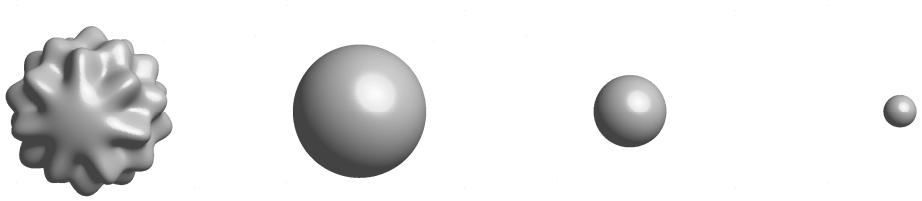
where

$$\theta = \begin{cases} \tan^{-1}\left(\frac{y-0.5}{x-0.5}\right) & \text{if } x > 0.5 \\ \pi + \tan^{-1}\left(\frac{y-0.5}{x-0.5}\right) & \text{otherwise.} \end{cases}$$

The tips of the star move inward, while the gaps between the tips move outward. Once it deforms to a circular shape, the radius of the circle shrinks with increasing speed.

Fig. 2.11 shows the evolution of a sphere perturbed with a spherical harmonic in curvature-driven flow on the computational domain, $\Omega = (0,1) \times (0,1) \times (0,1)$ with a $256 \times 256 \times 256$ mesh and $\Delta t =$E-5. The initial configuration is defined as follows

$$c(x,y,z,0) = \tanh \frac{0.25 + 0.1Y_{10,7}(\theta,\phi) - r}{\sqrt{2}\epsilon_8},$$



(a) $t = 0$ (b) $t = 0.003$ (c) $t = 0.012$ (d) $t = 0.015$

FIGURE 2.11. Evolution of a sphere perturbed with a spherical harmonic in a curvature-driven flow.

where $r = \sqrt{(x-0.5)^2 + (y-0.5)^2 + (z-0.5)^2}$, $Y_{10,7}(\theta, \phi)$ is a spherical harmonic [26],

$$\theta = \begin{cases} \tan^{-1}\left(\frac{y-0.5}{x-0.5}\right) & \text{if } x > 0.5 \\ \pi + \tan^{-1}\left(\frac{y-0.5}{x-0.5}\right) & \text{otherwise} \end{cases}$$

is a polar angle, and $\phi = \cos^{-1}\left(\frac{z-0.5}{r}\right)$ is the azimuthal angle. As in the two-dimensional case, once it deforms to a spherical shape, the radius of the sphere shrinks with increasing speed.

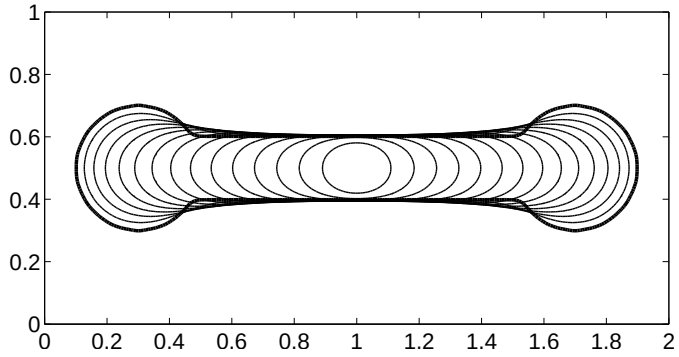
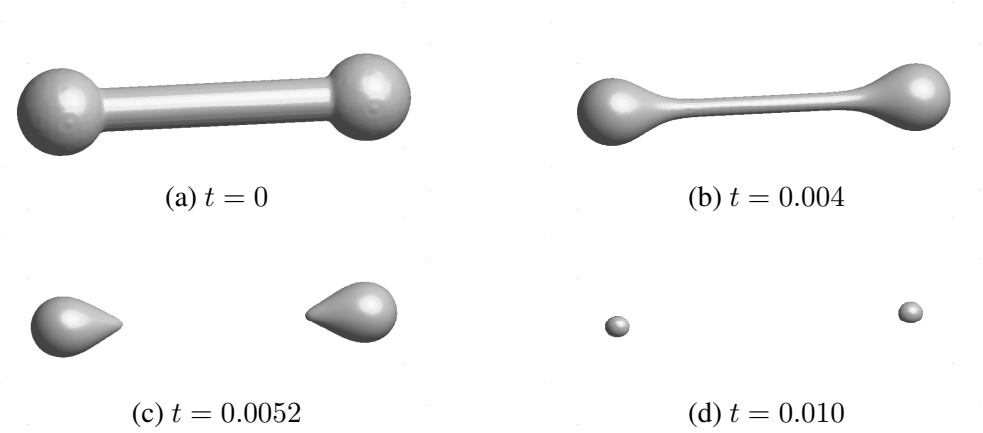


FIGURE 2.12. Temporal evolution of the two-dimensional dumbbell shape. The computation was performed on a 512×256 mesh with $h = 1/256$, ϵ_{15}, $\Delta t = 8E\text{-}5$, and $T = 0.072$. The thicker line represents the initial configuration and the succeeding contour lines are increased by the time interval, $4.8E\text{-}3$.

Fig. 2.12 shows the temporal evolution of the two-dimensional dumbbell shape. The initial configuration is given by a dumbbell where the inner strip is centered on $y = 0.5$, and has a width of 0.2, and the circles at either end of the strip have centers at $(0.3, 0.5)$, $(1.7, 0.5)$ with a radius of 0.2. The computation is performed on a 512×256 mesh with $h = 1/256$, ϵ_{15}, $\Delta t = 8$E-5, and $T = 0.072$. The thicker line represents the initial configuration and the succeeding contour lines are incremented by the time interval, 4.8E-3.

Fig. 2.13 shows the temporal evolution of a three-dimensional dumbbell shape. The initial configuration is given by a dumbbell where the inner cylinder, which is centered on

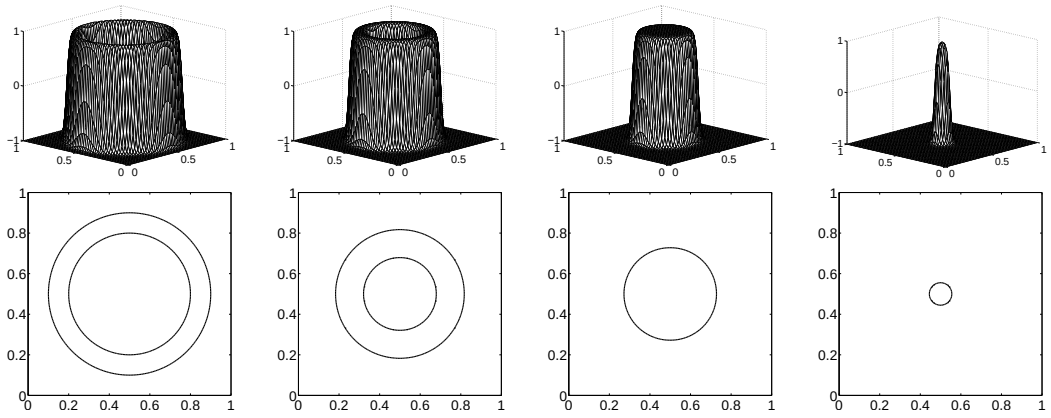


(a) $t = 0$ (b) $t = 0.004$

(c) $t = 0.0052$ (d) $t = 0.010$

FIGURE 2.13. Temporal evolution by the mean curvature of the three-dimensional dumbbell shape. (a) The initial surface. The times are shown below each figure. The computation was performed on a $256 \times 128 \times 128$ grid.

$(x, 0.5, 0.5)$, has a radius of 0.1, and the spheres at either end of the cylinder have centers at $(0.3, 0.5, 0.5)$, $(1.7, 0.5, 0.5)$ with a radius 0.2. The computation was carried out on a $256 \times 128 \times 128$ mesh with $h = 1/128$, ϵ_8, and $\Delta t = 4$E-5. Initially the radius of the handle was much smaller than that of the spheres on the end. Therefore, the handle was expected to shrink at a faster rate so that the handle would pinch off after some time. The handle pinches off at approximately $t = 0.005$, and the two spheres disappear at approximately $t = 0.0106$.



(a) $t = 0$ (b) $t = 0.0294$ (c) $t = 0.0534$ (d) $t = 0.0780$

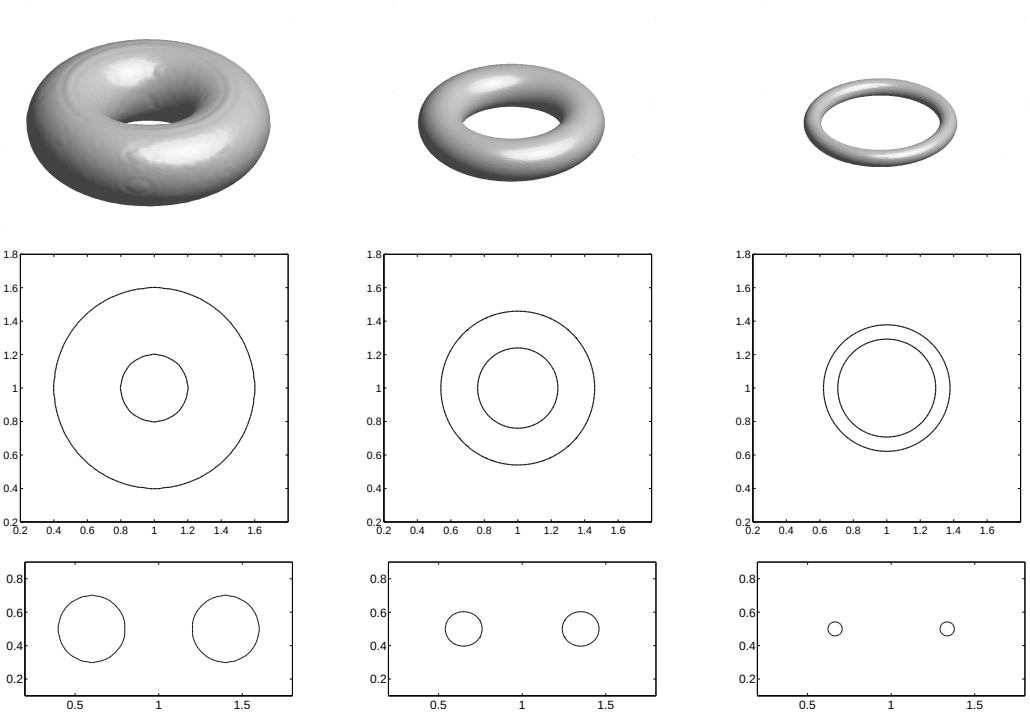
FIGURE 2.14. Evolution of the double circles in a curvature-driven flow on the domain $\Omega = (0, 1) \times (0, 1)$ with a 128×128 mesh, $\Delta t = 6E$-5, and ϵ_8.

Fig. 2.14 shows the evolution of the double circles on the computational domain $\Omega = (0, 1) \times (0, 1)$ with a 128×128 mesh and $\Delta t = 6$E-5. The initial configuration is defined as

follows:

$$c(x,y,0) = \tanh\frac{0.4-\sqrt{(x-0.5)^2+(y-0.5)^2}}{\sqrt{2}\epsilon} - \tanh\frac{0.3-\sqrt{(x-0.5)^2+(y-0.5)^2}}{\sqrt{2}\epsilon} - 1.$$

The smaller circle has a larger curvature, and shrinks faster than the larger circle. After the smaller circle disappears, the larger one also decreases in size.



(a) $t = 0$ (b) $t = 0.016$ (c) $t = 0.0208$

FIGURE 2.15. Evolution of the torus in curvature-driven flow on the domain $\Omega = (0,2) \times (0,2) \times (0,1)$ with a $256 \times 256 \times 128$ mesh, $\Delta t = 4E\text{-}5$, and ϵ_8. The times are shown below each figure.

Fig. 2.15 (a) shows the evolution of the torus in a curvature-driven flow on the computational domain, $\Omega = (0,2) \times (0,2) \times (0,1)$ with a $256 \times 256 \times 128$ mesh, $\Delta t = 4$E-5, and ϵ_8. Fig. 2.15 (b) and (c) show the horizontal and vertical sections of the torus, respectively. Unlike the two-dimensional case, the inner circle increases in size because the mean curvature drives the motion into the inside of the torus. This is a three-dimensional phenomenon.

2.3.6. The Allen-Cahn equation with a logarithmic free energy. Our proposed method can be applied to a class of potentials $F(c)$. For example, let us consider the logarithmic free energy, i.e., $F(c) = \frac{\theta}{2}[(1+c)\ln(1+c) + (1-c)\ln(1-c)] - \frac{\theta_c}{2}c^2$, where θ and θ_c are positive

constants with $\theta < \theta_c$. Fig. 2.16 shows the logarithmic free energy $F(c)$ with $\theta = 1.0$ and $\theta_c = 1.2$.

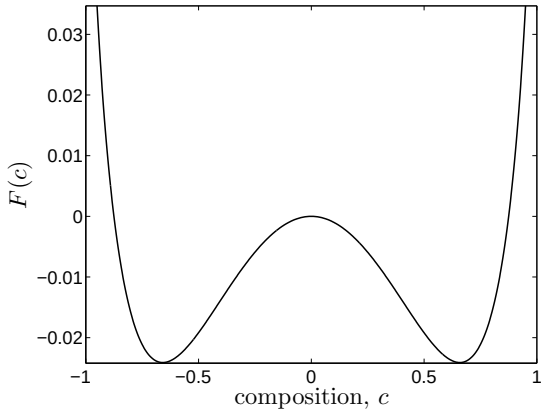


FIGURE 2.16. A logarithmic free energy, $F(c) = \frac{\theta}{2}[(1+c)\ln(1+c) + (1-c)\ln(1-c)] - \frac{\theta_c}{2}c^2$, with $\theta = 1.0$ and $\theta_c = 1.2$.

In this case, we propose the following operator splitting scheme:

$$\frac{c_i^* - c_i^n}{\Delta t} = \frac{1}{2}(\Delta_h c_i^* + \Delta_h c_i^n), \tag{2.19}$$

$$\frac{c_i^{n+1} - c_i^*}{\Delta t} = -\frac{1}{2\epsilon^2}(F'(c^{n+1}) + F'(c^*)). \tag{2.20}$$

Since a closed form solution is not available for $c_t = -F'(c)/\epsilon^2$ with a logarithmic free energy, we apply Newton's method [8] to solve Eq. (2.20). It turns out that Newton's algorithm typically requires only one or two iterations to achieve acceptable accuracy.

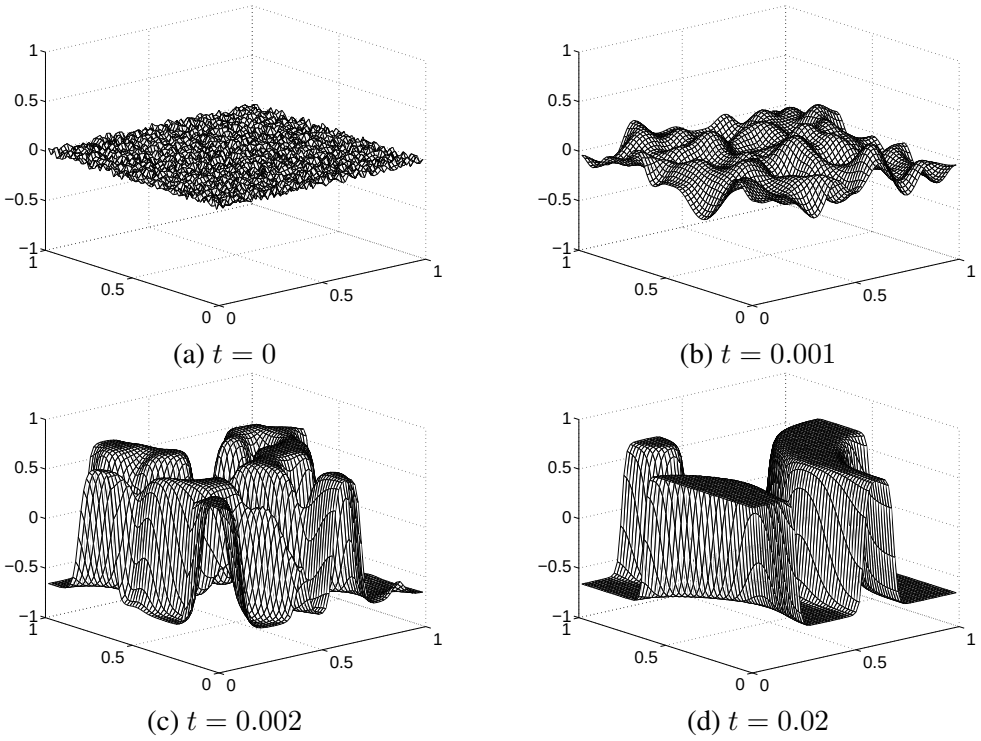
As a test problem, we consider spinodal decomposition of a binary mixture. The initial condition was random perturbation with the maximum amplitude 0.1

$$c(x, y, 0) = 0.1\text{rand}(x, y)$$

on the computational domain $\Omega = (0, 1) \times (0, 1)$. In this test, a 64×64 mesh and a time step $\Delta t = 5\text{E-}5$ were used. We took the simulation parameters, $h = 1/64$, $\epsilon = 0.0075$, $\theta = 1.0$, and $\theta_c = 1.2$. In Fig. 2.17, we display the evolution of the phase-field at different times. From the results it can be seen that our proposed scheme is applicable to the AC equation with potentials other than $F'(c) = c^3 - c$.

2.4. Conclusions

As Eyre noticed, the explicit Euler's scheme for the CH equation is not gradient stable and the stable time steps are severely restricted. The implicit Euler's scheme is conditionally gradient stable and it suffers from uniqueness problem of solution with large time steps. It turns out that the allowed time step sizes are just less than twice the explicit Euler's step. The Crank-Nicolson scheme also suffers from the solvability restriction. The semi-implicit Euler's scheme is much better compared to the explicit Euler's scheme, but still not gradient stable [48]. Eyre's



(a) $t = 0$ (b) $t = 0.001$

(c) $t = 0.002$ (d) $t = 0.02$

FIGURE 2.17. The evolution of the phase-field at different times. The times are shown below each figure.

scheme has also the equivalent time step limitation. In this Chapter, an unconditionally stable second-order hybrid numerical method was developed to solve the AC equation. The proposed method was based on operator splitting techniques. The linear equation was solved using a Crank-Nicolson scheme and the nonlinear equation was then solved analytically. A variety of numerical experiments were presented to confirm the accuracy, efficiency, and stability of the proposed method. In particular, the scheme was shown to be unconditionally stable and second-order accurate in both time and space.

This Chapter is published in Computers and Mathematics with Applications, Vol. 60, No. 6, pp. 1591–1606, 2010.

Chapter 3

An unconditionally stable hybrid method for image segmentation

3.1. Introduction

Image segmentation is one of the fundamental tasks in automatic image analysis. Its goal is to partition a given image into regions that contain distinct objects. For example, the segmentation of structures from images is an important first step for object recognition [14], interpretation [119], and image inpainting [12, 19, 33]. The most common form of segmentation is based on the assumption that distinct objects in an image have different and approximately constant colors. A natural approach is therefore to decompose an image domain into approximately homogeneous regions that are separated by sharp changes in image features. One of the general approaches for image segmentation is the minimizer of the piecewise constant Mumford-Shah functional [118]. Chan-Vese [36, 154] solved the minimization problem by the level set method proposed by Osher and Sethian[123]. The level set method is used to trace interfaces separating a domain into subdomains and contour the image in the zero level set of the interface. Recently, the Allen-Chan equation [1] has been used in solving the image segmentation problems [7, 22, 50, 104]. Esedo$\bar{g}$lu and Tsai [50] proposed the Allen-Cahn equation, also known as the phase field model, as a method to solve the reduced Mumford-Shah problem with the Chan-Vese fitting terms.

In this Chapter, we propose an unconditionally stable hybrid numerical method which consists of the Allen-Cahn equation and a fitting term. An operator splitting technique is used to solve the model numerically. We describe its numerical solution algorithm and give a proof of the unconditional stability of the scheme. We also present various numerical results on real images and synthetic images with various types and levels of noise to demonstrate the efficiency, robustness, and accuracy of the proposed numerical method.

This Chapter is organized as follows. In section 3.2, three models for image segmentation are briefly reviewed. In section 3.3, we describe the proposed unconditionally stable hybrid operator splitting method and provide a proof of the unconditional stability of the scheme. In section 3.4, we perform some characteristic numerical experiments for image segmentation. Finally, conclusions are given in section 3.5.

3.2. Description of the previous models

In this section, we briefly review three approaches such as Mumfor-Shah, Chan-Vese, and phase-field models for image segmentation.

3.2.1. Mumford-Shah model. With a given image f_0 on the image domain Ω and its segmenting curve C, Mumford and Shah [118] proposed that the segmentation of an image can be obtained through the minimization of the following Mumford-Shah energy functional:

$$\mathcal{E}^{MS}(f,C) = \mu Length(C) + \int_{\Omega} |f_0(\mathbf{x}) - f(\mathbf{x})|^2 d\mathbf{x} + \nu \int_{\Omega\setminus C} |\nabla f(\mathbf{x})|^2 d\mathbf{x}, \tag{3.1}$$

where μ, ν are positive parameters and f is the piecewise smooth approximation to f_0. However, in practice it is not easy to minimize this functional because of the unknown set C of lower dimension than f.

3.2.2. Chan-Vese model. Chan and Vese proposed an algorithm for decomposing the image into two regions with piecewise constant approximations by minimizing the energy of the Mumford and Shah functional

$$\begin{aligned} \mathcal{E}^{\mathrm{CV}}(\mathrm{c}_1,\mathrm{c}_2,\mathrm{C}) &= \mu Length(C) + \lambda_1 \int_{inside(C)} |f_0(\mathbf{x}) - c_1|^2 d\mathbf{x} \\ &+ \lambda_2 \int_{outside(C)} |f_0(\mathbf{x}) - c_2|^2 d\mathbf{x}, \end{aligned} \tag{3.2}$$

where μ, λ_1 and λ_2 are positive parameters [67, 113]. The constants c_1 and c_2 are the averages of f_0 inside and outside of C, respectively. To avoid the same problem as the Mumform and Shah model, Chan and Vese [36] replaced the unknown curve C by the level-set function $\phi(\mathbf{x})$,

$$\phi(\mathbf{x}) \begin{cases} > 0 & \text{if } \mathbf{x} \in \text{inside } C, \\ = 0 & \text{if } \mathbf{x} \in C, \\ < 0 & \text{if } \mathbf{x} \in \text{outside } C. \end{cases} \tag{3.3}$$

Then the energy functional $\mathcal{E}^{CV}(c_1,c_2,C)$ can be rewritten by $\mathcal{E}^{CV}(c_1,c_2,\phi)$ as

$$\begin{aligned} \mathcal{E}^{CV}(c_1,c_2,\phi) &= \mu \int_{\Omega} \delta_\epsilon(\phi(\mathbf{x}))\, |\nabla\phi(\mathbf{x})|\, d\mathbf{x} + \lambda_1 \int_{\Omega} |f_0(\mathbf{x}) - c_1|^2 H_\epsilon(\phi(\mathbf{x})) d\mathbf{x} \\ &+ \lambda_2 \int_{\Omega} |f_0(\mathbf{x}) - c_2|^2 (1 - H_\epsilon(\phi(\mathbf{x}))) d\mathbf{x}, \end{aligned} \tag{3.4}$$

where H_ϵ and δ_ϵ are the regularized approximations of the Heaviside function and the Dirac delta function, respectively, and are defined as

$$H_\epsilon(z) = \frac{1}{2} + \frac{1}{\pi} \arctan\left(\frac{z}{\epsilon}\right) \text{ and } \delta_\epsilon(z) = \frac{\epsilon}{\pi(\epsilon^2 + z^2)} \tag{3.5}$$

and the constants c_1 and c_2 represent the mean intensity value of their own regions and are defined by

$$c_1 = \frac{\int_\Omega f_0(\mathbf{x}) H_\epsilon(\phi(\mathbf{x})) d\mathbf{x}}{\int_\Omega H_\epsilon(\phi(\mathbf{x}))) d\mathbf{x}} \text{ and } c_2 = \frac{\int_\Omega f_0(\mathbf{x})(1 - H_\epsilon(\phi(\mathbf{x}))) d\mathbf{x}}{\int_\Omega (1 - H_\epsilon(\phi(\mathbf{x}))) d\mathbf{x}}. \tag{3.6}$$

By using the gradient descent method, we get the following:

$$\frac{\partial\phi}{\partial t} = \delta_\epsilon(\phi) \left[\mu\nabla \cdot \left(\frac{\nabla\phi}{|\nabla\phi|} \right) - \lambda_1 (f_0(\mathbf{x}) - c_1)^2 + \lambda_2 (f_0(\mathbf{x}) - c_2)^2 \right]. \tag{3.7}$$

The level set based algorithm of Chan and Vese can be used to process the image with a large amount of noise and detect objects whose boundaries can not be defined by gradient.

3.2.3. Phase-field model. A phase-field approximation for minimizing the Mumford-Shah functional, by using Allen-Cahn equation to replace the length of the segmenting curve C, is given by the following energy functional:

$$\mathcal{E}(\phi) = \int_\Omega \left(\frac{F(\phi)}{\varepsilon^2} + \frac{|\nabla\phi|^2}{2} + G(\phi, f_0) \right) d\mathbf{x}, \tag{3.8}$$

where $F(\phi) = 0.25(\phi^2 - 1)^2$ is a double-well potential as shown in Fig. 2.1, ϵ is the gradient energy coefficient related to the interfacial energy, and Ω is the image domain. The third term in the functional is defined as

$$G(\phi, f_0) = \frac{\lambda}{2} \left[(1+\phi)^2 (f_0 - c_1)^2 + (1-\phi)^2 (f_0 - c_2)^2 \right].$$

Here, λ is a nonnegative parameter and f_0 is the given image. Also c_1 and c_2 are the averages of f_0 in the regions $(\phi \geq 0)$ and $(\phi < 0)$, respectively:

$$c_1 = \frac{\int_\Omega f_0(\mathbf{x})(1+\phi(\mathbf{x}))d\mathbf{x}}{\int_\Omega (1+\phi(\mathbf{x}))d\mathbf{x}} \text{ and } c_2 = \frac{\int_\Omega f_0(\mathbf{x})(1-\phi(\mathbf{x}))d\mathbf{x}}{\int_\Omega (1-\phi(\mathbf{x}))d\mathbf{x}}. \tag{3.9}$$

Once ϕ reaches a steady state, the zero level set of ϕ becomes the contour that separates the object from the background. For this purpose, we seek a law of evolution in the form [34]:

$$\phi_t = -\text{grad}\mathcal{E}(\phi).$$

The symbol 'grad' here denotes the gradient in the space $L^2(\Omega)$. Let $\phi, \varphi \in D = \{c \in H^2(\Omega) | \frac{\partial c}{\partial n} = 0 \text{ on } \partial\Omega\}$. Then, we have

$$\begin{aligned}
&(\text{grad}\mathcal{E}(\phi), \varphi)_{L^2} = \lim_{h\to 0} \frac{\mathcal{E}(\phi + h\varphi) - \mathcal{E}(\phi)}{h} \\
= &\int_\Omega \left(\frac{F'(\phi)}{\varepsilon^2} - \Delta\phi + \lambda[(1+\phi)(f_0 - c_1)^2 - (1-\phi)(f_0 - c_2)^2] \right) \varphi d\mathbf{x} \\
= &\left(\frac{F'(\phi)}{\varepsilon^2} - \Delta\phi + \lambda[(1+\phi)(f_0 - c_1)^2 - (1-\phi)(f_0 - c_2)^2], \varphi \right)_{L^2}.
\end{aligned}$$

Therefore, we get the following gradient descent flow equation:

$$\phi_t = -\frac{F'(\phi)}{\varepsilon^2} + \Delta\phi - \lambda[(1+\phi)(f_0 - c_1)^2 - (1-\phi)(f_0 - c_2)^2]. \tag{3.10}$$

3.3. Numerical solution

In this section, we propose a new unconditionally stable hybrid numerical method for minimizing the piecewise constant Mumford-Shah functional of image segmentation. An operator splitting technique is used to solve the governing partial differential equations numerically. We split its numerical solution algorithm into two linear equations and one nonlinear equation. One of the linear equations and the nonlinear equation are solved analytically due to the availability of closed-form solutions. The other linear equation is discretized using an implicit scheme and the resulting discrete system of equations is solved by a fast numerical method such as a multigrid method. We analyze and prove the unconditional stability of the scheme.

3.3.1. Proposed numerical scheme. We shall discretize Eq. (3.10) in a two dimensional space, i.e.,$\Omega = (a, b) \times (c, d)$. Let N_x and N_y be positive even integers, $h = (b - a)/N_x$ be the uniform mesh size, and $\Omega_h = \{(x_i, y_j) : x_i = (i - 0.5)h,\ y_j = (j - 0.5)h,\ 1 \le i \le N_x,\ 1 \le j \le N_y\}$ be the set of cell-centers. Let ϕ_{ij}^n be approximations of $\phi(x_i, y_j, n\Delta t)$, 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. Then we propose the following operator splitting numerical algorithm:

$$\begin{aligned}\frac{\phi^{n+1} - \phi^n}{\Delta t} = &- \frac{F'(\phi^{n+1})}{\epsilon^2} + \Delta_d \phi^{n+1,2} \\ &- \lambda[(1 + \phi^{n+1,1})(f_0 - c_1^n)^2 - (1 - \phi^{n+1,1})(f_0 - c_2^n)^2], \end{aligned} \tag{3.11}$$

where $F'(\phi) = \phi(\phi^2 - 1)$ and $\phi^{n+1,k}$ for $k = 1, 2$ are defined in the following operator splitting scheme.

$$\frac{\phi^{n+1,1} - \phi^n}{\Delta t} = -\lambda[(1 + \phi^{n+1,1})(f_0 - c_1^n)^2 - (1 - \phi^{n+1,1})(f_0 - c_2^n)^2], \tag{3.12}$$

$$\frac{\phi^{n+1,2} - \phi^{n+1,1}}{\Delta t} = \Delta_d \phi^{n+1,2}, \tag{3.13}$$

$$\frac{\phi^{n+1} - \phi^{n+1,2}}{\Delta t} = -\frac{F'(\phi^{n+1})}{\epsilon^2}, \tag{3.14}$$

where c_1^n and c_2^n are defined as follows:

$$c_1^n = \frac{\sum_i^{N_x}\sum_j^{N_y} f_{0,ij}(1 + \phi_{ij}^n)}{\sum_i^{N_x}\sum_j^{N_y}(1 + \phi_{ij}^n)} \text{ and } c_2^n = \frac{\sum_i^{N_x}\sum_j^{N_y} f_{0,ij}(1 - \phi_{ij}^n)}{\sum_i^{N_x}\sum_j^{N_y}(1 - \phi_{ij}^n)}. \tag{3.15}$$

We can consider Eq. (3.12) is an approximation of the equation

$$\phi_t = -\lambda[(f_0 - c_1)^2 + (f_0 - c_2)^2]\phi - \lambda[(f_0 - c_1)^2 - (f_0 - c_2)^2] \tag{3.16}$$

by an implicit Euler's method with an initial condition ϕ^n. We can solve Eq. (3.16) analytically since it is a first-order linear differential equation and the solution at $t = \Delta t$ is given as

$$\begin{aligned}\phi^{n+1,1} = &\ e^{-\lambda[(f_0 - c_1^n)^2 + (f_0 - c_2^n)^2]\Delta t}\phi^n \\ &+ (e^{-\lambda[(f_0 - c_1^n)^2 + (f_0 - c_2^n)^2]\Delta t} - 1)\frac{(f_0 - c_1^n)^2 - (f_0 - c_2^n)^2}{(f_0 - c_1^n)^2 + (f_0 - c_2^n)^2}. \end{aligned} \tag{3.17}$$

Next the implicit discrete Eq. (3.13) can be solved by a multigrid method [11, 153] with the initial condition $\phi^{n+1,1}$. Finally we can consider Eq. (3.14) as an approximation of the equation

$$\phi_t = \frac{\phi - \phi^3}{\epsilon^2} \tag{3.18}$$

by an implicit Euler's method with the initial condition $\phi^{n+1,2}$. Then the solution at $t = \Delta t$ of Eq. (3.18), solved by the method of separation of variables [143], is given as

$$\phi^{n+1} = \frac{\phi^{n+1,2}}{\sqrt{e^{\frac{-2\Delta t}{\epsilon^2}} + (\phi^{n+1,2})^2(1 - e^{\frac{-2\Delta t}{\epsilon^2}})}}.$$

Finally, our proposed scheme is written as

$$\begin{aligned}
\phi^{n+1,1} &= e^{-\lambda[(f_0-c_1^n)^2+(f_0-c_2^n)^2]\Delta t}\phi^n \\
&\quad +(e^{-\lambda[(f_0-c_1^n)^2+(f_0-c_2^n)^2]\Delta t} - 1)\frac{(f_0 - c_1^n)^2 - (f_0 - c_2^n)^2}{(f_0 - c_1^n)^2 + (f_0 - c_2^n)^2}, && (3.19)\\
\frac{\phi^{n+1,2} - \phi^{n+1,1}}{\Delta t} &= \Delta_d \phi^{n+1,2}, && (3.20)\\
\phi^{n+1} &= \frac{\phi^{n+1,2}}{\sqrt{e^{\frac{-2\Delta t}{\epsilon^2}} + (\phi^{n+1,2})^2(1 - e^{\frac{-2\Delta t}{\epsilon^2}})}}. && (3.21)
\end{aligned}$$

The solutions of Eqs. (3.19) and (3.21) are explicitly defined. Eq. (3.20) is a heat equation and we apply a fast solver such as a multigrid method to solve the equation.

3.3.2. Stability analysis for the proposed scheme. When we solve time-dependent partial differential equations, stability of the numerical scheme to the equations is very important. Explicit time integration schemes are generally only conditionally stable and require small time steps to be employed to insure numerical stability. And the step size restriction is often more severe than accuracy considerations require. However, our proposed hybrid splitting method is an unconditionally stable scheme. For simplicity, let us define

$$\alpha = e^{-\lambda[(f-c_1^n)^2+(f-c_2^n)^2]\Delta t} \text{ and } \beta = \frac{(f_0 - c_1^n)^2 - (f_0 - c_2^n)^2}{(f_0 - c_1^n)^2 + (f_0 - c_2^n)^2}.$$

Then, Eq. (3.19) can be rewritten as follows:

$$\phi^{n+1,1} = \alpha\phi^n + (\alpha - 1)\beta.$$

Now, we get

$$|\phi^{n+1,1}| \le \alpha|\phi^n| + (1 - \alpha)|\beta| \le \alpha + 1 - \alpha = 1. \tag{3.22}$$

Here we have used $0 < \alpha \le 1$, $|\beta| \le 1$, and $|\phi^n| \le 1$. For Eq. (3.20), a von Neumann stability analysis [45] shows that an implicit Euler's method is unconditionally stable. The inequality $|\phi^{n+1,2}| \le \|\phi^{n+1,1}\|_\infty$ is satisfied by the discrete maximum principle for the heat equation [117]. Then by Eq. (3.22), we get $|\phi^{n+1,2}| \le 1$. And for the Eq. (3.21):

$$|\phi^{n+1}| = \frac{\left|\phi^{n+1,2}\right|}{\sqrt{e^{\frac{-2\Delta t}{\epsilon^2}} + (\phi^{n+1,2})^2(1 - e^{\frac{-2\Delta t}{\epsilon^2}})}} = \frac{1}{\sqrt{1 + \left(\frac{1}{(\phi^{n+1,2})^2} - 1\right)e^{\frac{-2\Delta t}{\epsilon^2}}}} \le 1.$$

Therefore, if $|\phi^n| \leq 1$, then we get $|\phi^{n+1}| \leq 1$. Therefore the proposed scheme is unconditionally stable for any time step. And we also define the numerical quadrature for the energy functional, Eq. (5.1).

$$\mathcal{E}(\phi) = \sum_{i}^{N_x}\sum_{j}^{N_y} h^2 \left(\frac{F(\phi_{ij})}{\varepsilon^2} + \frac{|\nabla\phi_{ij}|^2}{2} + \frac{\lambda}{2}(1+\phi_{ij})^2(f_{0,ij}-c_1)^2 + \frac{\lambda}{2}(1-\phi_{ij})^2(f_{0,ij}-c_2)^2 \right). \tag{3.23}$$

3.4. Experimental results

In this section, we present numerical results using the proposed numerical algorithm on various synthetic and real images. We show that a very fast and accurate minimization can be achieved by the proposed algorithm. In our numerical experiments, we normalize the given image f as $f_0 = \frac{f-f_{min}}{f_{max}-f_{min}}$, where f_{max} and f_{min} are the maximum and the minimum values of the given image, respectively. Across the interfacial regions, the phase field varies from -0.9 to 0.9 over a distance of approximately $2\sqrt{2}\epsilon\tanh^{-1}(0.9)$. Therefore, if we want this value to be approximately m grid points, then the ϵ value needs to be taken as follows:

$$\epsilon_m = \frac{hm}{2\sqrt{2}\tanh^{-1}(0.9)}.$$

Since solutions with the proposed numerical scheme are almost insensitive to the initial configuration of ϕ, we simply initialize $\phi^0 = 2f_0 - 1$.

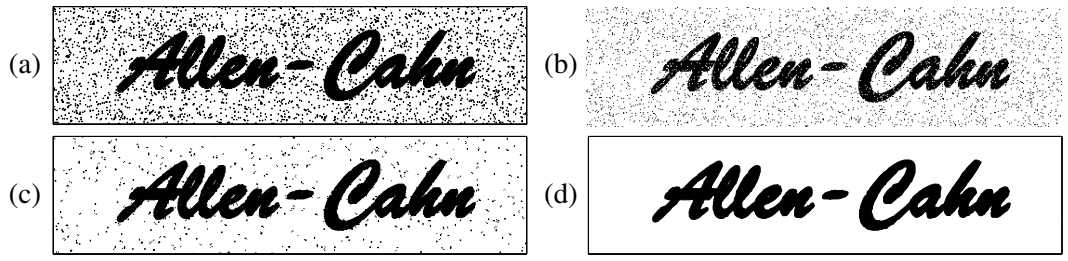


FIGURE 3.1. (a) Original image with 10% salt-and-pepper noise. (b), (c), and (d) are the contours at times $t = 0$, 2E-5 (1 iteration), and 8E-5 (4 iterations), respectively.

3.4.1. Salt-and-pepper noise. We start with an example with a synthetic image. In Fig. 3.1, (a) is the image named 'Allen-Cahn' with 'Brush Script MT' Font and deblurring in 10% salt-and-pepper noise [2] on the computational domain, $\Omega = (0,4)\times(0,1)$ with a 512×128 mesh. Interface parameter ϵ_8, $\Delta t =$ 2E-5, and $\lambda =$ 1E4 are used. Salt-and-pepper noise is defined as randomly occurring white and black pixels. The given probability $r\%$ means setting a fraction of $(r/2)\%$ randomly selected pixels to black and the other $(r/2)\%$ randomly to white. (b), (c), and (d) are zero-level filled contours at times $t = 0$, 2E-5 (1 iteration), and 8E-5 (4 iterations), respectively. We also note that we can segment the enclosed holes in the letters. Fig. 3.2 is an image segmentation for a fingerprint on the computational domain $\Omega = (0,1)\times(0,1)$

with a 256×256 mesh. Interface parameter ϵ_2, time step $\Delta t = 5\text{E-}6$, and $\lambda = 5\text{E}4$ are used. We can observe that the proposed model fully segments the image after only 4 iterations.

FIGURE 3.2. Finger print. From left to right: (The first row : Initial image, 10% salt-and-pepper noise is superposed over the original, contour of the initial image with noise, The second row : $t = 4\text{E-}6$, $t = 8\text{E-}6$, $t = 2\text{E-}5$).

3.4.2. Gray-scale noise. Fig. 3.3 shows characters with gray-scale noise. $r\%$ gray-scale noise means that $r\%$ of the pixels in the image are replaced with random numbers chosen from a uniform distribution between 0 and 1. The computational domain is set to $\Omega = (0, 4) \times (0, 1)$ with a 512×128 mesh. Interface parameter ϵ_{10}, time step $\Delta t = 5\text{E-}5$, and $\lambda = 5\text{E}3$ are used. The left column consists of the original images with 10%, 25%, 50% noise respectively, and the right column consists of the restored images. The proposed model segments the image with 10% and 25% gray-scale noise. It should be noticed that, since the noise is so much higher, the result for the 50% noise-blurred image cannot completely contour the image. But on the whole, our proposed model has done successfully.

3.4.3. Contours without gradient. In this example, we show that our proposed model can be used to detect cognitive contours from objects which cannot be defined by a gradient. We want to test our method on a very challenging image with scattered data, i.e., a satellite image of Europe showing clusters of light. In Fig. 3.4, the segmentation of Europe night-lights is shown. The computational domain is set to $\Omega = (0, 1) \times (0, 1)$ with a 256×256 mesh. Interface parameter ϵ_{600}, time step $\Delta t = 1.25\text{E-}4$, and $\lambda = 7.5\text{E}4$ are used. The method produces visually clear results. It only took 20 iterations, which is one order of magnitude smaller than the previous methods [36, 104].

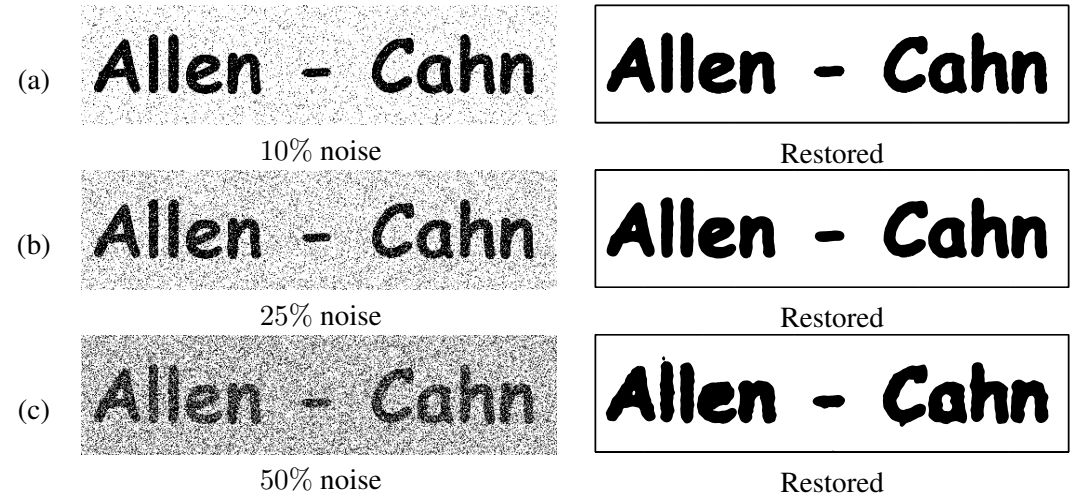


FIGURE 3.3. The same image with gray-scale noise. (a) (left) Initial image with 10% gray-scale noise and (right) reconstructed values with only 4 iterations. (b) (left) Initial image with 25% gray-scale noise and (right) reconstructed values with only 5 iterations. (c) (left) Initial image with 50% gray-scale noise and (right) reconstructed values with only 8 iterations.

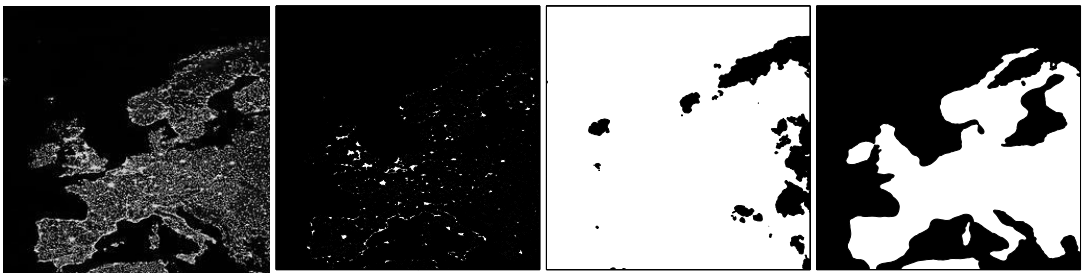
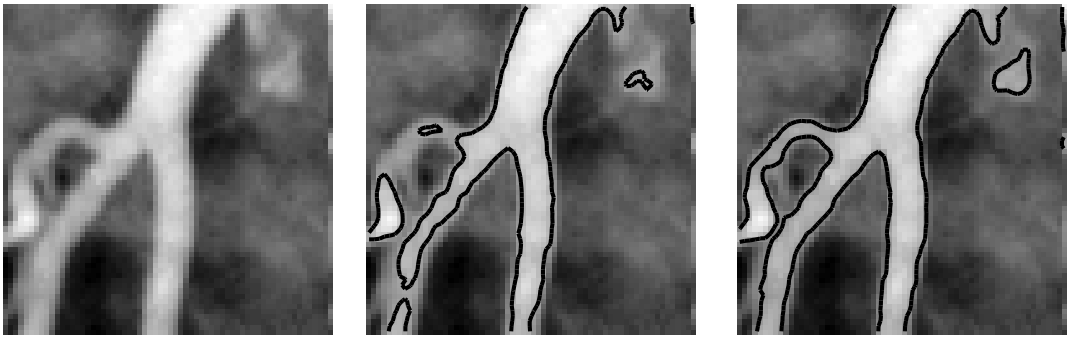
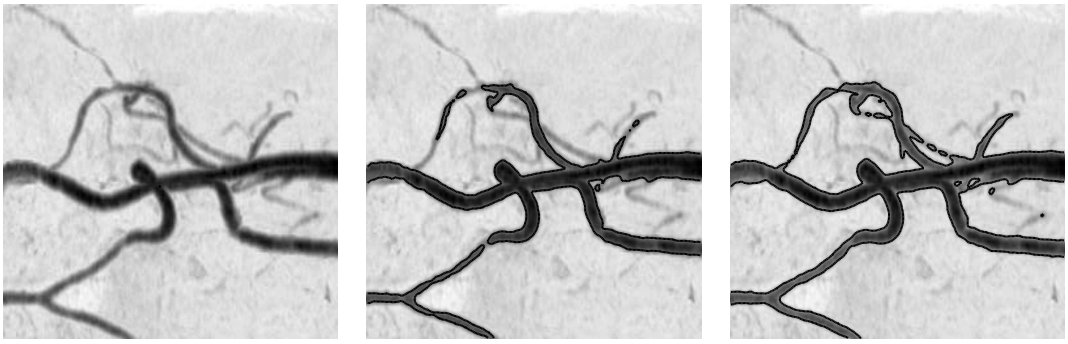


FIGURE 3.4. Europe night-lights. (a) initial image, (b) contour of the initial image, (c) $t = 2.5$E-4 (2 iterations), and (d) $t = 2.5$E-3 (20 iterations).

3.4.4. Blood vessel image. Fig. 3.5 shows the segmentation results for two real blood vessel (left anterior descending) images with inhomogeneous intensity via use of the proposed numerical method. The computational domain in the first row is set to $\Omega = (0,1) \times (0,1)$ with a 256×256 mesh. Interface parameter ϵ_{20}, time step $\Delta t = 1.6$E-6, and $\lambda = 1$E6 are used. And the second row is set to $\Omega = (0,1) \times (0,1)$ with a 64×64 mesh. Interface parameter ϵ_6, time step $\Delta t = 1.6$E-6, and $\lambda = 5$E5 are used. It can be seen from the third column of Fig. 3.5 that the images are successfully segmented after 16 iterations.

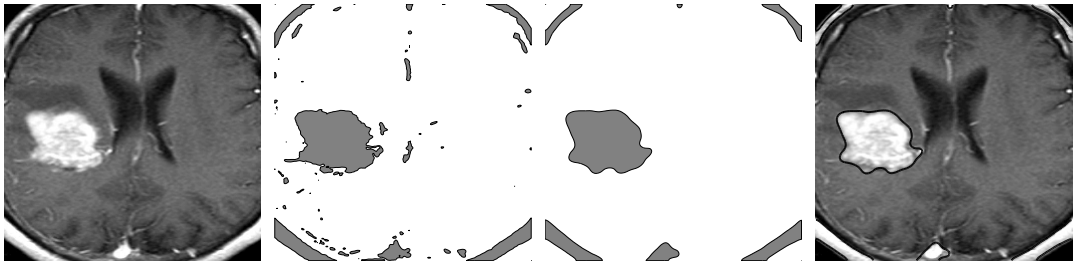


(a) 0 iteration (b) 8 iterations (c) 16 iterations



(d) 0 iteration (e) 8 iterations (f) 16 iterations

FIGURE 3.5. The images with two left anterior descending vessels. The first column contains initial images. The second and the third columns are images with contours. The iteration numbers are shown below each figure.

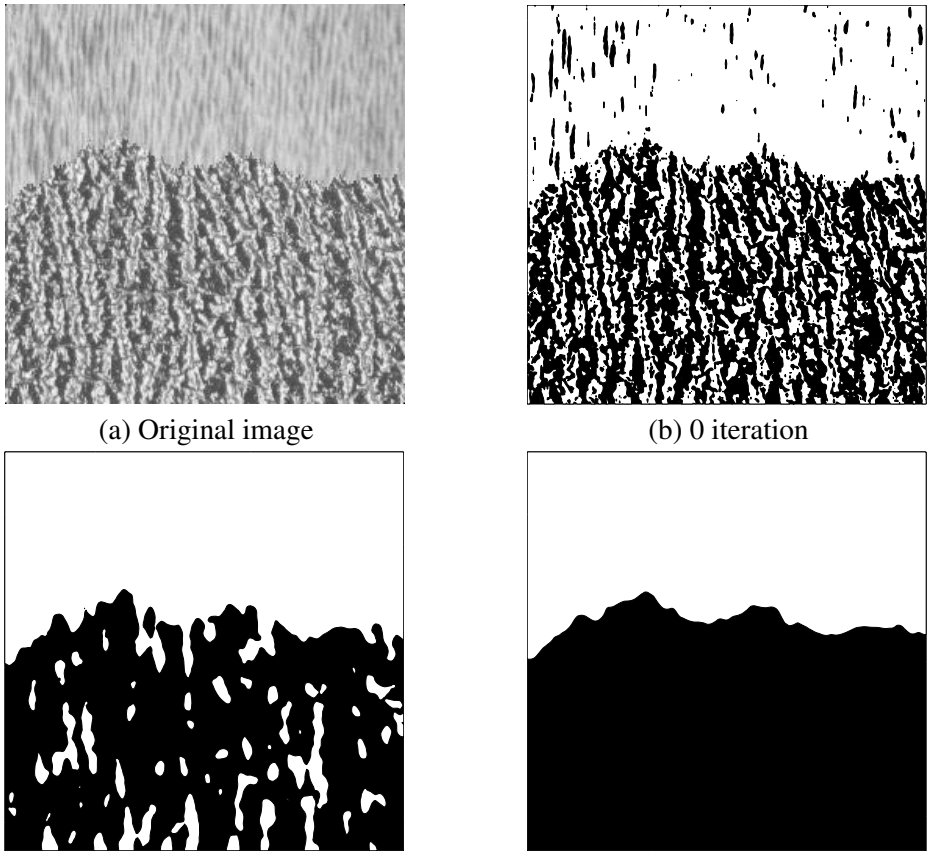


(a) (b) (c) (d)

FIGURE 3.6. Segmentation of the image for a solid brain tumor. (a) Initial image, (b) contour of the initial image, (c) contour the image (20 iteration), and (d) contour is superposed over the initial image to show the accuracy.

3.4.5. Brain MR image. We show that our proposed model can be used to analyze medical images to provide necessary and useful information for medical treatment. In Fig. 3.6, the segmentations of brain MR image are shown on the computational domain $\Omega = (0,1) \times (0,1)$ with a 256×256 mesh. Interface parameter ϵ_5, time step $\Delta t =$ 5E-6, and $\lambda =$ 1E4 are used. As

can be observed, the agreement between the area of the brain solid tumor and the segmentation of image is obvious.



(a) Original image (b) 0 iteration

(c) 2 iterations (d) 6 iterations

FIGURE 3.7. Texture image. (a) Initial image, (b) contour of the initial image, (c) $t =$ 2E-4 (2 iterations), and (d) $t =$ 6E-4 (6 iterations).

3.4.6. Texture image. Texture image, based on local spatial variations of intensity or color to identify these types of homogeneous image regions, is one of the most important attributes used in image analysis and pattern recognition. Fig. 3.7 shows that our proposed model can be very useful in detecting texture image segmentation. The computational domain is set to $\Omega = (0, 1) \times (0, 1)$ with a 256×256 mesh. Interface parameter ϵ_{50}, time step $\Delta t =$ 1E-4, and $\lambda =$ 1E4 are used. As can be seen, our proposed method has performed well in texture image segmentation with smoothing.

3.5. Conclusion

In this Chapter, we proposed an unconditionally stable hybrid numerical scheme for minimizing the problems associated with the piecewise constant Mumford-Shah functional of image segmentation. The model and its numerical scheme are based on the Allen-Cahn equation and

an operator splitting technique, respectively. We described the numerical solution algorithm and gave a proof of the unconditional stability of the numerical scheme. Finally various experimental results on real images and synthetic images with various types and levels of noise were presented to demonstrate the accuracy and efficiency of the proposed method.

Chapter 4

A fast and accurate numerical method for medical image segmentation

4.1. Introduction

Image segmentation is one of the fundamental tasks in automatic image analysis. Its goal is to partition a given image into several regions in each of which the intensity is homogeneous. Up to now, a great number of algorithms have been proposed to solve the image segmentation problem. Among them, there are roughly two classical basic models based on the edges or based on the regions, which are widely used with great efforts nowadays. For the geodesic snake [3, 21, 25, 98, 85, 165] based on the edges, gradient flow is used as a stopping operator to get accurate boundaries with high variation in gradient to attract the curve to object boundary, while Chan-Vese method [36, 154], which is a representative model based on the regions and widely applied for various applications in image aspects, may suffer from the intensity inhomogeneity, since it only relies on the global information of homogeneous regions. For the medical images are the kind of images, filled with multiple merging and splitting of contours, geometric active contour model [3, 21, 25, 98, 165], which is a famous theory based on the level set method [123] and acting as the most importance tools for simultaneously tracking the edges of objects by geometric flows, may be the better choice to efficiently perform those segmentations.

In this Chapter, we propose a robust and accurate geometric active contour model and its numerical solution algorithm for image segmentation. The model is based on the modified Allen-Cahn equation. An operator splitting technique is used to solve the model numerically. In particular, we propose an initialization algorithm based on an edge stepping function for the fast image segmentation. We present various numerical results on artificial and real images to demonstrate the robustness, and accuracy of the proposed numerical method.

The outline of this Chapter is the following. In Section 4.2, the governing equation based on Allen-Cahn equation is presented. In Section 4.3, we describe the proposed hybrid operator splitting method. In Section 4.4, we perform some characteristic numerical experiments for computational examples. Finally, in Section 4.5, conclusions are drawn.

4.2. Modified Allen-Cahn equation

For a given image $f_0(\mathbf{x})$, where $\mathbf{x} = (x, y)$, on the image domain Ω and its segmenting curve C. By the level-set function $\phi(\mathbf{x})$,

$$\phi(\mathbf{x}) \begin{cases} > 0 & \text{if } \mathbf{x} \in \text{inside } C, \\ = 0 & \text{if } \mathbf{x} \in C, \\ < 0 & \text{if } \mathbf{x} \in \text{outside } C. \end{cases} \tag{4.1}$$

The geometric active contour model based on the mean curvature motion is given by the following evolution equation [21]:

$$\phi_t = g(f_0)|\nabla\phi|\nabla\cdot\left(\frac{\nabla\phi}{|\nabla\phi|}\right) + \lambda g(f_0)|\nabla\phi|, \tag{4.2}$$

where λ is a parameter and g is an edge stopping function. For example,

$$g(f_0(\mathbf{x})) = \frac{1}{1+|\nabla(G_\sigma * f_0)(\mathbf{x})|^2}, \tag{4.3}$$

where $(G_\sigma * f_0)(\mathbf{x})$, a smoother version of f_0, is the convolution of the given image f_0 with the Gaussian function $G_\sigma = \frac{1}{2\pi\sigma^2}e^{-(x^2+y^2)/(2\sigma^2)}$. The function $g(f_0(\mathbf{x}))$ is close to zero where the gradient of the image is high and is close to one in homogeneous regions.

We propose a phase-field approximation for the geometric active segmentation by using the modified Allen-Cahn equation. In a phase-field model, we introduce an order parameter, ϕ, which is plus one (black) and minus one (white). Then, the governing equation is as follows:

$$\phi_t = g(f_0)\left(-\frac{F'(\phi)}{\epsilon^2} + \Delta\phi\right) + \lambda g(f_0)F(\phi), \tag{4.4}$$

where $F(\phi) = 0.25(1-\phi^2)^2$ is a double-well potential. Note that Eq. (4.4) is similar to the model proposed in [1], however, the new one is more robust and efficient compared to previous model.

4.3. Proposed numerical solution

In this section, we shall discretize Eq. (4.4) in a two dimensional space, i.e.,$\Omega = (a,b)\times(c,d)$. The numerical convolution $G_\sigma * f_0$ can be computed as following using a 3×3 smoothing kernel:

$$(G_\sigma * f_0)_{ij} = \sum_{p=i-1}^{i+1}\sum_{q=j-1}^{j+1}\frac{f_{0pq}}{2\pi\sigma^2}e^{-\frac{[(i-p)^2+(j-q)^2]h^2}{2\sigma^2}}. \tag{4.5}$$

Then, the edge function can be calculated by

$$g(f_0)_{ij} = \frac{1}{1+(G_\sigma * f_0)^2_{x,ij}+(G_\sigma * f_0)^2_{y,ij}}, \tag{4.6}$$

where $(G_\sigma * f_0)_{x,ij} = [(G_\sigma * f_0)_{i+1,j} - (G_\sigma * f_0)_{i-1,j}]/(2h)$. Then we propose the following operator splitting numerical algorithm for Eq (4.4) :

$$\frac{\phi^*_{ij}-\phi^n_{ij}}{\Delta t} = g_{ij}\Delta_d\phi^*_{ij} + \lambda g_{ij}F(\phi^n_{ij}), \tag{4.7}$$

$$\frac{\phi^{n+1}_{ij}-\phi^*_{ij}}{\Delta t} = -g_{ij}\frac{F'(\phi^{n+1}_{ij})}{\epsilon^2}, \tag{4.8}$$

If we add these two equations, then we have

$$\frac{\phi^{n+1}_{ij}-\phi^n_{ij}}{\Delta t} = g_{ij}\left(-\frac{F'(\phi^{n+1}_{ij})}{\epsilon^2} + \Delta_d\phi^*_{ij}\right) + \lambda g_{ij}F(\phi^n_{ij}). \tag{4.9}$$

The implicit discrete Eq. (4.7) can be solved by a multigrid method [153] with the initial condition ϕ^n. Since we can consider Eq. (4.8) as an approximation of the equation

$$\phi_t = g\frac{\phi - \phi^3}{\epsilon^2} \tag{4.10}$$

by an implicit Euler's method with the initial condition ϕ^*. Then the solution at $t = \Delta t$ of Eq. (6.8), solved by the method of separation of variables [143], is given as

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

Finally, our proposed scheme is written as

$$\frac{\phi_{ij}^* - \phi_{ij}^n}{\Delta t} = g_{ij}\Delta_d\phi_{ij}^* + \lambda g_{ij}F(\phi_{ij}^n), \tag{4.11}$$

$$\phi_{ij}^{n+1} = \frac{\phi_{ij}^*}{\sqrt{e^{-\frac{2g_{ij}\Delta t}{\epsilon^2}} + (\phi_{ij}^*)^2\left(1 - e^{-\frac{2g_{ij}\Delta t}{\epsilon^2}}\right)}}. \tag{4.12}$$

4.4. Computational examples

In this section, we present numerical results using the proposed numerical algorithm on various synthetic and real images. We show that a very fast and accurate minimization can be achieved by the proposed algorithm. In our numerical experiments, we normalize the given image f as $f_0 = \frac{f - f_{min}}{f_{max} - f_{min}}$, where f_{max} and f_{min} are the maximum and the minimum values of the given image, respectively. Across the interfacial regions, the phase field varies from -0.9 to 0.9 over a distance of approximately $2\sqrt{2}\epsilon\tanh^{-1}(0.9)$. Therefore, if we want this value to be approximately m grid points, then the ϵ value needs to be taken as follows:

$$\epsilon_m = \frac{hm}{2\sqrt{2}\tanh^{-1}(0.9)}.$$

To speed up simulation, we simply initialize ϕ^0 with an edge stopping function g and a given tolerance, tol. The initialization algorithm is listed as follows:

In our first example, we show that our proposed model can detect different objects. In Fig. 4.1, we show how our proposed initial algorithm works. (a) is the given image, (b), (c), (d), and (e) are steps 1, 2, 3, and 4 in the initialization algorithm, respectively. In Fig. 4.2, the computational domain is set to $\Omega = (0, 1) \times (0, 1)$ with a 64×64 mesh. Interface parameter ϵ_4, time step $\Delta t = 4$E-4, and $\lambda = 1$E4 are used. It only took 9 iterations, which is one order of magnitude smaller than the method in [98].

Fig. 4.3 shows our proposed method is faster than the previous method, which also used the Allen-Cahn equation for image segmentation [9]. The computational domain is set to $\Omega = (0, 1) \times (0, 1)$ with a 128×128 mesh. Interface parameter ϵ_8, time step $\Delta t = 6$E-4, and $\lambda = 2.5$E4 are used. It took only 20 iterations, which is one order of magnitude smaller than the previous method, which used 532 iterations.

Set a tolerance tol and $\phi^0 = -1$ everywhere in the computational domain. And take the following steps (1–4):

Step 1 for (i=1; $i \leq N_x$; i++) for (j=N_y; j$\geq$1 ; j- -) {
if ($g_{ij} < tol$) break ; else $\phi^0_{ij} = 1$;}

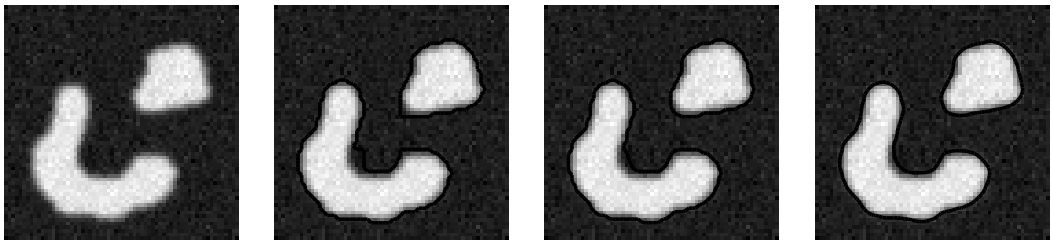
Step 2 for (j=1; $j \leq N_y$; j++) for (i=1; $i \leq N_x$; i++) {
if ($g_{ij} < tol$) break ; else $\phi^0_{ij} = 1$;}

Step 3 for (i=1; $i \leq N_x$; i++) for (j=1; $j \leq N_y$; j++) {
if ($g_{ij} < tol$) break ; else $\phi^0_{ij} = 1$;}

Step 4 for (j=1; $j \leq N_y$; j++) for (i=N_x; $i \geq 1$; i- -) {
if ($g_{ij} < tol$) break ; else $\phi^0_{ij} = 1$;}

(a) (b) (c) (d) (e)

FIGURE 4.1. Initial algorithm: (a) original image, (b) Step 1, (c) Step 2, (d) Step 3, and (e) Step 4.

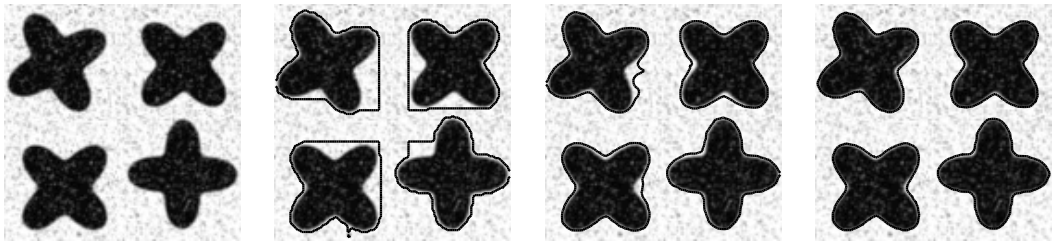


(a) (b) (c) (d)

FIGURE 4.2. (a) Original image, (b) 0 iteration, (c) 3 iterations, and (d) 9 iterations.

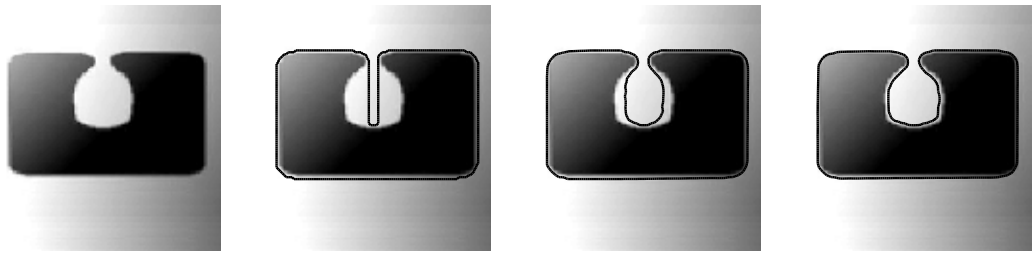
Next example image is from [70] and has varying illumination and highly concave shape (see Fig. 4.4 (a)). The computational domain is set to $\Omega = (0,1) \times (0,1)$ with a 128×128 mesh. Interface parameter ϵ_8, time step $\Delta t =$ 3E-4, and $\lambda =$ 2E4 are used. Fig. 4.4 (b) shows the initial configuration using our proposed initialization algorithm. As can be seen from Fig. 4.4 (d), image segmentation is successfully done only after 10 iterations.

In Fig. 4.5, we show a numerical result on a real medical image, a hip joint. The computational domain is set to $\Omega = (0,1) \times (0,1)$ with a 128×128 mesh. Interface parameter ϵ_8, time step $\Delta t =$ 3E-5, and $\lambda =$ 2.5E4 are used. The method produces visually clear results with only took 10 iterations. As can be seen, our proposed method has performed well in this medical image segmentation.



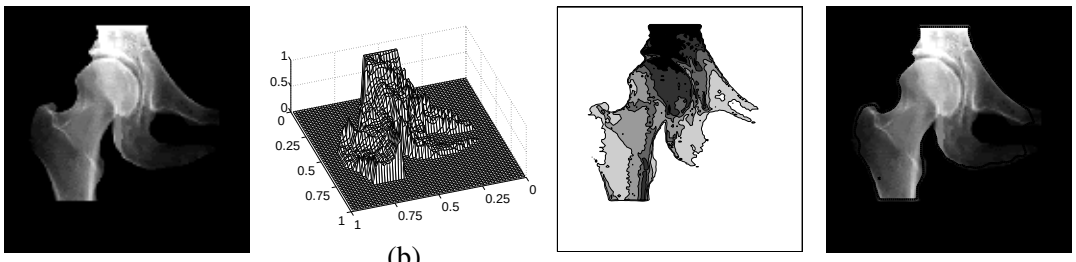
(a) (b) (c) (d)

FIGURE 4.3. (a) Original image, (b) 0 iteration, (c) 8 iterations, and (d) 20 iterations.



(a) (b) (c) (d)

FIGURE 4.4. Form left to right: Image which has varying illumination and highly concave shape, 0 iteration, 5 iterations, and 10 iterations.



(a) (b) (c) (d)

FIGURE 4.5. (a) Original image, (b) mesh image, (c) contour image (d) final iterations.

In Fig. 4.6, the segmentation of another hip joint image is shown on the computational domain $\Omega = (0,1) \times (0,1)$ with a 256×256 mesh. Interface parameter ϵ_8, time step $\Delta t =$ 5E-5, and $\lambda =$ 1.5E4 are used. As can be observed, the agreement between the area of hip and the segmentation of image is obvious.

4.5. Conclusion

We have shown that our proposed algorithm achieves faster segmentation of binary images than the previous methods. The numerical method used a fast solver such as a multigrid method for solving heat equation and an analytic solution for the nonlinear equation. To speed up image segmentation, we developed the initialization algorithm which initialized ϕ^0 with an

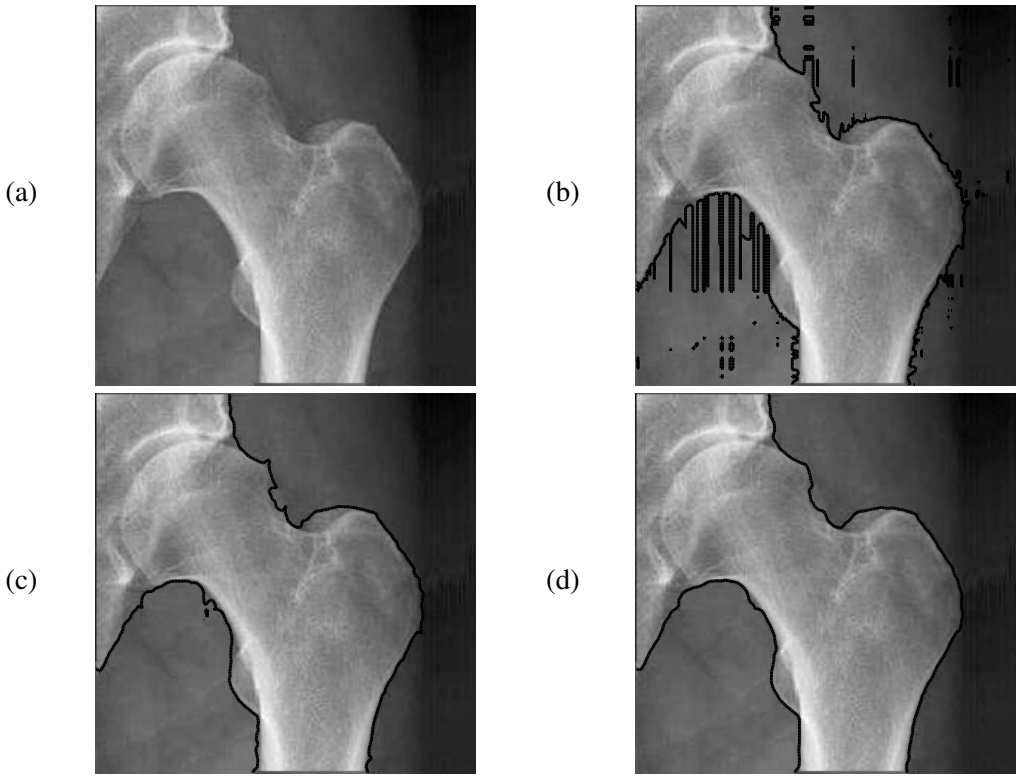


FIGURE 4.6. (a) Original image, (b) 0 iteration, (c) 4 iterations, and (d) 20 iterations.

edge stopping function g and a given tolerance, tol. We validated the proposed numerical method by various numerical results on artificial and real images.

This Chapter is published in J. KSIAM Vol 14, No. 4, pp. 201-210, 2010.

Chapter 5

Multiphase image segmentation using a phase-field model

5.1. Introduction

Image segmentation is one of the fundamental tasks in image processing and computer vision. It forms a crucial preliminary step for subsequent object recognition and interpretation [119]. Its goal is to partition a given image into regions that contain distinct objects. The most common form of segmentation is based on the assumption that distinct objects in an image have different and approximately constant colors. A natural approach is therefore to decompose an image domain into approximately homogeneous regions that are separated by sharp changes in image features.

Chan and Vese (CV) proposed a multiphase level-set framework for image segmentation using the Mumford and Shah model for piecewise constant and piecewise smooth optimal approximations [37]. Their model can segment 2^K phases of the image if K level-set functions are used. Thus, the multiphase CV model evolves more regions than necessary whenever the number of regions is not a power of two. Samson, Blanc-Feraud, Aubert, and Zerubia partitioned K phases using K level-set functions for multiphase image classification [142]. Lie, Lysaker, and Tai proposed a variant of the level-set formulation for multiphase image segmentation by introducing a piecewise constant level-set function and using each constant value to represent a unique phase [105]. In [78], Jung, Kang, and Shen proposed a phase-field method to solve the multiphase piecewise constant segmentation problem. The method is based on the phase transition model of Modica-Mortola with a sinusoidal potential and a fitting term. The proposed method is a variational partial differential equation approach that is closely connected to the Mumford-Shah model.

The objective of this Chapter is to propose a new, fast, and stable hybrid numerical method for multiphase image segmentation using a phase-field model which is based on the Allen-Cahn equation [1] with a multiple well potential and a data-fitting term. We employ the operator splitting method for the Allen-Cahn equation [99]. We split its numerical solution algorithm into a linear diffusion equation with a source term and a nonlinear equation. The linear equation is discretized using an implicit scheme and the resulting discrete system of equations is solved by a multigrid method. The nonlinear equation is solved using a closed-form solution. We also propose an initialization algorithm based on the target objects for the fast image segmentation.

This Chapter is organized as follows. In Section 5.2, the proposed model for multiphase image segmentation is given. In Section 5.3, we describe the proposed unconditionally stable hybrid operator splitting method. In Section 5.4, we perform some characteristic numerical experiments for multiphase image segmentation to demonstrate the efficiency and robustness of the proposed model and the numerical method. Finally, conclusions are given in Section 5.5.

5.2. Description of the proposed model

A multi phase-field approximation ($K+1$ phase fields) for minimizing the Mumford-Shah functional is given by the following energy functional:

$$\mathcal{E}(\phi) = \int_\Omega \left(\frac{F(\langle\phi\rangle)}{\varepsilon^2} + \frac{|\nabla\phi|^2}{2} + G_K(\phi, f_0) \right) d\mathbf{x}. \tag{5.1}$$

Here $\langle\phi\rangle = \phi - [\phi]$, where $[\phi]$ is the largest integer not greater than ϕ. $F(\phi) = 0.25\phi^2(\phi-1)^2$ is a double-well potential and $F(\langle\phi\rangle)$ is a periodic potential as shown in Fig. 5.1. ϵ is the gradient energy coefficient related to the interfacial energy and Ω is the image domain. The third term in the functional is defined as

$$G_K(\phi, f_0) = \frac{\lambda}{2} \sum_{k=0}^{K} (C_k - f_0)^2 \text{sinc}^2(\phi - k),$$

where λ is a nonnegative parameter, f_0 is the given image, and C_k is the average of f_0 in the k-level ($k = 0, 1, ..., K$), i.e.,

$$C_k = \frac{\int_\Omega f_0(\mathbf{x}) \text{sinc}^2(\phi(\mathbf{x}) - k) d\mathbf{x}}{\int_\Omega \text{sinc}^2(\phi(\mathbf{x}) - k) d\mathbf{x}}.$$

Functions $\text{sinc}(\phi) = \sin(\pi\phi)/(\pi\phi)$ and $\text{sinc}^2(\phi)$ are shown in Fig. 5.2. We note that our model is similar to that of Jung et al. [78] except that they used a sinusoidal potential while we use a periodic quartic polynomial as a potential. By using the polynomial potential, we can derive a very efficient and accurate numerical scheme based on an operator splitting technique.

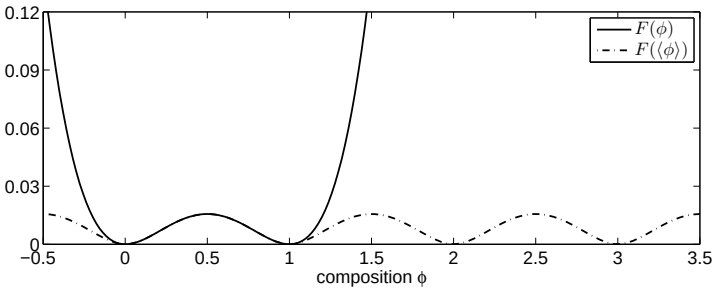


FIGURE 5.1. A double well potential, $F(\phi) = 0.25\phi^2(\phi-1)^2$ and a periodic potential, $F(\langle\phi\rangle) = 0.25\langle\phi\rangle^2(\langle\phi\rangle - 1)^2$.

For a governing equation, we seek a law of evolution in the form [34]:

$$\phi_t = -\text{grad}\mathcal{E}(\phi).$$

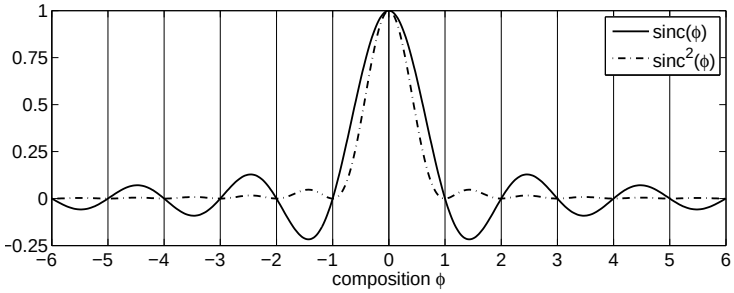


FIGURE 5.2. $\text{sinc}(\phi) = \sin(\pi\phi)/(\pi\phi)$ and $\text{sinc}^2(\phi)$.

The symbol 'grad' here denotes the gradient in the space $L^2(\Omega)$. Therefore, we get the following gradient descent flow equation:

$$\begin{aligned}\phi_t &= -\text{grad}\mathcal{E}(\phi) \\ &= -\frac{F'(\langle\phi\rangle)}{\varepsilon^2} + \Delta\phi \\ &\quad -\lambda\sum_{k=0}^{K}(C_k - f_0)^2\left(\frac{\sin(2\pi(\phi-k))}{\pi(\phi-k)^2} - \frac{2\sin^2(\pi(\phi-k))}{\pi^2(\phi-k)^3}\right).\end{aligned} \tag{5.2}$$

5.3. Description of the numerical algorithms

We propose the following operator splitting numerical algorithm.

$$\frac{\phi^{n+1}-\phi^n}{\Delta t} = -\frac{F'(\langle\phi^{n+1}\rangle)}{\epsilon^2} + \Delta_d\phi^{n+\frac{1}{2}} \tag{5.3}$$

$$-\lambda\sum_{k=0}^{K}(C_k^n - f_0)^2\left(\frac{\sin(2\pi(\phi^n-k))}{\pi(\phi^n-k)^2+\delta} - \frac{2(\phi^n-k)\sin^2(\pi(\phi^n-k))}{\pi^2(\phi^n-k)^4+\delta}\right), \tag{5.4}$$

where we added a small value δ in the denominators to avoid singularities and C_k^n is defined as following

$$C_k^n = \frac{\sum_i^{N_x}\sum_j^{N_y} f_{0,ij}\text{sinc}^2(\phi_{ij}-k)}{\sum_i^{N_x}\sum_j^{N_y}\text{sinc}^2(\phi_{ij}-k)}. \tag{5.5}$$

We take the following two steps:

$$\frac{\phi^{n+\frac{1}{2}}-\phi^{n}}{\Delta t}=\Delta_d\phi^{n+\frac{1}{2}} \tag{5.6}$$
$$-\lambda\sum_{k=0}^{K}(C_k^n-f_0)^2\left(\frac{\sin(2\pi(\phi^n-k))}{\pi(\phi^n-k)^2+\delta}-\frac{2(\phi^n-k)\sin^2(\pi(\phi^n-k))}{\pi^2(\phi^n-k)^4+\delta}\right)$$

and

$$\frac{\phi^{n+1}-\phi^{n+\frac{1}{2}}}{\Delta t}=-\frac{F'(\langle\phi^{n+1}\rangle)}{\epsilon^2}. \tag{5.7}$$

We solve Eq. (5.6) by a multigrid method [11, 153]. We can consider Eq. (5.7) as an implicit scheme for the following equation (6.8) with the initial condition $\phi^{n+\frac{1}{2}}$.

$$\phi_t=-\frac{F'(\langle\phi\rangle)}{\epsilon^2}. \tag{5.8}$$

The solution at $t=\Delta t$ of Eq. (6.8), solved by the method of separation of variables [143], is given as

$$\langle\phi^{n+1}\rangle=0.5+\frac{\langle\phi^{n+\frac{1}{2}}\rangle-0.5}{\sqrt{e^{\frac{-\Delta t}{2\epsilon^2}}+(2\langle\phi^{n+\frac{1}{2}}\rangle-1)^2(1-e^{\frac{-\Delta t}{2\epsilon^2}})}}. \tag{5.9}$$

Hence, the solution of Eq. (5.7) is

$$\phi^{n+1}=0.5+\frac{\langle\phi^{n+\frac{1}{2}}\rangle-0.5}{\sqrt{e^{\frac{-\Delta t}{2\epsilon^2}}+(2\langle\phi^{n+\frac{1}{2}}\rangle-1)^2(1-e^{\frac{-\Delta t}{2\epsilon^2}})}}+[\phi^{n+\frac{1}{2}}]. \tag{5.10}$$

Finally, our proposed scheme is written as

$$\frac{\phi^{n+\frac{1}{2}}-\phi^{n}}{\Delta t}=\Delta_d\phi^{n+\frac{1}{2}} \tag{5.11}$$
$$-\lambda\sum_{k=0}^{K}(C_k^n-f_0)^2\left(\frac{\sin(2\pi(\phi^n-k))}{\pi(\phi^n-k)^2+\delta}-\frac{2(\phi^n-k)\sin^2(\pi(\phi^n-k))}{\pi^2(\phi^n-k)^4+\delta}\right),$$
$$\phi^{n+1}=0.5+\frac{\langle\phi^{n+\frac{1}{2}}\rangle-0.5}{\sqrt{e^{\frac{-\Delta t}{2\epsilon^2}}+(2\langle\phi^{n+\frac{1}{2}}\rangle-1)^2(1-e^{\frac{-\Delta t}{2\epsilon^2}})}}+[\phi^{n+\frac{1}{2}}]. \tag{5.12}$$

The homogeneous Neumann boundary condition is applied to the domain. However, it is not restricted to the Neumann condition, we can use other boundary conditions such as periodic, Dirichlet, and combinations of these. We note that the first Eq. (5.11) is the discrete implicit diffusion equation with a source term. The second Eq. (5.12) is the analytic solution for the logistic model equation. Both steps are unconditionally stable, which means that solutions are stable regardless of time step sizes. The unconditional stability of the proposed numerical schemes will be demonstrated numerically in the next section.

5.4. Numerical experiments

In this section, we tested our proposed model and its computational algorithm on generic numerical experiments such as piecewise constant images with and without noises, landscape image, and MR images.

5.4.1. Proposed initialization. Regarding the initial guess for the phase-field ϕ, authors in [78] have typically adopted random values between -0.5 and $K + 0.5$. In our numerical experiments, we normalize a given image f as $f_0 = \frac{f-f_{min}}{f_{max}-f_{min}}$, where f_{max} and f_{min} are the maximum and the minimum values of the given image, respectively. Across the interfacial regions, the concentration field varies from 0.1 to 0.9 over a distance of approximately $4\sqrt{2}\epsilon \tanh^{-1}(0.9)$. Therefore, if we want this value to be approximately m grid points, the ϵ value needs to be taken as follows:

$$\epsilon_m = \frac{hm}{4\sqrt{2}\tanh^{-1}(0.9)}. \tag{5.13}$$

In this chapter, we used ϵ_2, ϵ_3, and ϵ_5 depending on test images and mesh sizes. We found the results with different ϵ_m values make little differences. We use same computational domain $\Omega = (0, 1) \times (0, 1)$ throughout the chapter.

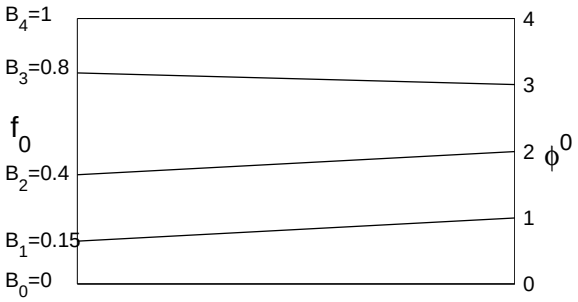
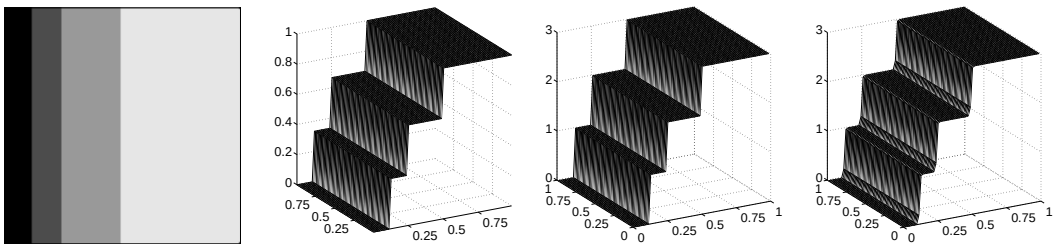


FIGURE 5.3. Proposed initialization.



(a) (b) (c) (d)

FIGURE 5.4. Multiphase image with uniform heights. (a): Original image f_0, (b): the value of original image f_0, (c): initial phase-field ϕ^0, and (d): at convergence ϕ approaches 4 constant values.

For the initial profile we propose the following:

$$\phi_{ij}^0 = k - 1 + \frac{f_{ij,0} - B_{k-1}}{B_k - B_{k-1}} \text{ if } f_{ij,0} \in [B_{k-1}, B_k] \text{ for } k = 1, \cdots, K, \tag{5.14}$$

where B_k for $k = 0, \cdots, K$ are target levels of the image. For example, if we have five target values, $B_0 = 0$, $B_1 = 0.15,\ B_2 = 0.4,\ B_3 = 0.8$, $B_4 = 1$, then we have the initial phase-field according to Eq. (5.14) and this is shown in Fig. 5.3.

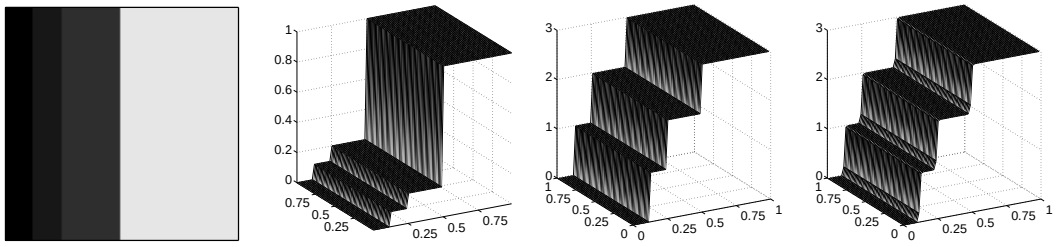


FIGURE 5.5. Multiphase image with nonuniform heights. (a): Original image f_0, (b): the value of original image f_0, (c): initial phase-field ϕ^0, and (d): at convergence ϕ approaches 4 constant values.

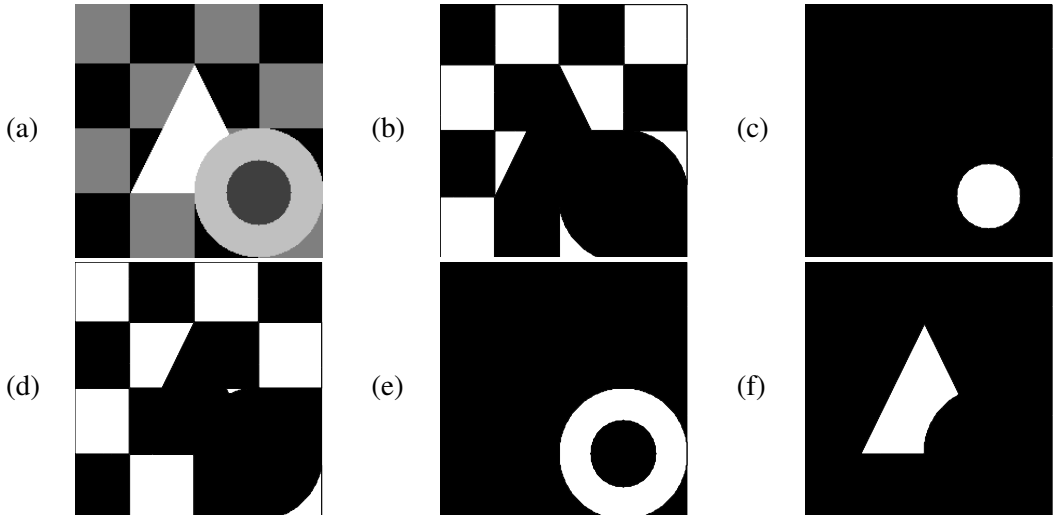


FIGURE 5.6. A complex synthetic image with multiple objects and several generic visual structures (a) Original image and (b)-(f) the field contours for each level from $k = 0$ to $k = 4$.

Fig. 5.4 shows a simple synthetic image that contains four level of colors. (a) and (b) are the original image and the value of original image f_0, respectively. (c) is the initial phase-field ϕ^0. (d) shows the converged phase-field after 4 iterations with ϵ_3, Δt = 5E-6, $h = 1/128$, $\lambda = 10$, and target values $B_0 = 0$, $B_1 = 0.3333$, $B_2 = 0.6667$, and $B_3 = 1$.

Next, we set nonuniform heights with 0, 0.1, 0.2, and 1 as shown in Fig. 5.5(a) and (b). Fig. 5.5(c) and (d) are initial profile and converged solution with four iterations. Here we used same parameters of previous test with $B_0 = 0$, $B_1 = 0.1026$, $B_2 = 0.2003$, and $B_3 = 1.0$. We can confirm that the new initialization works well with nonuniform heights also.

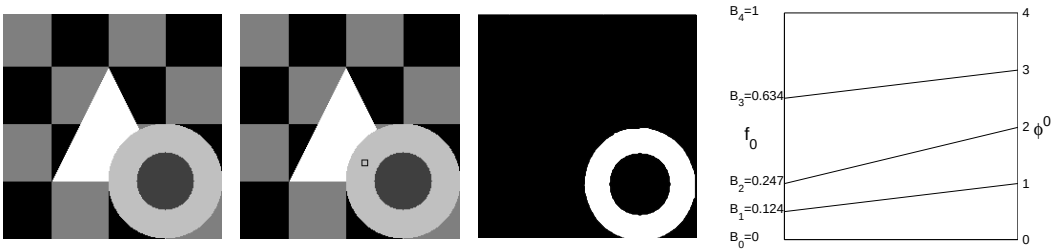


FIGURE 5.7. A complex synthetic image with multiple objects and several generic visual structures (a) original image, (b) target image area, (c) result, and (d) the proposed initialization.

5.4.2. Complex synthetic image. The next example is taken from [78]. Fig. 5.6(a) shows a complex synthetic image with multiple objects and several generic visual structures. The interface parameter ϵ_5, time step $\Delta t =$ 5E-6, $h = 1/256$, and $\lambda = 10$ are used with the initial value $B_0 = 0$, $B_1 = 0.2471$, $B_2 = 0.4980$, $B_3 = 0.7529$, and $B_4 = 1$. Fig. 5.6(b)-(f) show the filled contours for each level from $k = 0$ to $k = 4$. White regions are segmented areas.

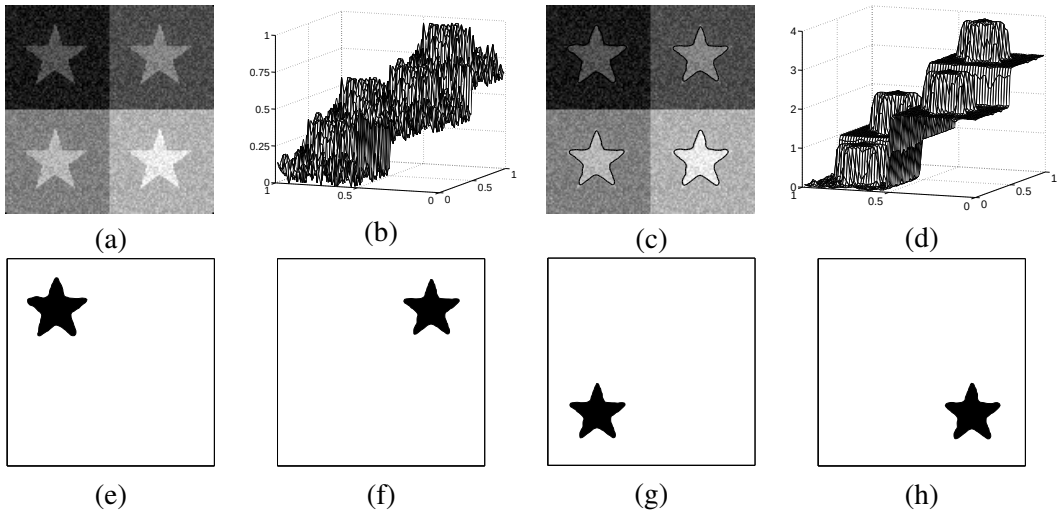
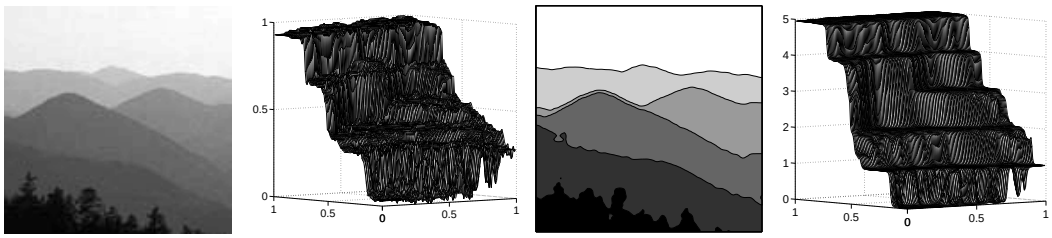


FIGURE 5.8. (a) Original image f_0, (b) the value of original image f_0, (c) final phase-field, (d) the value of the phase-field ϕ, and (e)-(h) the filled contours with $\phi = 0.5$, $\phi = 1.5$, $\phi = 2.5$, and $\phi = 3.5$, respectively.

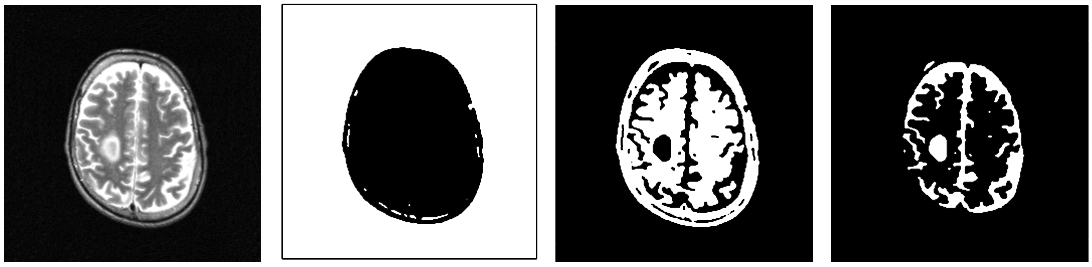
In many real applications the number of levels to detect is not known a priori. A robust and reliable algorithm should find the correct segmentation even when the exact number of

phases is not known. To see if our algorithm can handle such a case we choose only one phase among multiple phases. Fig. 5.7 shows the application of the proposed model for segmenting the region of image which we are interested. We get $B_2 = 0.2471$ and set $B_0 = 0$, $B_1 = B_0(1-tol)+B_2tol$, $B_3 = B_2(1-tol)+B_4tol$, and $B_4 = 1$, here we take a specific tolerance, $tol = 0.5$. Fig. 5.7(d) shows the initialization with these values.



(a) (b) (c) (d)

FIGURE 5.9. Landscape image segmentation. (a) Original image. (b) The value of original image. (c) Steady state filled contours. (d) The value of steady state.



(a) (b) (c) (d)

FIGURE 5.10. Segmentation of a brain MRI using a three level phase-field. (a) is a brain MRI. (b), (c), and (d) are filled contours at $\phi = 0,\ 1$, and 2, respectively.

5.4.3. Synthetic image with noise. To show the efficiency of our proposed numerical scheme we choose a typical experiment from [105]. The example is a noisy synthetic image containing 4 stars on 4 different backgrounds (see Fig. 5.8(a) and (b)). Interface parameter ϵ_3, $\Delta t =$ 4E-6, $h = 1/128$, and $\lambda = 10$ are used with $B_0 = 0.0988$, $B_1 = 0.3281$, $B_2 = 0.5257$, $B_3 = 0.7190$, and $B_4 = 0.9209$. As can be seen, the method produces visually clear results with the five phase fields after 15 iterations.

5.4.4. Real landscape image. Fig. 5.9 shows the application of the proposed model to a real landscape image. The interface parameter ϵ_3, $\Delta t =$ 4E-6, $h = 1/128$, and $\lambda = 10$ are used with $B_0 = 0.1127$, $B_1 = 0.3146$, $B_2 = 0.4128$, $B_3 = 0.5264$, $B_4 = 0.7167$, and $B_5 = 0.9643$. We show the final five segments detected by our proposed algorithm after 10 iterations (see Fig. 5.9(c) and (d)).

5.4.5. Brain MRI. Fig. 5.10 shows the application of the proposed model to a brain MRI segmentation with three levels. The interface parameter ϵ_2, Δt = 5E-6, $h = 1/256$, and $\lambda = 10$ are used with $B_0 = 0.0348$, $B_1 = 0.4286$, and $B_2 = 0.8106$. We can see how the model can handle complex topologies. Fig. 5.10(a) is a brain MRI. Numerical results (b), (c), and (d) are filled contours at $\phi = 0,\ 1$, and 2, respectively. It only took 8 iterations.

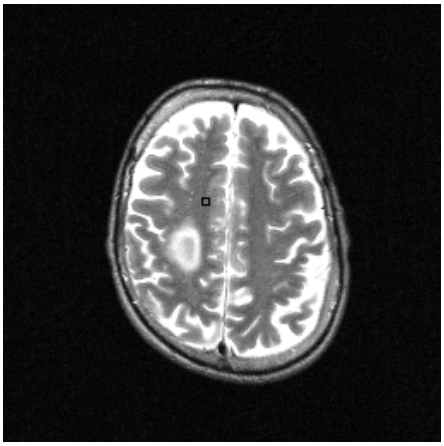
(a)  (b) 

FIGURE 5.11. Segmentation of a brain MRI using a three level phase-field. (a) is a brain MRI with a selected region. (b) is the segmentation result for the selected region.

Final experiment shows the application of the proposed model for segmenting the selected region of image. Suppose that we only want to segment the gray part (small square box) in Fig. 5.11(a). We get the mean value inside the box as $B_2 = 0.4286$. Then set $B_0 = 0$, $B_1 = B_0(1 - tol) + B_2 tol$, $B_3 = B_2(1 - tol) + B_4 tol$, and $B_4 = 1$ with $tol = 0.75$. The gray area of brain is segmented with bright color as shown the result in Fig. 5.11(b) after 10 iterations.

5.5. Conclusion

Inspired by the multiphase image segmentation via Modica-Mortola phase transition we have proposed multiphase image segmentation using a phase-field model. We have shown that our proposed algorithm achieves faster segmentation of images than the previous methods. We used a fast solver such as a multigrid method for solving heat equation and an analytic solution for the nonlinear equation. We also propose an initialization algorithm based on the target objects for the fast image segmentation. We validated the proposed numerical method by various numerical results on artificial and real images.

Chapter 6

A fast, robust, and accurate operator splitting method for phase-field simulations of crystal growth

6.1. Introduction

Crystal growth is a classical example of phase transformations from the liquid phase to the solid phase via heat transfer. In the past, to understand and simulate crystal growth, several methods have been developed including boundary integral [102, 115, 141, 149], cellular automaton [106, 163, 167, 169], front-tracking [3, 73, 75, 151, 168], level-set [27, 58, 84, 159], Monte-Carlo [130, 140], and phase-field [20, 26, 35, 44, 77, 89, 94, 93, 90, 125, 124, 134, 136, 144, 152, 161, 156, 162] methods. Among these various methods, the phase-field method are popular and widely used. Its advantage is that the explicit tracking of the interface is unnecessary by introducing an order parameter, i.e., a phase-field variable. In this Chapter, we focus the phase-field method for crystal growth problems which avoids difficulties associated with tracking the interface and computes complex crystal shapes.

We consider here the solidification of a pure substance from its supercooled melt in both two- and three-dimensional space. A great challenge in the simulation with various supercoolings is the large difference in time and length scales. In order to overcome this, many numerical methods have been proposed such as explicit [75, 93, 134, 156, 79], mixed implicit-explicit [124, 161, 162], and adaptive methods [26, 35, 125, 136, 144]. In the case of explicit methods, which are widely used, the solutions become unstable for large time steps. For this reason, in [75, 156], the authors suggested $\Delta t < h^2/(4D)$ for stability of explicit methods. Here, Δt is the time step, h is the mesh size, and D is the thermal diffusivity. In [75], the time step is also restricted to $\Delta t \leq h/(10|V_{\max}|)$, where $|V_{\max}|$ is the magnitude of the maximum value of the interface velocity. Also, in [156], the authors showed that $\Delta t = h^2/(5D_L)$ works well through numerical experiments, where $D_L = M_\phi \epsilon^2$, M_ϕ is the kinetic mobility, and ϵ is the interface energy anisotropy. Implicit methods allow relatively larger time steps, however they are computationally more expensive per step than explicit ones. Another classical method [152] is 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. The use of mesh adaptivity, which is based on the choice of a suitable time integration method, is a natural choice to overcome this problem. However adaptive technology also suffers the time step restriction and crystal growth simulation with various supercoolings is still very difficult. Therefore we need a scheme that allows the use of a sufficiently large time step without the technical limitations. In this Chapter we present a new, computationally efficient, and robust operator splitting algorithm for solving the crystal growth phase-field simulation and demonstrate the accuracy of the method by a set of representative numerical experiments.

This Chapter is organized as follows: in Section 6.2 the governing equations for crystal growth based on the phase-field method are given. In Section 6.3 we describe the computationally efficient operator splitting algorithm. In Section 6.4 we present numerical results for solving the crystal growth simulation both in 2D and 3D. Finally, conclusions are given in Section 6.5.

6.2. The phase-field model

The basic equations of the phase-field model are derived from a single Lyapounov functional [97]. We model the solidification in two and three dimensions using a standard form of phase-field equations. The model is given by:

$$\begin{aligned}
\epsilon^2(\phi)\frac{\partial \phi}{\partial t} &= \nabla \cdot (\epsilon^2(\phi)\nabla\phi) + [\phi - \lambda U(1-\phi^2)](1-\phi^2) \\
&\quad + \left(|\nabla\phi|^2\epsilon(\phi)\frac{\partial \epsilon(\phi)}{\partial \phi_x}\right)_x + \left(|\nabla\phi|^2\epsilon(\phi)\frac{\partial \epsilon(\phi)}{\partial \phi_y}\right)_y \\
&\quad + \left(|\nabla\phi|^2\epsilon(\phi)\frac{\partial \epsilon(\phi)}{\partial \phi_z}\right)_z, \qquad (6.1) \\
\frac{\partial U}{\partial t} &= D\Delta U + \frac{1}{2}\frac{\partial \phi}{\partial t}, \qquad (6.2)
\end{aligned}$$

where ϕ is the order parameter, $\epsilon(\phi)$ is the anisotropic function, λ 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, $D = \alpha\tau_0/\epsilon_0^2$, α is the thermal diffusivity, τ_0 is the characteristic time, and ϵ_0 is the characteristic length. The order parameter is defined by $\phi = 1$ in the solid phase and $\phi = -1$ in the liquid phase. The interface is defined by $\phi = 0$ and λ is given as $\lambda = D/a_2$ with $a_2 = 0.6267$ [94, 93]. For the four-fold symmetry, $\epsilon(\phi)$ is defined as:

$$\epsilon(\phi) = (1 - 3\epsilon_4)\left(1 + \frac{4\epsilon_4}{1-3\epsilon_4}\frac{\phi_x^4 + \phi_y^4 + \phi_z^4}{|\nabla\phi|^4}\right),$$

where ϵ_4 is a parameter for the anisotropy of interfacial energy.

6.3. Numerical solution

In this section, we propose a robust hybrid numerical method for crystal growth simulation. The discrete differentiation operator is $\nabla_d\phi_{ij} = (\phi_{i+1,j} - \phi_{i-1,j}, \phi_{i,j+1} - \phi_{i,j-1})/(2h)$. We then define the discrete Laplacian by $\Delta_d\phi_{ij} = (\phi_{i+1,j} + \phi_{i-1,j} - 4\phi_{ij} + \phi_{i,j+1} + \phi_{i,j-1})/h^2$.

We discretize Eqs. (6.1) and (6.2):

$$\begin{aligned}
\epsilon^2(\phi^n)\frac{\phi^{n+1}-\phi^n}{\Delta t} &= \epsilon^2(\phi^n)\Delta_d\phi^{n+1,2} + 2\epsilon(\phi^n)\nabla_d\epsilon(\phi^n)\cdot\nabla_d\phi^n \\
&\quad -F'(\phi^{n+1}) - 4\lambda U^n F(\phi^{n+1,1}) \\
&\quad + \left(|\nabla_d\phi|^2\epsilon(\phi)\frac{\partial\epsilon(\phi)}{\partial\phi_x}\right)_x^n + \left(|\nabla_d\phi|^2\epsilon(\phi)\frac{\partial\epsilon(\phi)}{\partial\phi_y}\right)_y^n, \qquad (6.3)\\
\frac{U^{n+1}-U^n}{\Delta t} &= D\Delta_d U^{n+1} + \frac{\phi^{n+1}-\phi^n}{2\Delta t}, \qquad (6.4)
\end{aligned}$$

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

$$\begin{aligned}
\epsilon^2(\phi^n)\frac{\phi^{n+1,1}-\phi^n}{\Delta t} &= 2\epsilon(\phi^n)\nabla_d\epsilon(\phi^n)\cdot\nabla_d\phi^n \qquad (6.5)\\
&\quad + \left(|\nabla_d\phi|^2\epsilon(\phi)\frac{\partial\epsilon(\phi)}{\partial\phi_x}\right)_x^n + \left(|\nabla_d\phi|^2\epsilon(\phi)\frac{\partial\epsilon(\phi)}{\partial\phi_y}\right)_y^n, \\
\epsilon^2(\phi^n)\frac{\phi^{n+1,2}-\phi^{n+1,1}}{\Delta t} &= \epsilon^2(\phi^n)\Delta_d\phi^{n+1,2} - 4\lambda U^n F(\phi^{n+1,1}), \qquad (6.6)\\
\epsilon^2(\phi^n)\frac{\phi^{n+1}-\phi^{n+1,2}}{\Delta t} &= -F'(\phi^{n+1}). \qquad (6.7)
\end{aligned}$$

In Eq. (6.5), we can simplify the following terms

$$|\nabla_d\phi|^2\frac{\partial\epsilon(\phi)}{\partial\phi_x} = \frac{16\epsilon_4\phi_x(\phi_x^2\phi_y^2-\phi_y^4)}{|\nabla_d\phi|^4},\ |\nabla_d\phi|^2\frac{\partial\epsilon(\phi)}{\partial\phi_y} = \frac{16\epsilon_4\phi_y(\phi_x^2\phi_y^2-\phi_x^4)}{|\nabla_d\phi|^4}.$$

Eq. (6.7) can be considered as an approximation of the equation

$$\phi_t = \frac{\phi-\phi^3}{\epsilon^2} \qquad (6.8)$$

by an implicit Euler's method with the initial condition $\phi^{n+1,2}$. We can solve Eq. (6.8) analytically by the method of separation of variables [143]. The solution is given as follows:

$$\phi^{n+1} = \frac{\phi^{n+1,2}}{\sqrt{e^{-\frac{2\Delta t}{\epsilon^2(\phi^n)}} + (\phi^{n+1,2})^2\left(1-e^{-\frac{2\Delta t}{\epsilon^2(\phi^n)}}\right)}}. \qquad (6.9)$$

Finally, the proposed scheme can be written as follows:

$$\epsilon(\phi^n)\frac{\phi^{n+1,1}-\phi^n}{\Delta t} = 2\epsilon(\phi^n)_x\phi^n_x + 2\epsilon(\phi^n)_y\phi^n_y \tag{6.10}$$
$$+\left(\frac{16\epsilon_4\phi_x(\phi_x^2\phi_y^2-\phi_y^4)}{|\nabla_d\phi|^4}\right)^n_x + \left(\frac{16\epsilon_4\phi_y(\phi_x^2\phi_y^2-\phi_x^4)}{|\nabla_d\phi|^4}\right)^n_y,$$

$$\epsilon^2(\phi^n)\frac{\phi^{n+1,2}-\phi^{n+1,1}}{\Delta t} = \epsilon^2(\phi^n)\Delta_d\phi^{n+1,2} - 4\lambda U^n F(\phi^{n+1,1}), \tag{6.11}$$

$$\phi^{n+1} = \frac{\phi^{n+1,2}}{\sqrt{e^{-\frac{2\Delta t}{\epsilon^2(\phi^n)}} + (\phi^{n+1,2})^2\left(1-e^{-\frac{2\Delta t}{\epsilon^2(\phi^n)}}\right)}}, \tag{6.12}$$

$$\frac{U^{n+1}-U^n}{\Delta t} = D\Delta_d U^{n+1} + \frac{\phi^{n+1}-\phi^n}{2\Delta t}. \tag{6.13}$$

Eqs. (6.11) and (6.13) can be solved by a multigrid method [18, 153].

6.3.1. Calculation of the crystal tip position and velocity. The crystal tip position and velocity are the important parameters in the phase field simulation. To calculate these parameters with a high degree of accuracy we use a method based on the quadratic polynomial approximation. For simplicity, we only describe the procedure along the y-axis since the crystal is symmetric. Let y_k be the maximum y position on the interface at each time, and the quadratic polynomial approximation be:

$$y = \alpha x^2 + \beta x + \gamma.$$

Given three points: (x_{k-1}, y_{k-1}), (x_k, y_k), and (x_{k+1}, y_{k+1}) on the interface, where one of the three y points is a maximum value along the interface points, we calculate the parameters α, β, and γ from:

$$\begin{pmatrix}\alpha\\ \beta\\ \gamma\end{pmatrix} = \begin{pmatrix} x_{k-1}^2 & x_{k-1} & 1\\ x_k^2 & x_k & 1\\ x_{k+1}^2 & x_{k+1} & 1\end{pmatrix}^{-1}\begin{pmatrix} y_{k-1}\\ y_k\\ y_{k+1}\end{pmatrix}.$$

Then using α, β, and γ, we find the tip position y_* which satisfies the following conditions:

$$\left.\frac{dy}{dx}\right|_{x_*} = 0 \text{ and } y_* = \alpha x_*^2 + \beta x_* + \gamma.$$

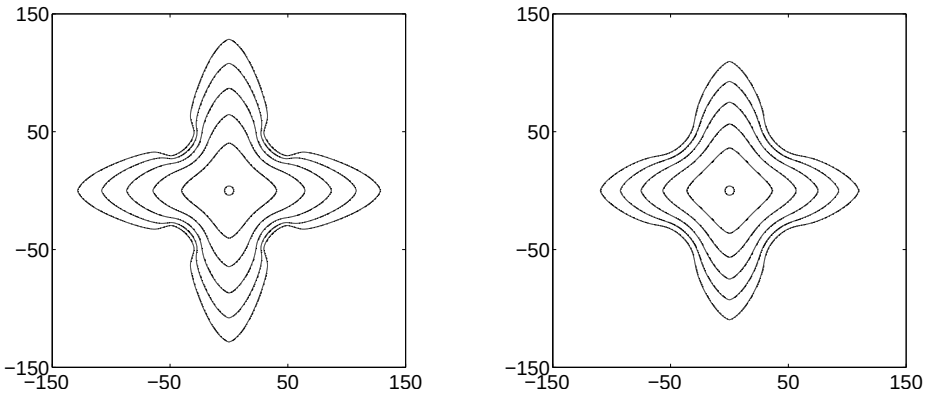
Furthermore, the crystal tip velocity can be obtained from the difference of tip positions at each time.

6.4. Numerical results

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

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

The zero level set ($\phi = 0$) 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 extension to three dimensions is straightforward. The capillary length, d_0, is defined as $d_0 = a_1/\lambda$ [20, 136, 97] with $a_1 = 0.8839$ [94, 93, 136] and $\lambda = 3.1913$ [136]. And other parameter is chosen as follows: $\epsilon_4 = 0.05$.

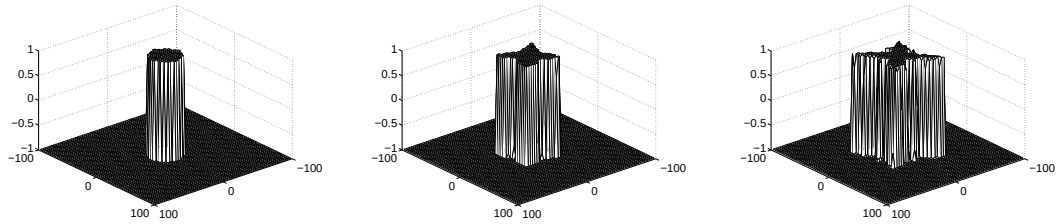


(a) $\Delta t = 0.3$ (b) $\Delta t = 0.6$

FIGURE 6.1. (a) and (b) show the sequence of interfaces with different time steps $\Delta t = 0.3$ and $\Delta t = 0.6$, respectively. The times are $t = 0$, 180, 360, 540, 720, and 900 (from inside to outside).

6.4.1. Stability of the operator splitting algorithm. As already mentioned in Section 6.1, the previous methods suffer from time step restrictions $\Delta t \leq O(h^2)$ for stability. In order to show the stability of our proposed method we consider the evolution of an interface with arbitrarily large time steps. In these simulations a 2048×2048 mesh is used on the computational domain $\Omega = (-200, 200)^2$. We choose $R_0 = 14d_0$, $\Delta = -0.55$, and $h = 0.1953$. The calculations are run up to time $T = 900$ with different time steps $\Delta t = 0.3$ and $\Delta t = 0.6$. Note that both time steps are larger than h. Figs. 6.1 (a) and (b) show evolutions of the interface with different time steps $\Delta t = 0.3$ and $\Delta t = 0.6$, respectively. In general, large time steps may cause large truncation errors, however, as can be seen in Fig. 6.1 our proposed scheme works well with large time steps.

Next, we perform a number of simulations on a set of increasingly finer grids to show that our proposed method is restricted by the stability constraint $\Delta t \leq 5.5h$. The computational domain is $\Omega = (-200, 200)^2$ and we take $R_0 = 14d_0$ and $\Delta = -0.55$. The numerical solutions are computed on the uniform grids $h = 400/2^n$ with corresponding time steps $\Delta t = 5.5h$ for $n = 8$, 9, and 10. Fig. 6.2 shows the crystal growth after time $T = 85.94$ with different time steps. From these results it is clear that our scheme is stable for time steps $\Delta t \leq 5.5h$. And we try to find the maximum Δt corresponding to different spatial grid sizes h so that stable solutions can be computed after 20 time step iterations. The results are shown in Table 6.1 and we obtain stable solutions for all three mesh sizes. Note that there is a linear relation between the time step and mesh sizes. Thus, for finer mesh sizes we may use larger time steps than previous conventional methods.



(a) $\Delta t = 8.60$ (b) $\Delta t = 4.30$ (c) $\Delta t = 2.15$

FIGURE 6.2. The stability of crystal growth with different time steps: (a) $\Delta t = 8.60$ (256×256 mesh), (b) $\Delta t = 4.30$ (512×512 mesh), and (c) $\Delta t = 2.15$ (1024×1024 mesh).

TABLE 6.1. Stability constraint of Δt for the proposed scheme.

Mesh size	$h = 400/256$	$h = 400/512$	$h = 400/1024$
Time step	$\Delta t \leq 20h$	$\Delta t \leq 15h$	$\Delta t \leq 12h$

6.4.2. Comparison of the dimensionless steady-state tip velocities. To verify the accuracy of our proposed scheme we compare the dimensionless steady-state tip velocities obtained by our proposed scheme with phase-field simulations [93] and Green's function calculations [93]. A 1024×1024 mesh is used on the domain $\Omega = (-200, 200)^2$. We choose $R_0 = 6.924$, $W_0 = 1$, and $\lambda = D/a_2$. The results are shown in Table 6.2, as can be seen the values obtained by our proposed scheme are in good agreement with results of previous phase-field and Green's theory over the whole range of d_0, Δ, and ϵ_4 investigated here. Note that despite the relatively large time step ($\Delta t = 5\Delta t^{\text{KR}} = 0.08$) used in our scheme, the results are almost identical.

TABLE 6.2. Comparison of dimensionless steady-state tip velocities calculated by our proposed scheme ($V_{\text{tip}} = Vd_0/D$), calculated by phase-field simulations ($V_{\text{tip}}^{\text{KR}}$), and calculated by the Green function method ($V_{\text{tip}}^{\text{GF}}$).

Δ	ϵ_4	D	d_0/W_0	V_{tip}	$V_{\text{tip}}^{\text{KR}}$	$V_{\text{tip}}^{\text{GF}}$
0.65	0.05	1	0.554	0.0470	0.0465	0.0469
0.55	0.05	2	0.277	0.0171	0.0168	0.0170
0.55	0.05	3	0.185	0.0174	0.0175	0.0170
0.55	0.05	4	0.139	0.0172	0.0174	0.0170
0.50	0.05	3	0.185	0.01030	0.01005	0.00985
0.45	0.05	3	0.185	0.00599	0.00557	0.00545
0.45	0.05	4	0.139	0.00598	0.00540	0.00545

6.4.3. Effect of time step, mesh, radius, and undercooling. In the first experiment we consider the evolution of the interface with different time steps in order to investigate the effect

of time step. A 1024×1024 mesh is used on the domain $\Omega = (-400, 400)^2$ and we take $h = 0.7813$, $R_0 = 14d_0$, and $\Delta = -0.55$. The simulations are run up to time $T = 1800$. Figs. 6.3 (a) and (b) show the position and velocity of the tip versus time respectively, both for different time steps $\Delta t = 0.6$, 0.3, 0.15, and 0.075. Fig. 6.3 (c) shows evolutions of the interface with time step $\Delta t = 0.15$ at times $t = 0$, 225, 450, 675, 900, 1125, 1350, 1575, and 1800 (from inside to outside). For different time steps, the interfaces at time $T = 1800$ are shown in Fig. 6.3 (d). The velocity of the tip at time $T = 1800$ versus time step is shown in Fig. 6.4. The results suggest that the velocity of the tip is linear in the time step.

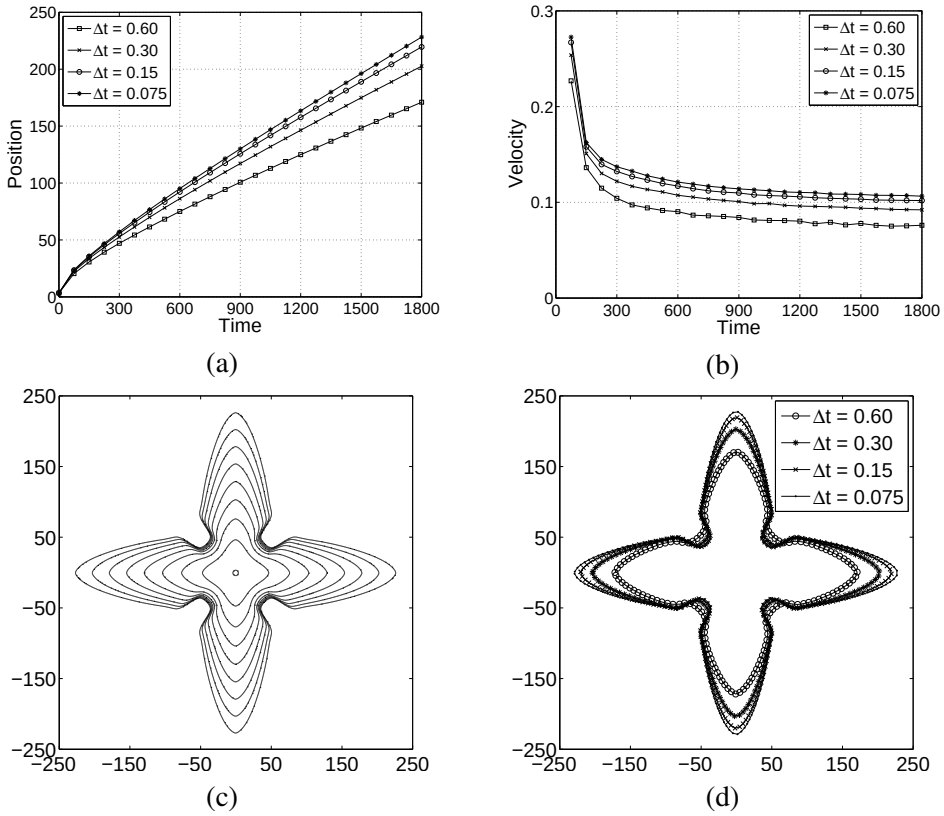


FIGURE 6.3. (a) and (b) show the position and velocity of the tip versus time respectively, for different time steps. (c) Evolutions of the interface with time step $\Delta t = 0.15$. (d) The interfaces at time $T = 1800$ for different time steps.

Total CPU times and average CPU times ($\overline{\text{CPU}}$) of the simulations for different time steps are listed in Table 6.3. The average CPU time is defined as the real computational time (excluding data printing times) divided by the total number of iterations, the results are shown in Table 6.3, corresponding to data in Fig. 6.3. Table 6.3 suggests that our proposed scheme is accurate and robust for different time steps.

In the second experiment we consider the evolution of the interface with different mesh sizes in order to find an effective mesh size for our proposed method. 256×256, 512×512, 1024×1024, and 2048×2048 meshes are used on the domain $\Omega = (-200, 200)^2$, i.e., we use

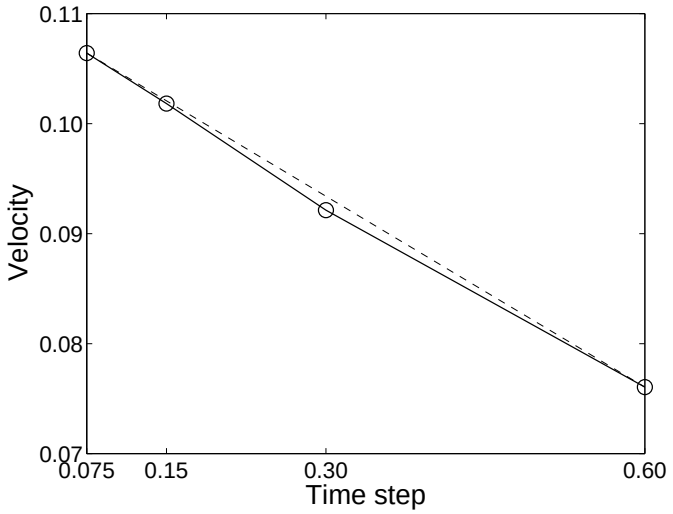


FIGURE 6.4. The final velocity of the tip versus time step.

TABLE 6.3. Total CPU times and average CPU times ($\overline{\text{CPU}}$) for different time steps.

Case	$\Delta t = 0.6$	$\Delta t = 0.3$	$\Delta t = 0.15$	$\Delta t = 0.075$
CPU time (h)	5.07	9.06	16.77	32.59
$\overline{\text{CPU}}$ time (s)	5.87	5.19	4.80	4.84

four different $h = 1.5626, 0.7813, 0.3906$, and 0.1953. The parameters used are $R_0 = 14d_0$, $\Delta = -0.55$, $\Delta t = 0.15$, and $T = 900$. Figs. 6.5 (a), (b), (c), and (d) show sequences of interface for different mesh sizes. The position and velocity of the tip versus time are shown in Fig. 6.5 (e) and (f), respectively. From the results shown in Fig. 6.5 we can observe that the spatial step size $h = 0.3906$ is enough to simulate accurately and robustly the evolution of crystal growth in our proposed method.

In the third experiment we investigate the effects of radius and undercooling of the initial solid seed. For each test a 1024×1024 mesh is used on the domain $\Omega = (-400, 400)^2$ and we choose $\Delta t = 0.15$ and $T = 1500$. The top row of Fig. 6.6 shows sequences of interfaces with different radii $R_0 = 15d_0, 50d_0$, and $100d_0$ (from left to right). In this test we take $\Delta = -0.55$. From the top row of Fig. 6.6 we can see that for an increase in the initial radius the dendrite grows faster up to the fourfold symmetry. Sequences of interfaces with different undercooling sizes $\Delta = -0.45$, $\Delta = -0.55$, and $\Delta = -0.65$ are presented in the bottom row of Fig. 6.6. In this test we take $R_0 = 14d_0$. From the bottom row of Fig. 6.6 we observe that the large initial undercooling causes the dendrite to grow faster.

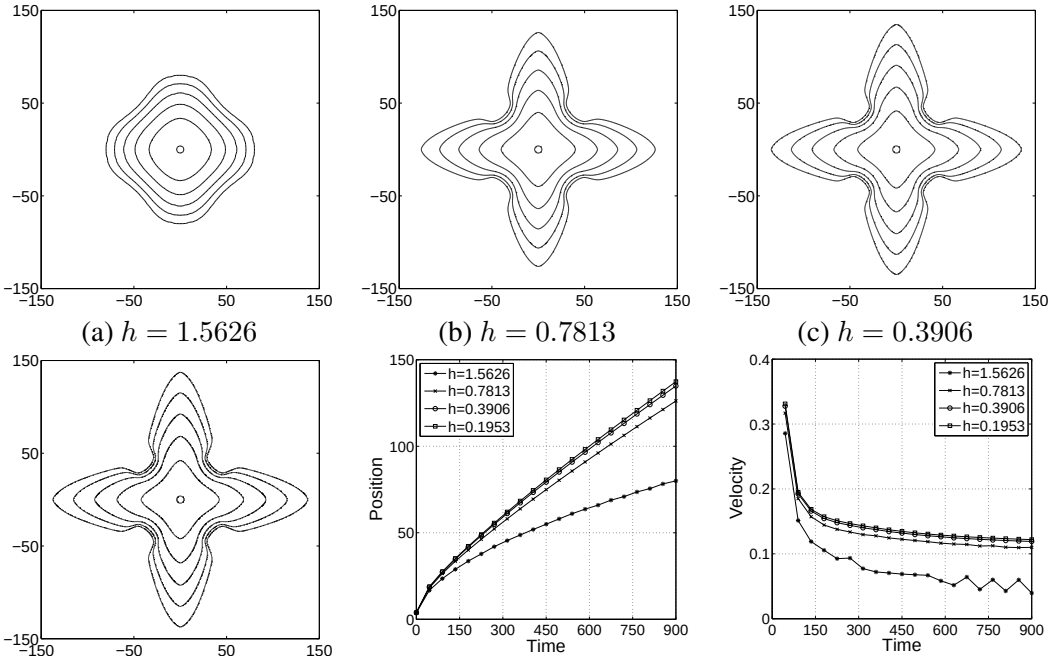


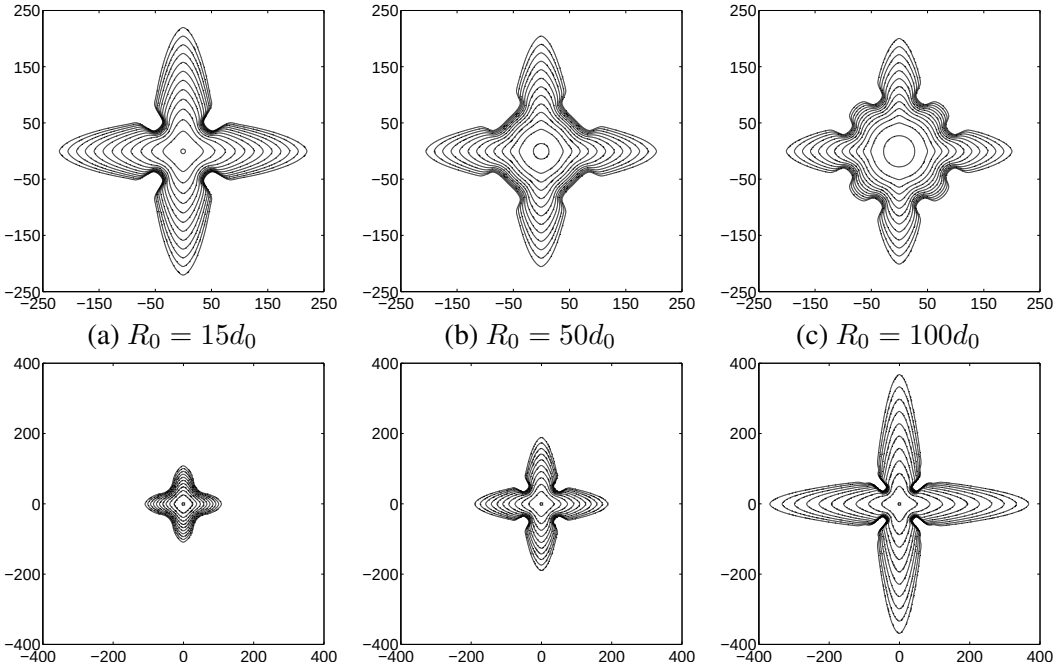
FIGURE 6.5. Sequences of interfaces with different spatial step sizes: (a) $h = 1.5626$, (b) $h = 0.7813$, (c) $h = 0.3906$, and (d) $h = 0.1953$. (e) and (f) show the position and velocity of the tip versus time, respectively.

6.4.4. Three-dimensional crystal growth. In this section we consider a three-dimensional crystal growth. The initial conditions are:

$$\begin{aligned} \phi(x,y,z,0) &= \tanh\left(\frac{R_0 - \sqrt{x^2+y^2+z^2}}{\sqrt{2}}\right), \\ U(x,y,z,0) &= \begin{cases} 0 & \text{if } \phi > 0 \\ \Delta & \text{else} \end{cases} \end{aligned}$$

on the domain $\Omega = (-100, 100)^3$ with a mesh $256 \times 256 \times 256$. The simulation parameters are $R_0 = 14d_0$, $\Delta = -0.55$, $\Delta t = 0.15$, and $T = 270$. Fig. 6.7 shows three-dimensional structures at different times. Structures with different undercooling sizes $\Delta = -0.45$, $\Delta = -0.55$, and $\Delta = -0.65$ at time $T = 200$ are presented in Fig. 6.8 (a), (b), and (c), respectively. As Figs. 6.7 and 6.8 show, our proposed scheme can straightforwardly deal with three-dimensional crystal growth.

6.4.5. Tail morphology. Brener [5] derived a theory of the tail shape of a 3D needle crystal with the assumption that the cross section of a 3D needle crystal should grow as the time dependent 2D growth shapes away from the tip. In [93], Karma and Rappel compared the steady-state growth velocities from simulation and theory derived by Brener.



(a) $R_0 = 15d_0$ (b) $R_0 = 50d_0$ (c) $R_0 = 100d_0$

(d) $\Delta = -0.45$ (e) $\Delta = -0.55$ (f) $\Delta = -0.65$

FIGURE 6.6. Sequences of interfaces with different initial parameters. Top: evolutions of the dendrite with $R_0 = 15d_0$, $50d_0$, and $100d_0$. Bottom: evolutions of the dendrite with $\Delta = -0.45$, $\Delta = -0.55$, and $\Delta = -0.65$.

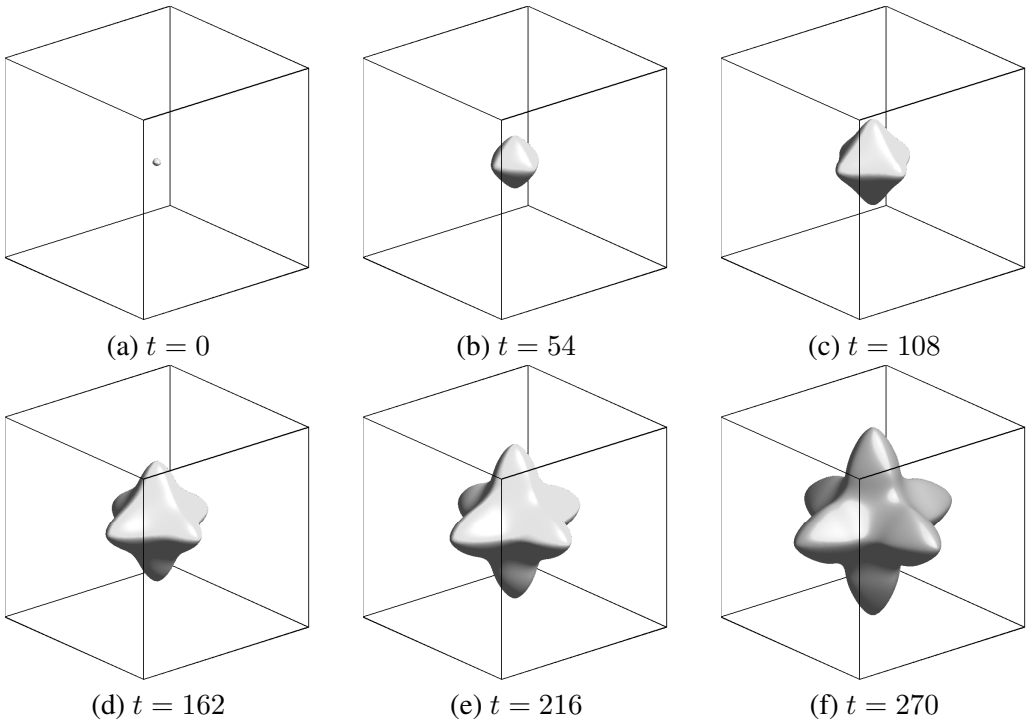
In this section we compare the velocities calculated by our scheme and those given in [93]. In 2D and 3D simulations, we choose $h = 0.3906$, $R_0 = 14d_0$, $\Delta t = 0.15$, and two different undercoolings $\Delta = -0.65$ and $\Delta = -0.70$. In the 2D test a 1024×1024 mesh is used on the domain $\Omega = (-200, 200)^2$ and the simulation time is $T = 750$. In the 3D test a $256 \times 256 \times 256$ mesh is used on the domain $\Omega = (-50, 50)^3$ and the simulation time is $T = 90$. Results of steady-state growth velocities obtained from 2D and 3D simulations are given in Table 6.4. Our results show good agreement with those of Karma and Rappel.

TABLE 6.4. Results of steady-state growth velocities.

Δ	ϵ_4	V_{2D}	V_{3D}	V_{2D}/V_{3D}	V_{2D}^{KR}/V_{3D}^{KR}	Slope
-0.70	0.0294	0.0353	0.0813	0.434	0.44	0.43
-0.65	0.0294	0.0243	0.0620	0.392	0.39	0.40

6.5. Conclusion

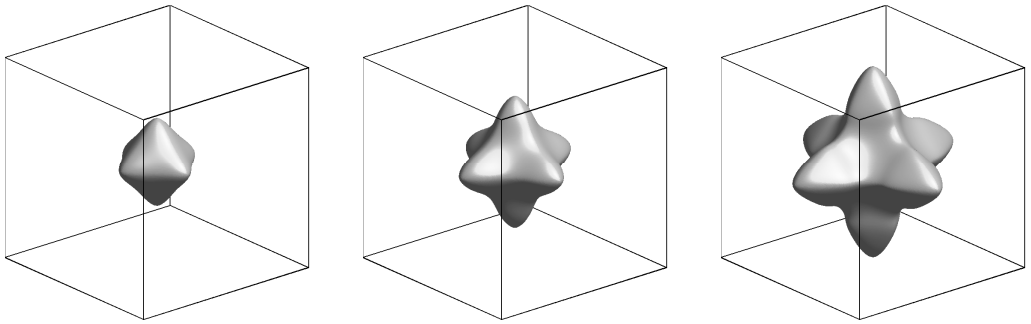
In this Chapter we proposed a fast, robust, and accurate operator splitting method for crystal growth phase-field simulation of dendritic growth in both two- and three-dimensional space.



(a) $t = 0$ (b) $t = 54$ (c) $t = 108$

(d) $t = 162$ (e) $t = 216$ (f) $t = 270$

FIGURE 6.7. Three-dimensional structures with $R_0 = 14d_0$ and $\Delta = -0.55$ at different times. The times are shown below each figure.



(a) $\Delta = -0.45$ (b) $\Delta = -0.55$ (c) $\Delta = -0.65$

FIGURE 6.8. Structures with different undercooling sizes (a) $\Delta = -0.45$, (b) $\Delta = -0.55$, and (c) $\Delta = -0.65$ at time $T = 200$.

The proposed method is based on operator splitting techniques. We split the governing phase-field equation into three parts. The first equation is calculated explicitly, the second is a heat equation with source term and is solved by a fast solver such as a multigrid method, and the third is evaluated using a closed form solution. We also presented a set of representative numerical experiments for crystal simulation to demonstrate the accuracy of the proposed method. Our simulation results are also consistent with previous numerical experiments.

Chapter 7

Numerical studies of the fingering phenomena for the thin film equation

7.1. Introduction

Thin coating flows are of great technical, scientific, and industrial interest. Coating processes normally require an external driving force such as spinning to spread a liquid film along a solid substrate. For situations in which the substrate cannot be moved, surface forces can be manipulated to drive the spreading process. For example, thermal gradients provide a way to direct thin films into small crevices requiring lubrication. A liquid film supported on a substrate subject to a thermal gradient will experience a varying surface tension depending on the local temperature since colder regions maintain a higher surface tension than the warmer regions. This thermally induced Marangoni stress will force the liquid to spread [96]. In many situations, for example, spin coating process [120, 148], the fronts become unstable, leading to the formation of fingerlike patterns, which is undesirable in technological applications. From a more fundamental point of view, one wishes to understand these strongly non-linear fingering dynamics [43]. Thin film flows have been extensively studied experimentally [16, 139], analytically [96, 6, 86, 143, 150], and numerically [88, 147, 69, 30, 156, 114, 42, 62, 63, 64, 110, 109, 145, 155, 81, 95]. Karlsen and Lie [88] proposed an unconditionally stable scheme based on operator splitting combined with a front tracking method for a class of nonlinear parabolic equations. Sellier and Panda [147] described a first-order PDE with non-constant coefficients, which involves fourth-order derivatives of the desired free surface profile and solved the first-order PDE using the method of characteristics. Ha et al. [69] presented a comparison of numerical schemes (Crank-Nicolson, fully implicit, Godunov, adapted upwind, and WENO schemes) for the convection term of a fourth order thin film equation. An alternating direction implicit (ADI) scheme, which split the n-dimensional problem into n one-dimensional implicit problems, has been used to solve the thin film equation [30, 156, 114]. This approach is a more popular alternative to explicit one which is extremely restrictive in the choice of time step. Witelski and Bowen [156] constructed an ADI scheme for the solution of two-dimensional higher-order linear and nonlinear diffusion equations, particularly including the fourth-order thin film equation for surface tension driven fluid flows. Myers et al. [114] solved the flow of a thin film, with and without solidification, on an arbitrary three-dimensional substrate by combining an ADI scheme with a shock capturing method. Recently, a multigrid approach has been developed as a more robust and efficient alternative to ADI scheme. Daniels et al. [42] showed that a fully implicit multigrid solver is more robust, returns an order of magnitude improvement in the rate of convergence, and requires low memory. Gaskell et al. applied a multigrid approach to droplet spreading flows [62] and continuous film flows with [63] or without [64] evaporation. Lee et al. [110] solved a thin film flow over a plane containing well-defined single

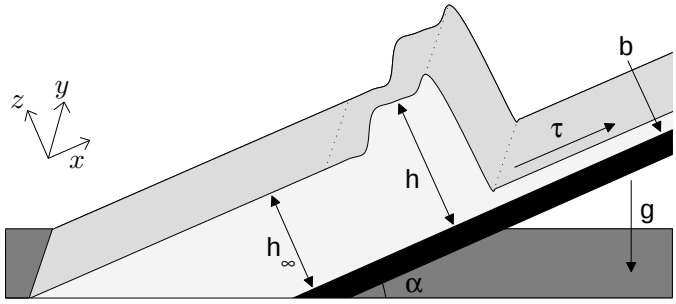


FIGURE 7.1. A schematic diagram of the physical problem. A thin layer of thickness h on an inclined surface driven by Marangoni stresses created by a temperature gradient on the plane. Gravity works against the stress to drive fluid back down the plane.

and grouped topographic features using a full approximation storage (FAS) multigrid algorithm by employing automatic mesh adaptivity. Also the authors presented FILMPAR which is a parallel multigrid algorithm for solving a three-dimensional gravity-driven continuous thin film free-surface flow over substrates containing micro-scale topography in [109]. Sellier et al. [145] used the multigrid and COMSOL solvers to solve the flow of thin liquid films on a plane surface containing occlusions. Veremieiev et al. [155] modelled an inertial thin film flow on inclined planar surfaces featuring topography via a depth-averaged form and solved using an FAS algorithm and a full multigrid (FMG) technique. Kim [81] developed an adaptive finite difference method for a class of fully nonlinear time-dependent thin liquid film equations. Kim and Sur [95] constructed a hybrid scheme which combines ENO scheme for treating convection term and a nonlinear multigrid method for the diffusion term. For more details on the thin film equation, see the review Chapter [28] and references therein. In this Chapter, we present a new interpretation of the mechanism of fingering phenomena by splitting the thin film equation into two parts: the advection part and diffusion part. It is the aim of this Chapter to investigate the mechanism of the fingering formation of the thin liquid film layer through numerical experiments.

This Chapter is organized as follows. In Section 7.2, we review the governing equation. In Section 7.3, the fully discrete, nonlinear FAS multigrid scheme for the thin film equation is given. In Section 7.4, we present numerical results. Conclusions are made in Section 7.5.

7.2. Governing 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 configuration is shown in Fig. 7.1. The spatial variables x and y denote the direction of the flow and the direction normal to the flow, respectively. Let α, ρ, g, η, γ, 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 [28, 17].

We model the dynamics of the draining film using the lubrication approximation with a "depth averaged" velocity $\bar{\mathbf{u}} = (\bar{u}, \bar{v})$ [16, 155]:

$$\begin{aligned}
\bar{u} &= \frac{1}{h}\int_0^h u dz = \frac{\tau h}{2\eta} - \frac{\rho g h^2 \sin\alpha}{3\eta} + \frac{\gamma h^2 \Delta h_x}{3\eta} - \frac{\rho g h^2 h_x \cos\alpha}{3\eta}, \\
\bar{v} &= \frac{1}{h}\int_0^h v dz = \frac{\gamma h^2 \Delta h_y}{3\eta} - \frac{\rho g h^2 h_y \cos\alpha}{3\eta}, \\
\bar{\mathbf{u}} &= \left(\frac{\tau h}{2\eta} - \frac{\rho g h^2 \sin\alpha}{3\eta}\right)\vec{e}_x + \frac{\gamma h^2 \nabla\Delta h}{3\eta} - \frac{\rho g h^2 \nabla h \cos\alpha}{3\eta},
\end{aligned} \tag{7.1}$$

where $\vec{e}_x = (1, 0)$ and $\Delta = \nabla\cdot\nabla$ is the Laplacian operator. For example, from the experimental setting in [139], the surface tension gradient τ is taken 0.11 Pa. In Eq. (7.1), 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 Eq. (7.1) with mass conservation, we obtain

$$h_t + \nabla \cdot (h\bar{\mathbf{u}}) = 0. \tag{7.2}$$

To non-dimensionalize Eq. (7.2), we employ the non-dimensional variables (denoted by hats)

$$h = H\hat{h},\ x = L\hat{x},\ y = L\hat{y},\ \text{and } t = T\hat{t}$$

to obtain

$$\begin{aligned}
\frac{H}{T}\hat{h}_{\hat{t}} &+ \frac{1}{L}\hat{\nabla}\cdot\left[\left(\frac{H^2\tau\hat{h}^2}{2\eta} - \frac{H^3\rho g\hat{h}^3\sin\alpha}{3\eta}\right)\vec{e}_x\right. \\
&\left. + \frac{H^4\gamma\hat{h}^3\hat{\nabla}\hat{\Delta}\hat{h}}{3L^3\eta} - \frac{H^4\rho g\hat{h}^3\hat{\nabla}\hat{h}\cos\alpha}{3L\eta}\right] = 0.
\end{aligned} \tag{7.3}$$

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

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

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

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

Now choose the time scale T so that gravity, Marangoni, and surface tension forces balance:

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

We substitute the expressions for H, L, and T into Eq. (7.3) and drop the '$\hat{}$' to obtain the dimensionless 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 $D = \frac{TH^3\rho g\cos\alpha}{3\eta L^2}$. Since an inclined plane is close vertical to the surface, we take $D = 0$. Thus we have derived the dimensionless thin film equation

$$h_t + (h^2 - h^3)_x = -\nabla \cdot (h^3 \nabla \Delta h). \tag{7.4}$$

This equation is a fourth order nonlinear singular perturbation of the conservative law $h_t + (h^2 - h^3)_x = 0$ [17].

7.3. Numerical Method

Firstly, we split the fourth order Eq.(7.4) into a system of second order equations

$$\begin{aligned} h_t + f_x(h) &= \nabla \cdot (M(h)\nabla\mu),\ h = h(x,y,t), & (7.5)\\ \mu &= -\Delta h,\ (x,y) \in \Omega = (0, L_x) \times (0, L_y),\ t > 0, & (7.6) \end{aligned}$$

where $f(h) = h^2 - h^3$ and $M(h) = h^3$. Boundary conditions are given by

$$\begin{aligned} h(0,y,t) &= h_\infty,\ h(L_x,y,t) = b,\ h(x,0,t) = h(x,L_y,t),\\ \mu_x(0,y,t) &= \mu_x(L_x,y,t) = 0,\ \mu(x,0,t) = \mu(x,L_y,t), \end{aligned}$$

where h_∞ is a constant upstream height and b is a precursor film thickness.

7.3.1. Discretization of the proposed scheme. Now we present fully discrete schemes for the Eqs.(7.5) and (7.6) in two dimensional space. A semi-implicit time and centered difference space discretization of Eqs. (7.5) and (7.6) is

$$\begin{aligned} \frac{h_{ij}^{n+1} - h_{ij}^n}{\Delta t} &= \nabla_d \cdot \left(M(h)_{ij}^{n+1} \nabla_d \mu_{ij}^{n+1} \right) - f_x(h_{ij}^{n+1}), & (7.7)\\ \mu_{ij}^{n+1} &= -\Delta_d h_{ij}^{n+1}. & (7.8) \end{aligned}$$

$f_x(h_{ij}^{n+1})$ is treated by using an implicit essentially non-oscillatory (ENO) type scheme [146]. Since $f'(h) = 2h - 3h^2 > 0$ if $0 < h < \frac{2}{3}$, we define

$$f_x(h_{ij}^{n+1}) := f'(h_{ij}^{n+1}) \left(\frac{h_{ij}^{n+1} - h_{i-1,j}^{n+1}}{\Delta x} \right) + \Im(h_{ij}^n),$$

where $\Im(h^n)$ is computed as follows:

$$a = \frac{h_{ij}^n - h_{i-1,j}^n}{\Delta x}\ ,\ c = \frac{h_{i+1,j}^n - h_{ij}^n}{\Delta x}\ ,\ d_{ij} = \begin{cases} a & \text{if } |a| \le |c| \\ c & \text{otherwise.} \end{cases}$$

Then we have $\Im(h_{ij}^n) = 0.5(d_{ij} - d_{i-1,j}) f'(h_{ij}^n)$. We define the boundary condition as

$$\begin{aligned} &h_{0,j} = 2h_\infty - h_{1,j},\ h_{N_x+1,j} = 2b - h_{N_x,j},\ h_{i,0} = h_{i,N_y},\ h_{i,N_y+1} = h_{i,1},\\ &\mu_{0,j} = \mu_{1,j},\ \mu_{N_x+1,j} = \mu_{N_x,j},\ \mu_{i,0} = \mu_{i,N_y},\ \mu_{i,N_y+1} = \mu_{i,1}. \end{aligned}$$

We use a nonlinear Full Approximation Storage (FAS) multigrid method to solve the nonlinear discrete system (7.7) and (7.8) at the implicit time level. A pointwise Gauss-Seidel relaxation scheme is used as the smoother in the multigrid method. See the reference text [153] for additional details and background. For a detailed description of the algorithm of the nonlinear multigrid method for solving the discrete system, please refer to Refs. [81, 95, 110].

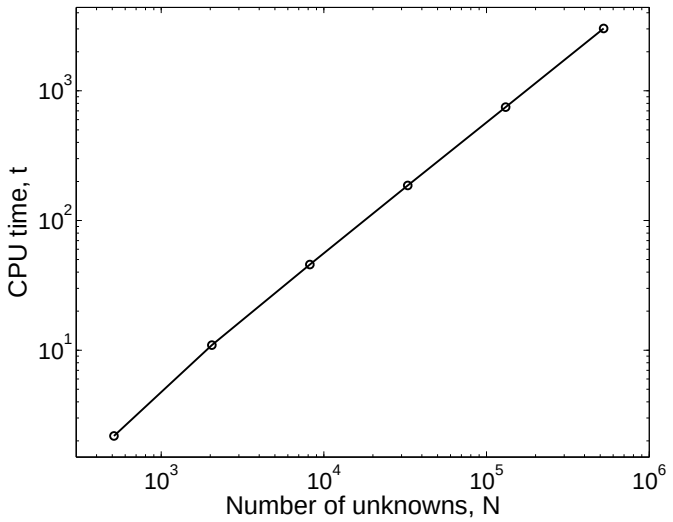


FIGURE 7.2. CPU time versus the number of unknowns. The multigrid solver achieves the $O(N)$ efficiency, where N is the number of unknowns.

7.4. Numerical Experiment

In this subsection, we perform numerical experiments such as efficiency and performance of multigrid solver, comparison of different advection schemes, role of the diffusion term in the thin film equation without the advection term, effect of h_∞, fingering instability, comparison with the experimental data, and long time evolution for the Marangoni-driven flow.

7.4.1. Efficiency of the multigrid solver. To investigate the efficiency of the multigrid method, we measure CPU times needed to solve the following problem:

$$h(x, y, 0) = 0.5[h_\infty + b - (h_\infty - b)\tanh(3(x - 5) + \text{rand}(x, y))],$$

where $h_\infty = 0.175$ and $b = 0.002$. $2^n \times 2^{n-1}$ meshes are used on $\Omega = (0, 5{\cdot}2^{n-3}) \times (0, 5{\cdot}2^{n-4})$ for $n = 5,\ 6,\ 7,\ 8,\ 9,$ and 10. Each calculation is run up to time $T = 50$ with a time step $\Delta t = 0.5$. Fig. 7.2 shows CPU times versus the number of unknowns. The results fit a straight line. This implies that the multigrid solver achieves the $O(N)$ efficiency, where N is the number of unknowns.

7.4.2. Performance of smoothers and grid levels in the multigrid solver. In the multigrid method, a robust smoother is a necessary requirement to achieve grid independent convergence. We compare the performance of Jacobi, Red-Black, and Gauss-Seidel relaxations as a smoother. The initial condition is

$$h(x, y, 0) = 0.5[h_\infty + b - (h_\infty - b)\tanh(3(x - 2) + \text{rand}(x, y))], \tag{7.9}$$

where $h_\infty = 0.225$ and $b = 0.002$. A 64×64 mesh is used on $\Omega = (0, 10) \times (0, 10)$. Solutions are computed up to time $T = 10$ with a time step $\Delta t = 0.25$. Each iteration is run until the maximum error is less than 10^{-8}. Table 7.1 shows CPU times for three different relaxation schemes. We can see from Table 7.1 that the Gauss-Seidel method is the most efficient smoother.

TABLE 7.1. CPU times for three different relaxation schemes.

Case	Jacobi	Red-Black	Gauss-Seidel
CPU time (s)	201.80	17.67	16.49

Next, we compare the performance of coarse grid levels in the multigrid solver. The initial condition is given by Eq. (7.9). A 128×128 mesh is used on $\Omega = (0, 10) \times (0, 10)$. We take $h_\infty = 0.225$, $b = 0.002$, $\Delta t = 0.25$, and $T = 10$. In this test, the experiment is performed using $(3, 3)$, $(4, 4)$, $(5, 5)$, and $(6, 6)$ V-cycles, where (ν_1, ν_2) indicates ν_1 presmoothing and ν_2 postsmoothing iterations. Note that we use Gauss-Seidel relaxation in each V-cycle and each iteration is run until the maximum error is less than 10^{-8}. Table 7.2 shows CPU times for different coarse grid levels. With increasing number of grid levels from $V(4, 4)$ to $V(6, 6)$, we get only a slight increase of the CPU time. But, in the case of $V(3, 3)$, many smoothings are required since the coarsest grid is not sufficiently coarse to solve the problem. This causes a dramatic increase in the CPU time. Therefore we need sufficiently coarse grid levels to calculate fast.

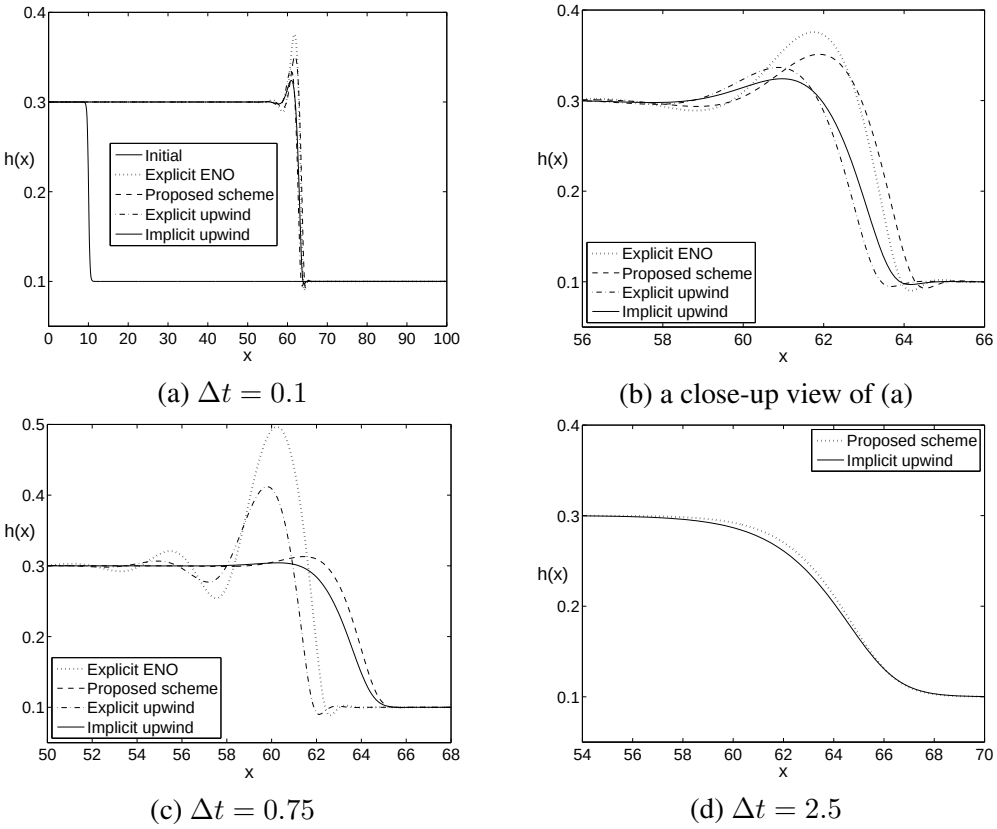
TABLE 7.2. CPU times for different coarse grid levels.

Case	V(3,3)	V(4,4)	V(5,5)	V(6,6)
CPU time (s)	109.516	71.60	72.14	72.31

7.4.3. Stability test: comparison of different advection schemes. Now, we test the stability of proposed scheme with explicit ENO, explicit upwind, and implicit upwind schemes for the advection term. For the diffusion term, we use the same implicit scheme. The initial condition in one dimension is

$$h(x, 0) = 0.5[h_\infty + b - (h_\infty - b)\tanh(3(x - 10))],$$

where $h_\infty = 0.3$ and $b = 0.1$ (see the thick solid line in Fig. 7.3(a)). A 1024 grid is used on $\Omega = (0, 100)$. Solutions are computed up to time $T = 200$ with different time steps. Fig. 7.3 shows the stability of four different advection schemes. Fig. 7.3(a) shows numerical solutions with four different schemes using a time step $\Delta t = 0.1$. In Fig. 7.3(b) (the close-up view of Fig. 7.3(a)), the dotted, dashed, dash-dot, and solid lines are results with explicit ENO, our proposed, explicit upwind, and implicit upwind schemes, respectively. As can be seen, all schemes seem well to describe the simulation of thin film by using a small time step. While, Fig. 7.3(c) shows numerical solutions with four different schemes using a slightly larger time step $\Delta t = 0.75$. This result shows that the explicit ENO and explicit upwind schemes are unstable. Our proposed and implicit upwind schemes are stable. However, in the case of the implicit upwind scheme, the hump disappeared due to the numerical diffusion. Fig. 7.3(d) shows numerical solutions with two different schemes using a time step $\Delta t = 2.5$. The dotted and solid lines are results with our proposed and implicit upwind schemes, respectively. This



(a) $\Delta t = 0.1$ (b) a close-up view of (a)

(c) $\Delta t = 0.75$ (d) $\Delta t = 2.5$

FIGURE 7.3. The evolutions with three different time steps. (a) and (b) Using a small time step, all schemes are stable. (c) Using a slightly larger time step, only our proposed and implicit upwind schemes are stable. (d) However, in the case of the implicit upwind scheme, the hump disappeared due to the numerical diffusion.

result suggests that if we use a large time step, the hump disappeared due to the numerical diffusion. Therefore, to capture all phenomena of the solution, we need to use not only a fine grid but also a sufficiently small time step.

7.4.4. Role of the diffusion term in the thin liquid film equation without the advection term. Next, we consider the role of diffusion in the thin liquid film equation without the advection. The initial condition in one dimension is

$$h(x,0) = 0.5[h_\infty + b - (h_\infty - b)\tanh(3(x-110))],$$

where $h_\infty = 0.3$. The computational domain is $\Omega = (0, 200)$. The uniform time step $\Delta t = 100$ is used. Solutions are computed up to time $T = 20000$. Fig. 7.4(a) shows the evolutions of the thin film height h with the precursor film thickness $b = 0.1$ and a grid size $N_x = 1024$. In Fig. 7.4(b), the stared and circled lines represent the evolutions of $h_{max} - h_\infty$ and $b - h_{min}$, respectively. Here, $h_{max} = \max_{x\in\Omega} h(x)$ and $h_{min} = \min_{x\in\Omega} h(x)$. They start at zero and converge quickly to constant values. Fig. 7.4(c), (d), and (e) show the thin film heights at time

$T = 20000$ with different precursor film thicknesses $b = 0.01$, 0.1, and 0.2 on three different grids, respectively. The stared, circled, and plused lines represent the results on different grids $N_x = 256$, 512, and 1024, respectively. The results show the convergence as we refine the grids. Fig. 7.4(f) shows $h_{max} - h_\infty$ and $b - h_{min}$ with different b values on the grids $N_x = 256$, 512, and 1024. This result shows that with fixed h_∞ value, $h_{max} - h_\infty$ is decreasing and $b - h_{min}$ is concave with respect to b.

Now, let us consider the numerical result with the precursor film thickness $b = 0.1$. In Fig. 7.5, the dashed and circled lines represent $h(x, 50)$ and $-(h^3 h_{xxx})_x \times 50$ at $t = 50$, respectively. The bright gray region represents $-(h^3 h_{xxx})_x > 0$ and the dark gray region represents $-(h^3 h_{xxx})_x < 0$. Positive (or negative) value of $-(h^3 h_{xxx})_x$ implies that h_t is greater (or less) than zero, i.e., $h(x, t)$ will increase (or decrease). As predicted, in the bright gray region, $h(x, 100)$ increased than $h(x, 50)$ (see the solid line in Fig. 7.5).

We also perform a similar numerical experiment in two dimensions with the following initial condition:

$$h(x, y, 0) = 0.5[h_\infty + b - (h_\infty - b)\tanh(3(x - 10) - 10\cos(\pi y/10))],$$

where $h_\infty = 0.3$ and $b = 0.01$. A 256×256 mesh is used on $\Omega = (0, 20) \times (0, 20)$. We take $\Delta t = 0.5$. Fig. 7.6(a1), (a2), and (a3) show $h(x, y, t)$ at $t = 0$, 1, and 9, respectively. In Fig. 7.6(b1), (b2), and (b3), filled contours of $-\nabla \cdot (h^3 \nabla \Delta h)$ are shown. The bright region represents $-\nabla \cdot (h^3 \nabla \Delta h) > 0$ and the dark region represents $-\nabla \cdot (h^3 \nabla \Delta h) < 0$. Positive and negative values of $-\nabla \cdot (h^3 \nabla \Delta h)$ imply that $h(x, y, t)$ will increase and decrease, respectively.

7.4.5. Convection with non-convex flux, $f(h) = h^2 - h^3$: the effect of h_∞. As shown in [17] Eq. (7.4) has a single Lax shock solution

$$h(x, t) = \begin{cases} h_\infty & \text{if } x < st, \\ b & \text{if } x > st, \end{cases}$$

where the shock speed s is given by

$$\begin{aligned} s = \frac{f(h_\infty) - f(b)}{h_\infty - b} &= -h_\infty^2 + (1 - b)h_\infty + b - b^2 \\ &= -(h_\infty - \frac{1 - b}{2})^2 + \frac{(1 - b)(1 + 3b)}{4}. \end{aligned} \tag{7.10}$$

As we can see from Eq. (7.10), the shock speed is an increasing function of h_∞ until $h_\infty = (1 - b)/2$ (see Fig. 7.7(a)). To confirm this numerically, we consider the shock speed with different h_∞ values. The initial condition in one dimension is

$$h(x, 0) = 0.5[h_\infty + b - (h_\infty - b)\tanh(3(x - 5))],$$

where $b = 0.05$. A 1024 grid is used on $\Omega = (0, 40)$. Solutions are computed up to time $T = 80$ with a time step $\Delta t = 0.5$. Fig. 7.7(b) shows the evolutions of the film height with $h_\infty = 0.1$, 0.2, and 0.3. The result indicates that higher the film height, the faster the film front evolves.

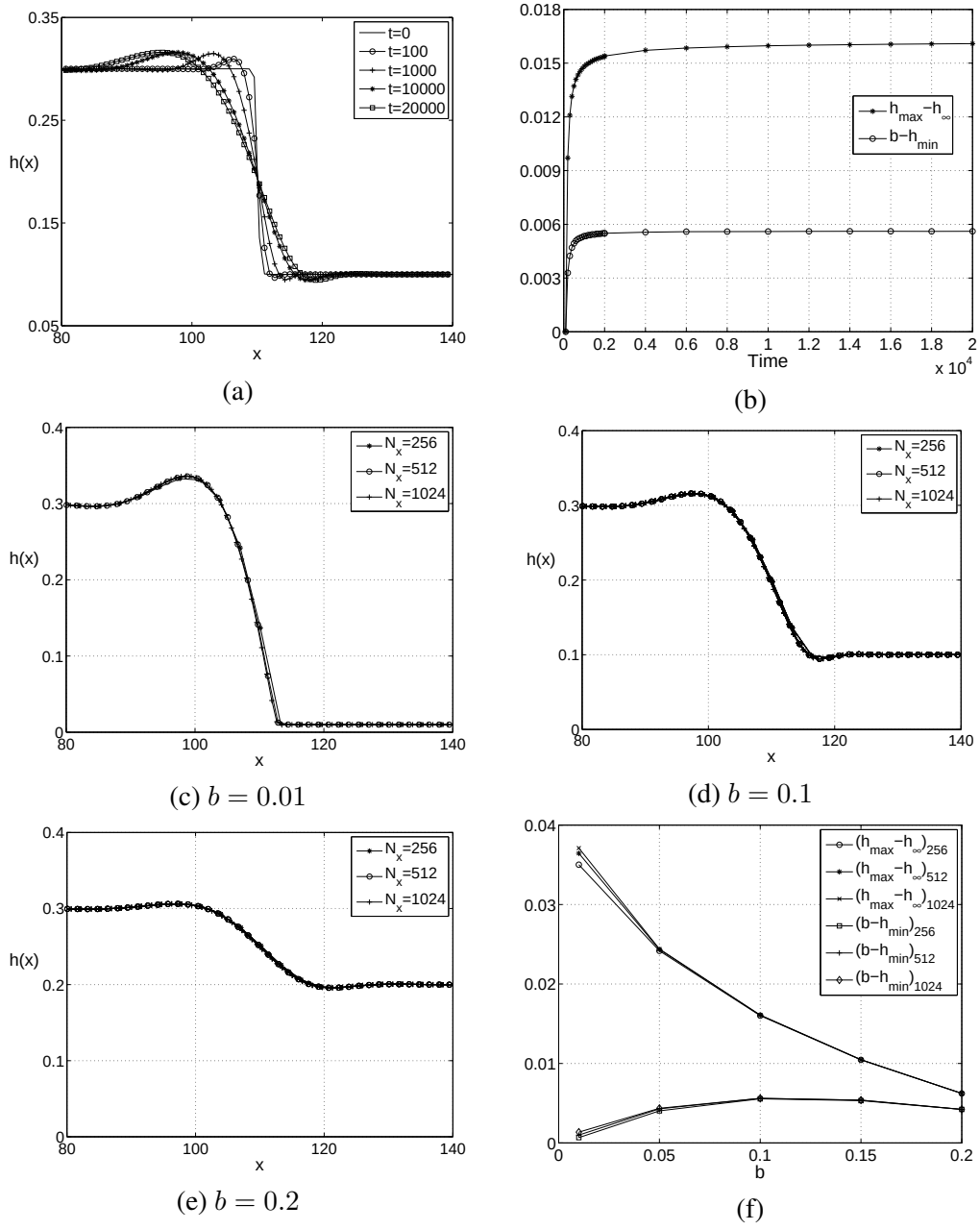


FIGURE 7.4. (a) The evolutions of the thin film height h with $b = 0.1$ and $N_x = 1024$. (b) The stared and circled lines represent evolutions of $h_{max} - h_\infty$ and $b - h_{min}$, respectively. (c), (d), and (e) show the thin film heights with $b = 0.01$, 0.1, and 0.2 on the different grids, respectively. (f) $h_{max} - h_\infty$ and $b - h_{min}$ with different b values on the grids.

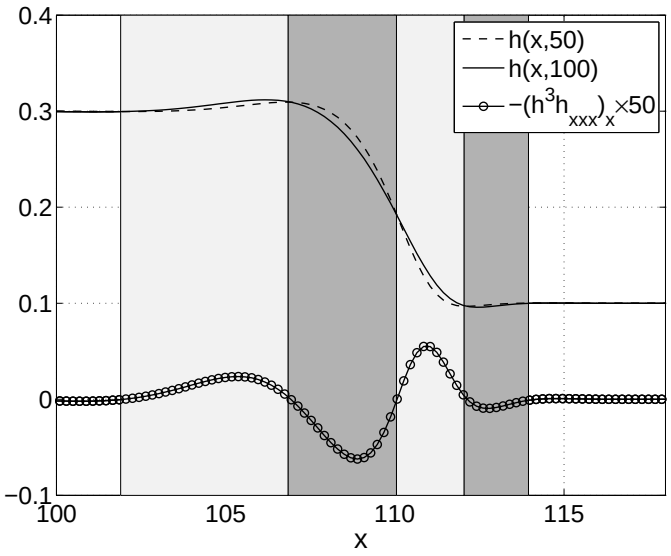


FIGURE 7.5. The bright gray region represents $-(h^3 h_{xxx})_x > 0$ and the dark gray region represents $-(h^3 h_{xxx})_x < 0$. Positive (or negative) value of $-(h^3 h_{xxx})_x$ implies that h_t is greater (or less) than zero, i.e., $h(x,t)$ will increase (or decrease).

7.4.6. Fingering instability. In this section, we study the effect of h_∞ on the finger shape for Marangoni-driven flow. The initial condition is

$$h(x,y,0) = 0.5[h_\infty + b - (h_\infty - b)\tanh(3(x-5) + \text{rand}(x,y))],$$

where $b = 0.002$ and rand(x,y) is a random number in $[-1,1]$. These perturbations model deviations from the straight front in the experiments. A 256×128 mesh is used on $\Omega = (0,100) \times (0,50)$. Solutions are run up to time $T = 600$ with a time step $\Delta t = 0.5$. Fig. 7.8(a), (b), and (c) show the evolutions of the fluid front with different $h_\infty = 0.175$, 0.2, and (c) $h_\infty = 0.225$, respectively. The times are $t = 300,\ 360,\ 450$, and 600 from bottom to top. The results show that the speed of the fluid front is an increasing function of h_∞ until $h_\infty = (1-b)/2$. We note that the fingering phenomena occurs by two mechanisms. One is that the concave front has higher film height than the convex front and the other is that higher the film height, the faster the film front evolves. The first and second mechanisms were observed from the numerical experiments in Fig. 7.6 and Fig. 7.7(b), respectively. By combination of these two effects, the fingering phenomena happens.

7.4.7. Comparison with the experimental data. Next, we compare our results with two experimental data. We consider the thin film experiments which were performed by Sur et al. in [139]. In the first experiment, the initial condition is

$$h(x,0) = 0.5[h_f + b - (h_f - b)\tanh(3(x-18))],$$

where $h_f = 0.75$, $h_\infty = 0.025$, and $b = 0.005$. A 1024 grid is used on $\Omega = (0,50)$. Calculation is run up to time $T = 120$ with a time step $\Delta t = 0.05$. The result is shown in Fig.

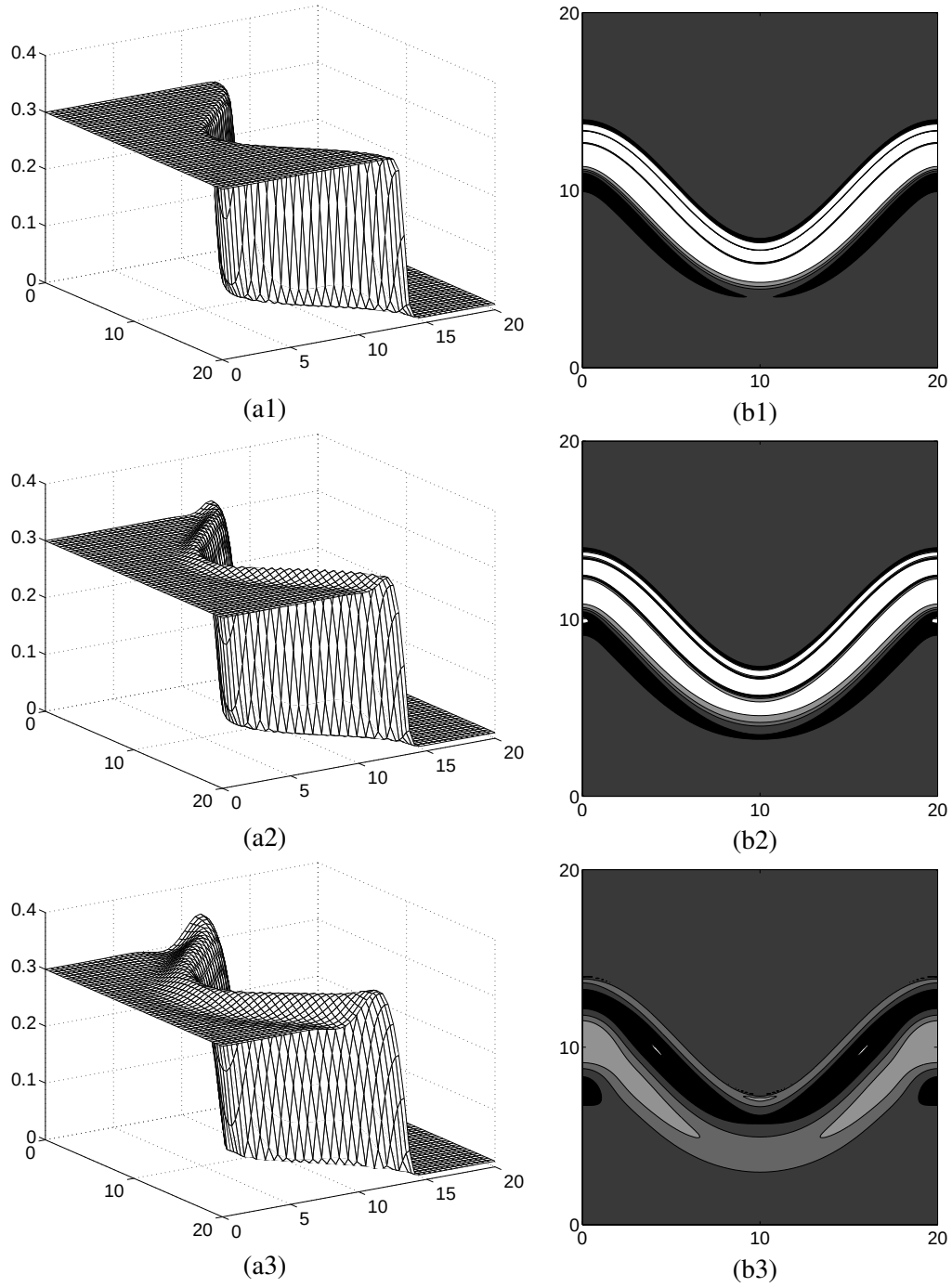
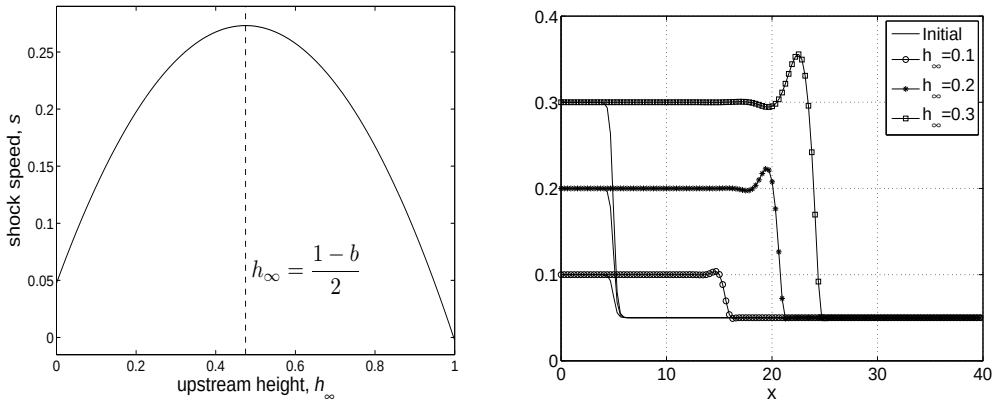
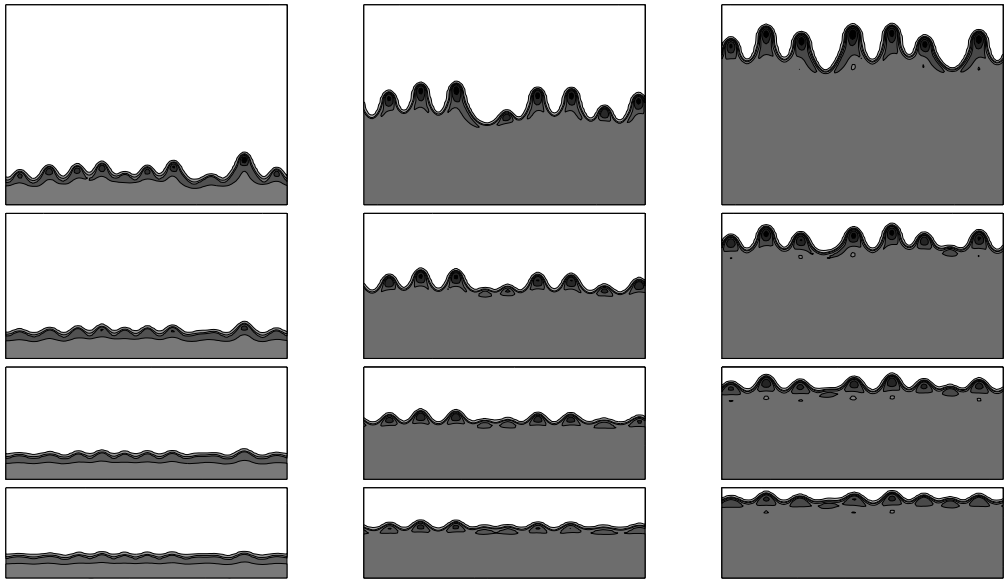


FIGURE 7.6. (a1)-(a3): the thin film height $h(x, y, t)$ at $t = 0$, 1, and 9, respectively. (b1)-(b3): Filled contours of $-\nabla \cdot (h^3 \nabla \Delta h)$ at $t = 0$, 1, and 9, respectively. The bright region represents $-\nabla \cdot (h^3 \nabla \Delta h) > 0$ and the dark region represents $-\nabla \cdot (h^3 \nabla \Delta h) < 0$.



(a) (b)

FIGURE 7.7. (a) The upstream height and shock speed. (b) The shock speed with different h_∞ values. Higher the film height, the faster the film front evolves.



(a) (b) (c)

FIGURE 7.8. The effect of h_∞ on the finger shape for Marangoni-driven flow. (a) $h_\infty = 0.175$, (b) $h_\infty = 0.20$, and (c) $h_\infty = 0.225$. The evolution of the fluid front is from bottom to top. The times are $t = 300$, 360, 450, and 600.

9.4(a). In the second experiment, the initial condition is

$$h(x,0) = 0.5[h_f + b - (h_f - b)\tanh(3(x - 15.4))],$$

(a)

(b)

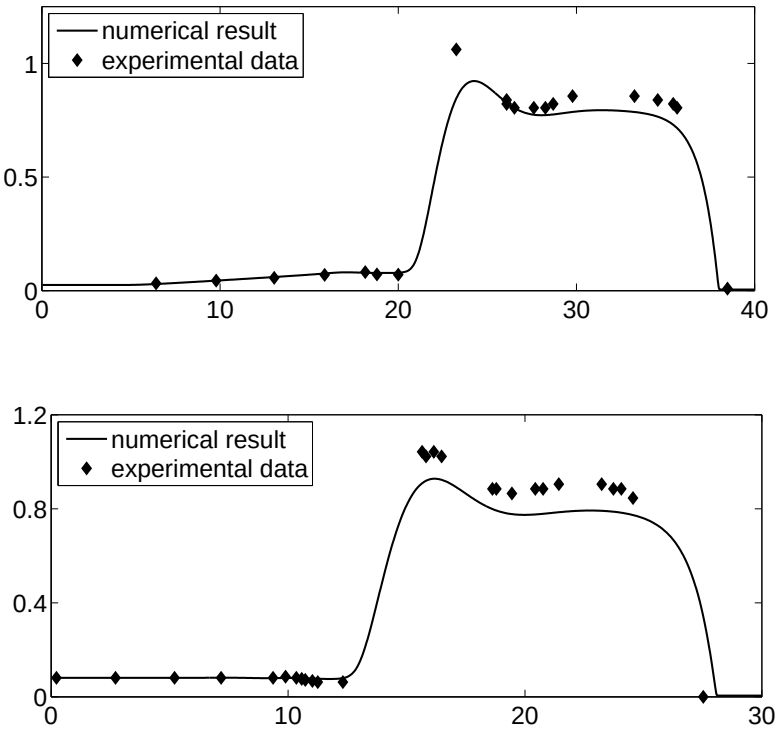


FIGURE 7.9. Comparison with the experimental data [139]. (a) $h(x,0) = 0.5[h_f + b - (h_f - b)\tanh(3(x-18))]$, $h_f = 0.75$, $h_\infty = 0.025$, and $b = 0.005$. (b) $h(x,0) = 0.5[h_f + b - (h_f - b)\tanh(3(x-15.4))]$, $h_f = 0.75$, $h_\infty = 0.081$, and $b = 0.005$.

where $h_f = 0.75$, $h_\infty = 0.081$, and $b = 0.005$. A 512 grid is used on $\Omega = (0, 40)$. Calculation is run up to time $T = 75$ with a time step $\Delta t = 0.05$. The result is shown in Fig. 9.4(b). These results show that our computational results are in qualitative agreement with experimental data.

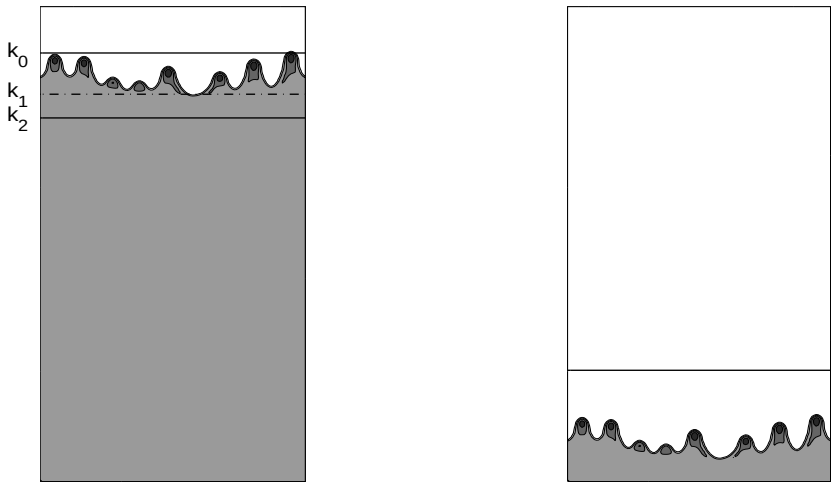
7.4.8. Long time evolution for Marangoni-driven flow. We perform a long time evolution for Marangoni-driven flow. In [17], a reference frame moving with the speed of the front was used to locally reduce the numerical diffusion. To calculate a long time evolution, we designed a shifting mesh algorithm. The proposed automatic shifting algorithm is given in Appendix. Here we shifted the mesh when the front of the flow reached nine-tenths of the domain. The reason is that if we shift the mesh at early stage, then it is inefficient since we shifted the mesh although the flow can evolve more. And if we shift the mesh at late stage, then we can not see correctly evolutions of the flow by the effect of the boundary condition b. And we did *Steps 1-2* when the maximum of the film height is greater than $(h_\infty + 8b)/9$. Because if we consider that the maximum of the film height is greater than some position that is higher than $(h_\infty + 8b)/9$, for example, $(h_\infty + 4b)/5$, then it is too late to shift the mesh since the front of the flow can be influenced by the boundary condition b. The reason that we

Algorithm:
Given an integer $k_0 = [0.9N_x]$ and a tolerance $tol = 10^{-5}$, where $[x]$ is the greatest integer not greater than x.
If $\left(\max_{1\le j\le N_y} h_{k_0 j} > \dfrac{h_\infty + 8b}{9} \right)$
then do *Steps 1-2*

Step 1 Set $k_1 = k_0$
While $\left(\max_{1\le j\le N_y} |\nabla_d h_{k_1 j}|^2 > tol \right)$ do
$k_1 = k_1 - 1$
End
Set $k_2 = [k_1 - 0.05N_x]$

Step 2 If ($1 \le i \le N_x - k_2 + 1$ and $1 \le j \le N_y$)
$\bar{h}_{ij} = h_{k_2+i-1,j}$
Else
$\bar{h}_{ij} = b$
End

We continue the calculation with $\bar{h}_{ij}$.



(a) (b)

FIGURE 7.10. The proposed automatic shifting algorithm. (a) and (b) are before and after shifting, respectively.

set $k_2 = [k_1 - 0.05N_x]$ is to give sufficiently mesh from h_∞ to hump region. Fig. 7.10 shows the proposed automatic shifting algorithm. The initial condition is

$$h(x, y, 0) = 0.5[h_\infty + b - (h_\infty - b)\tanh(3(x - 7) + \text{rand}(x, y))],$$

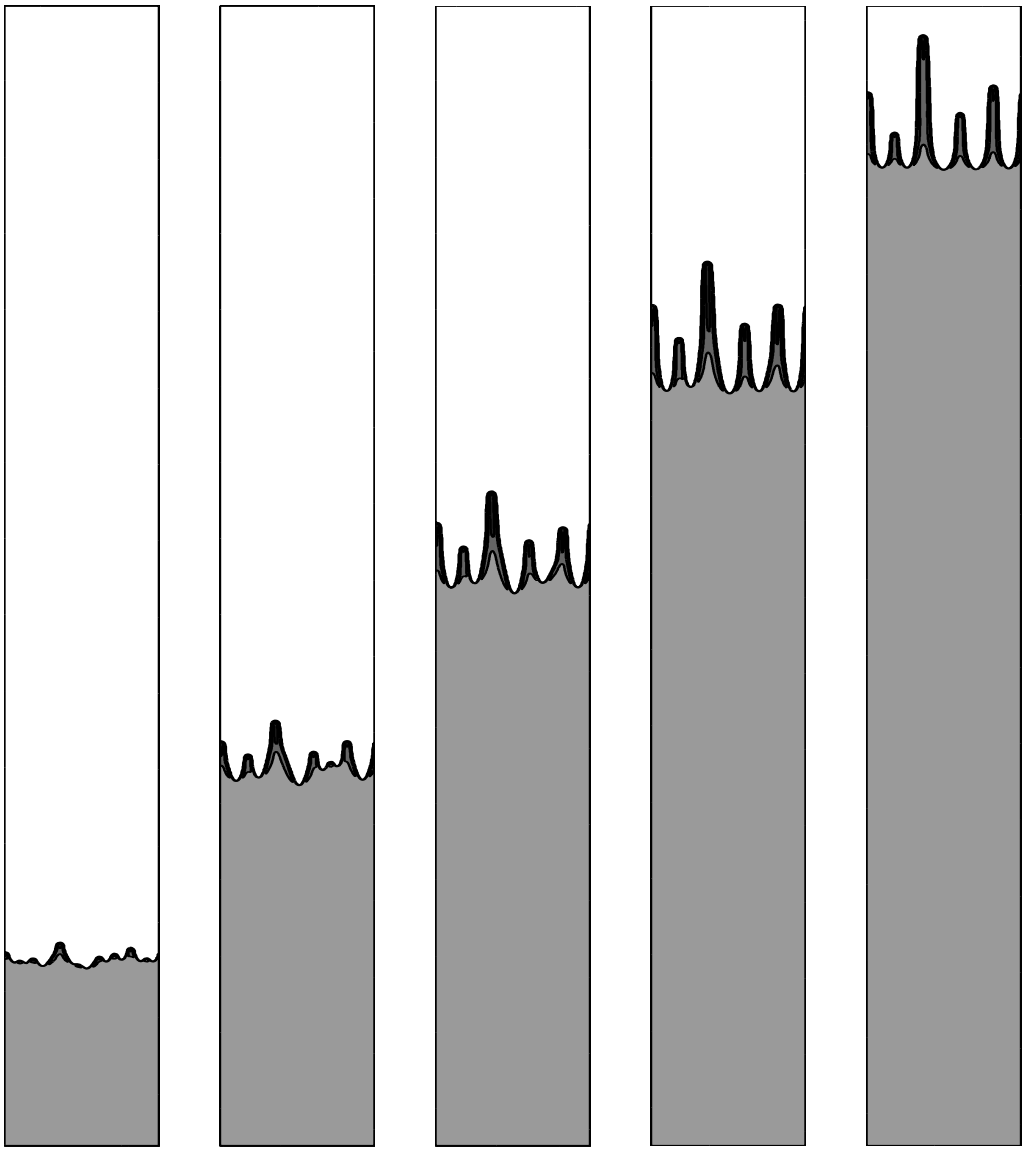


FIGURE 7.11. The finger shape for Marangoni-driven flow at the following times: (a) $t = 800$, (b) $t = 1600$, (c) $t = 2400$, (d) $t = 3200$, and (e) $t = 4000$. The effective domain size is $\Omega = (0, 578) \times (0, 50)$ and contours in $0.0019, 0.10, 0.15, 0.20, 0.25, 0.31$, and 0.35, respectively.

where $h_\infty = 0.175$ and $b = 0.002$. The computational domain using a spatial mesh of 256×128 is $\Omega = (0, 100) \times (0, 50)$. The uniform time step $\Delta t = 0.5$ is used. Fig. 7.11 shows the long time evolution with only took CPU time 3.34h.

7.5. Conclusions

We presented a new interpretation of the fingering phenomena of the thin liquid film layer through numerical investigations. A robust and accurate finite difference method was developed for the thin liquid film equations. For the advection part $(h^2 - h^3)_x$, we used an implicit ENO type scheme and got a good stability property. The resulting nonlinear discrete system was solved by an efficient nonlinear multigrid method. Numerical experiments indicated that higher the film thickness, the faster the film front evolves. The concave front has higher film thickness than the convex front. Therefore, the concave front has higher speed than the convex front and this leads to the fingering phenomena.

This Chapter is accepted in International Journal for Numerical Methods in Fluids, 2010. (in press)

Chapter 8

Conservative immersed boundary methods for two-phase fluid flows

8.1. Introduction

Many important industrial problems involve flows with multiple constitutive components. Due to inherent nonlinearities and the complexity of dealing with unknown moving interfaces, multiphase flows are challenging. There are many ways to model moving interfaces. The two main approaches to simulating multiphase and multi-component flows are interface tracking and interface capturing. In interface tracking methods (front-tracking [61], immersed interface [137], and immersed boundary [126, 91]), Lagrangian particles are used to track the interfaces. In interface capturing methods such as level-set [24, 59] and phase-field methods [81, 82], the interface is implicitly captured by a contour of a particular scalar function. The numerical method we will take is the immersed boundary method (IBM), which is originally developed by Peskin [126]. IBM started to be applied to two-phase fluid flows [92, 53, 54]. The motion of the fluid is influenced by the force generated by the interface and the interface moves at the local fluid velocity. The strength of this method is that it can handle accurately the complicated and time dependent geometry of the interface.

Consider a viscous incompressible fluid which fills a rectangular domain Ω and an interface Γ which is contained in the domain. We shall now consider the mathematical formulation of the equations of motion for the immiscible two-phase fluid. Let ρ be the variable density and μ be the variable viscosity. The equations of motion of the mixture are then as follows:

$$\begin{aligned}
\rho(I)\left(\frac{\partial \mathbf{u}(\mathbf{x},t)}{\partial t} + \mathbf{u}(\mathbf{x},t)\cdot\nabla\mathbf{u}(\mathbf{x},t)\right) &= -\nabla p(\mathbf{x},t) \\
&+ \nabla\cdot[\mu(I)(\nabla\mathbf{u}(\mathbf{x},t)+\nabla\mathbf{u}(\mathbf{x},t)^T)] + \mathbf{F}(\mathbf{x},t) + \rho(I)\mathbf{g}, \qquad (8.1)\\
\nabla\cdot\mathbf{u}(\mathbf{x},t) &= 0,\\
\mathbf{F}(\mathbf{x},t) &= \int_\Gamma \mathbf{f}(s,t)\delta^2(\mathbf{x}-\mathbf{X}(s,t))ds,\\
\mathbf{f}(s,t) &= \sigma\frac{\partial^2\mathbf{X}(s,t)}{\partial s^2},\\
\frac{\partial\mathbf{X}(s,t)}{\partial t} &= \mathbf{U}(s,t), \qquad (8.2)\\
\mathbf{U}(s,t) &= \int_\Omega \mathbf{u}(\mathbf{x},t)\delta^2(\mathbf{x}-\mathbf{X}(s,t))d\mathbf{x}. \qquad (8.3)
\end{aligned}$$

The fluid velocity $\mathbf{u}(\mathbf{x},t)$, fluid pressure $p(\mathbf{x},t)$, and singular surface tension force density $\mathbf{F}(\mathbf{x},t)$ are functions of $(\mathbf{x},t)$, where $\mathbf{x}=(x,y)$ are Cartesian coordinates. t is the time

and $\mathbf{g}$ is the gravity. The configuration of the immersed boundary is described by the function $\mathbf{X}(s,t)$, where $0 \leq s \leq L$ and L is the unstressed length of the boundary (see Fig. 8.1 (a)). The boundary force density $\mathbf{f}(s,t)$ and the boundary velocity $\mathbf{U}(s,t)$ are also functions of s and t. The core of the immersed boundary method is the delta function, which describes the interaction between the fluid and the immersed boundary. Discontinuous material properties can easily be accommodated through the numerical construction of an indicator function, $I(\mathbf{x},t)$. Let us define a gradient field,

$$\nabla I(\mathbf{x},t) = \int_{\Gamma} \mathbf{n}(\mathbf{X}(s,t))\delta^2(\mathbf{x} - \mathbf{X}(s,t))ds, \tag{8.4}$$

which is zero except near the interface. To find the indicator function, the Poisson equation

$$\Delta I(\mathbf{x},t) = \nabla \cdot \int_{\Gamma} \mathbf{n}(\mathbf{X}(s,t))\delta^2(\mathbf{x} - \mathbf{X}(s,t))ds$$

is solved. Then, the variable fluid properties, ρ and μ, can be represented by

$$\rho(I(\mathbf{x},t)) = \rho_1 + (\rho_2 - \rho_1)I(\mathbf{x},t) \text{ and } \mu(I(\mathbf{x},t)) = \mu_1 + (\mu_2 - \mu_1)I(\mathbf{x},t),$$

where ρ_i and μ_i for $i = 1, 2$ are density and viscosity of fluid i, respectively. There is no guarantee that the advected immersed boundary preserves area in time. An area conservation is an important issue in modeling free interface problems. If the area loss happens, it could increase a local curvature of the interface and results in overestimating surface tension force. Overestimated surface tension force induces a wrong velocity field which moves the interface to wrong position.

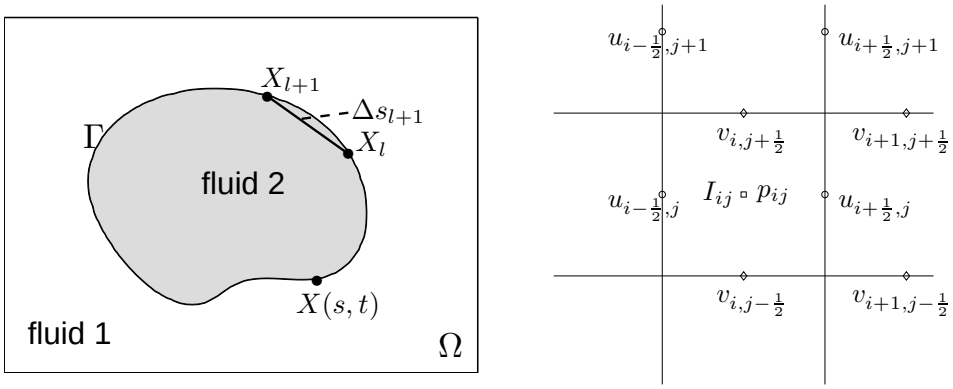
When an exact projection method is used for solving the Navier-Stokes equations, the velocity field on the Eulerian grid will be discretely divergence-free, but this does not guarantee that the interpolated velocity field through the delta function is continuously divergence free. This can result in the volume loss [121]. This is particularly problematic for interfaces under tension, which have been shown to exhibit volume loss with the IB method [28, 103, 127, 132]. A solution for fixing the volume loss problem is using the modified divergence stencils of Peskin and Printz. Peskin and Printz [132] made a dramatic improvement in the overall volume conservation. The key idea is the introduction of a new finite difference divergence operator which is constructed in such a way that the interpolated velocity field in which the immersed boundary moves is more nearly divergence-free.

In this Chapter, we propose a simple area preserving correction scheme for the two-phase immiscible incompressible Navier-Stokes flows with an immersed boundary method. The idea of area preserving correction scheme is to correct the interface location normally to the interface so that the area remains constant. The rest of the Chapter is organized as follows. In Section 8.2, the numerical method will be introduced. The experimental results will be discussed in Section 8.3. Finally, some conclusions will be drawn in Section 8.4.

8.2. Numerical method

In this section, we present the numerical solution algorithm. A staggered marker-and-cell (MAC) mesh of Harlow and Welch [71] is used in which pressure and indicator function are stored at cell centers and velocities at cell interfaces (see Fig. 8.1(b)).

Let a computational domain be partitioned in Cartesian geometry into a uniform mesh with mesh spacing h. The center of each cell, Ω_{ij}, is located at $\mathbf{x}_{ij} = (x_i, y_j) = ((i -$



(a) (b)

FIGURE 8.1. (a) Velocities are defined at cell boundaries while the pressure and indicator function are defined at the cell centers. (b) Immersed boundary points.

$0.5)h, (j-0.5)h)$ 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. We use a set of M Lagrangian points $\mathbf{X}_l^n = (x_l^n, y_l^n)$ for $l = 1, \ldots, M$ to discretize the immersed boundary with the initial boundary mesh width $\Delta s_{l+1} = \sqrt{(x_{l+1} - x_l)^2 + (y_{l+1} - y_l)^2}$.

At the beginning of each time step, given a divergence free velocity field $\mathbf{u}^n$, a boundary force $\mathbf{F}^n$, and a given boundary configuration $\mathbf{X}^n$, we want to find $\mathbf{u}^{n+1}$, p^{n+1}, and $\mathbf{X}^{n+1}$ which solve the following semi-implicit scheme in time and space:

$$\begin{aligned} \rho^n \frac{\mathbf{u}^{n+1} - \mathbf{u}^n}{\Delta t} &= -\rho^n \mathbf{u}^n \cdot \nabla_d \mathbf{u}^n - \nabla_d p^{n+1} + \mu \Delta_d \mathbf{u}^n + \mathbf{F}^n + \rho^n \mathbf{g}, \\ \nabla_d \cdot \mathbf{u}^{n+1} &= 0, \end{aligned}$$

where $\rho^n = \rho_1 + (\rho_2 - \rho_1) I^n$ and $\mathbf{g} = (0, -g)$.

Step 1. Find the force $\mathbf{f}^n$ on the immersed boundary from the given boundary configuration $\mathbf{X}^n$. For $l = 1, \ldots, M$,

$$\mathbf{f}_l^n = \sigma \frac{\mathbf{X}_{l+1}^n - 2\mathbf{X}_l^n + \mathbf{X}_{l-1}^n}{\Delta s_{l+1/2}^2}, \tag{8.5}$$

where $\Delta s_{l+1/2} = (\Delta s_l + \Delta s_{l+1})/2$ and σ is a surface tension coefficient. Note that the subscript arithmetic on l in Eq. (8.5) has to be interpreted in a periodic sense, since the boundary is closed: if $l = M$, then $l + 1 = 1$; if $l = 1$, then $l - 1 = M$.

Step 2. Spread the boundary force into the nearby lattice points of the fluid.

$$\mathbf{F}_{ij}^n = \sum_{l=1}^{M} \mathbf{f}_l^n \delta_h^2(\mathbf{x}_{ij} - \mathbf{X}_l^n) \Delta s_{l+1/2} \text{ for } i = 1, \ldots, N_x \text{ and } j = 1, \ldots, N_y,$$

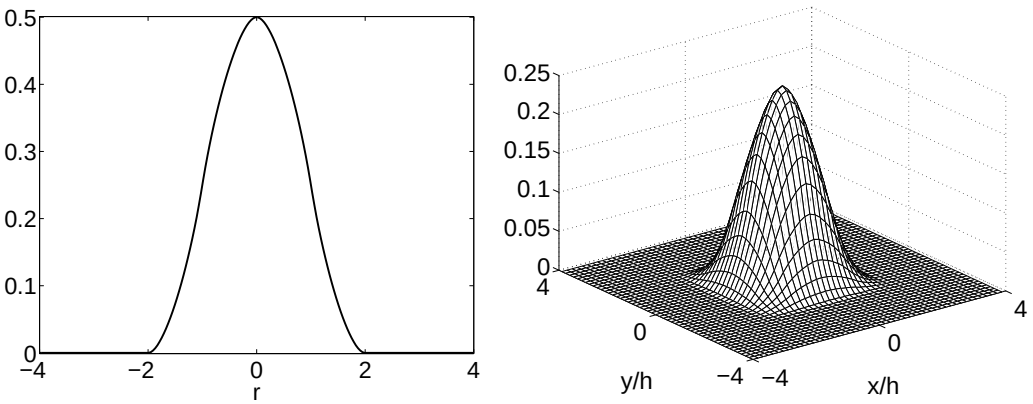
where δ_h^2 is a smoothed approximation to the two-dimensional Dirac delta function:

$$\delta_h^2(\mathbf{x}) = \frac{1}{h^2} \phi\left(\frac{x}{h}\right) \phi\left(\frac{y}{h}\right),$$

where

$$\phi(r) = \begin{cases} \frac{3-2|r|+\sqrt{1+4|r|-4r^2}}{8} & \text{if } |r| \le 1, \\ \frac{5-2|r|-\sqrt{-7+12|r|-4r^2}}{8} & \text{if } 1 < |r| \le 2, \\ 0 & \text{if } 2 < |r|. \end{cases}$$

One- and two-dimensional Dirac delta functions are shown in Fig. 8.2. The motivation for this particular choice of $\phi(r)$ is given in [131].



(a) (b)

FIGURE 8.2. (a) One- and (b) Two-dimensional Dirac delta functions.

Step 3. Solve the Navier-Stokes equations on the rectangular lattice to get the update $\mathbf{u}^{n+1}$ and p^{n+1} from $\mathbf{u}^n$ and $\mathbf{X}^n$.

Solve an intermediate velocity field, $\tilde{\mathbf{u}}$, which generally does not satisfy the incompressible condition, without the pressure gradient term,

$$\frac{\tilde{\mathbf{u}} - \mathbf{u}^n}{\Delta t} + \mathbf{u}^n \cdot \nabla_d \mathbf{u}^n = \frac{\mu}{\rho^n} \Delta_d \mathbf{u}^n + \frac{1}{\rho^n} \mathbf{F}^n + \mathbf{g}.$$

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

$$\begin{aligned} \tilde{u}_{i+\frac{1}{2},j} &= u^n_{i+\frac{1}{2},j} - \Delta t (u u_x + v u_y)^n_{i+\frac{1}{2},j} + \frac{\Delta t}{\rho^n_{i+\frac{1}{2},j}} F^{x-edge}_{i+\frac{1}{2},j} \\ &+ \frac{\mu \Delta t}{h^2 \rho^n_{i+\frac{1}{2},j}} \left(u^n_{i+\frac{3}{2},j} + u^n_{i-\frac{1}{2},j} - 4u^n_{i+\frac{1}{2},j} + u^n_{i+\frac{1}{2},j+1} + u^n_{i+\frac{1}{2},j-1} \right), \\ \tilde{v}_{i,j+\frac{1}{2}} &= v^n_{i,j+\frac{1}{2}} - \Delta t (u v_x + v v_y)^n_{i,j+\frac{1}{2}} + \frac{\Delta t}{\rho^n_{i,j+\frac{1}{2}}} F^{y-edge}_{i,j+\frac{1}{2}} - g \Delta t \\ &+ \frac{\mu \Delta t}{h^2 \rho^n_{i,j+\frac{1}{2}}} \left(v^n_{i,j+\frac{3}{2}} + v^n_{i,j-\frac{1}{2}} - 4v^n_{i,j+\frac{1}{2}} + v^n_{i+1,j+\frac{1}{2}} + v^n_{i-1,j+\frac{1}{2}} \right), \end{aligned}$$

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

$$\begin{aligned}
(uu_x + vu_y)^n_{i+\frac{1}{2},j} &= u^n_{i+\frac{1}{2},j}\bar{u}^n_{x_{i+\frac{1}{2},j}} \\
&\quad + \frac{v^n_{i,j-\frac{1}{2}} + v^n_{i+1,j-\frac{1}{2}} + v^n_{i,j+\frac{1}{2}} + v^n_{i+1,j+\frac{1}{2}}}{4}\bar{u}^n_{y_{i+\frac{1}{2},j}}, \\
(uv_x + vv_y)^n_{i,j+\frac{1}{2}} &= v^n_{i,j+\frac{1}{2}}\bar{v}^n_{y_{i,j+\frac{1}{2}}} \\
&\quad + \frac{u^n_{i-\frac{1}{2},j} + u^n_{i-\frac{1}{2},j+1} + u^n_{i+\frac{1}{2},j} + u^n_{i+\frac{1}{2},j+1}}{4}\bar{v}^n_{x_{i,j+\frac{1}{2}}}.
\end{aligned}$$

The values $\bar{u}^n_{x_{i+\frac{1}{2},j}}$ and $\bar{u}^n_{y_{i+\frac{1}{2},j}}$ are computed using the upwind procedure. The procedure is

$$\bar{u}^n_{x_{i+\frac{1}{2},j}} = \begin{cases} \frac{u^n_{i+\frac{1}{2},j} - u^n_{i-\frac{1}{2},j}}{h} & \text{if } u^n_{i+\frac{1}{2},j} > 0 \\ \frac{u^n_{i+\frac{3}{2},j} - u^n_{i+\frac{1}{2},j}}{h} & \text{otherwise} \end{cases}$$

and

$$\bar{u}^n_{y_{i+\frac{1}{2},j}} = \begin{cases} \frac{u^n_{i+\frac{1}{2},j} - u^n_{i+\frac{1}{2},j-1}}{h} & \text{if } v^n_{i,j-\frac{1}{2}} + v^n_{i+1,j-\frac{1}{2}} + v^n_{i,j+\frac{1}{2}} + v^n_{i+1,j+\frac{1}{2}} > 0 \\ \frac{u^n_{i+\frac{1}{2},j+1} - u^n_{i+\frac{1}{2},j}}{h} & \text{otherwise.} \end{cases}$$

The quantities $\bar{v}^n_{x_{i,j+\frac{1}{2}}}$ and $\bar{v}^n_{y_{i,j+\frac{1}{2}}}$ are computed in a similar manner. Then, we solve the following equations for the advanced pressure field at $(n+1)$ time step.

$$\frac{\mathbf{u}^{n+1} - \tilde{\mathbf{u}}}{\Delta t} = -\frac{1}{\rho^n}\nabla_d p^{n+1}, \tag{8.6}$$

$$\nabla_d \cdot \mathbf{u}^{n+1} = 0. \tag{8.7}$$

With application of the divergence operator to Eq. (8.6), we find that the Poisson equation for the pressure at the advanced time $(n+1)$.

$$\nabla_d \cdot \left(\frac{1}{\rho^n}\nabla_d p^{n+1}\right) = \frac{1}{\Delta t}\nabla_d \cdot \tilde{\mathbf{u}}, \tag{8.8}$$

where we have made use of the Eq. (8.7) and the terms are defined as in the following.

$$\begin{aligned}
\nabla_d \cdot \left(\frac{1}{\rho^n}\nabla_d p^{n+1}_{ij}\right) &= \frac{\frac{1}{\rho^n_{i+\frac{1}{2},j}}p^{n+1}_{i+1,j} + \frac{1}{\rho^n_{i-\frac{1}{2},j}}p^{n+1}_{i-1,j} + \frac{1}{\rho^n_{i,j+\frac{1}{2}}}p^{n+1}_{i,j+1} + \frac{1}{\rho^n_{i,j-\frac{1}{2}}}p^{n+1}_{i,j-1}}{h^2} \\
&\quad - \frac{\frac{1}{\rho^n_{i+\frac{1}{2},j}} + \frac{1}{\rho^n_{i-\frac{1}{2},j}} + \frac{1}{\rho^n_{i,j+\frac{1}{2}}} + \frac{1}{\rho^n_{i,j-\frac{1}{2}}}}{h^2}p^{n+1}_{ij}, \\
\nabla_d \cdot \tilde{\mathbf{u}}_{ij} &= \frac{\tilde{u}_{i+\frac{1}{2},j} - \tilde{u}_{i-\frac{1}{2},j}}{h} + \frac{\tilde{v}_{i,j+\frac{1}{2}} - \tilde{v}_{i,j-\frac{1}{2}}}{h},
\end{aligned}$$

where $\rho^n_{i+\frac{1}{2},j} = (\rho^n_{ij} + \rho^n_{i+1,j})/2$ and the other terms are similarly defined. The boundary condition for the pressure is

$$\mathbf{n} \cdot \nabla_d p^{n+1} = \mathbf{n} \cdot \left(-\rho^n \frac{\mathbf{u}^{n+1} - \mathbf{u}^n}{\Delta t} - \rho^n (\mathbf{u} \cdot \nabla_d \mathbf{u})^n + \mu \Delta_d \mathbf{u}^n + \mathbf{F}^n + \rho^n \mathbf{g} \right),$$

where $\mathbf{n}$ is the unit normal vector to the domain boundary. And also we get

$$\mathbf{n} \cdot \nabla_d p^{n+1} = \mathbf{n} \cdot \rho^n \mathbf{g}.$$

The resulting linear system of Eq. (8.8) is solved using a multigrid method [153], specifically, V-cycles with a Gauss-Seidel relaxation. Then the divergence-free normal velocities u^{n+1} and v^{n+1} are defined by

$$\begin{aligned}
\mathbf{u}^{n+1} &= \tilde{\mathbf{u}} - \frac{\Delta t}{\rho^n} \nabla_d p^{n+1}, \text{ i.e.,} \\
u^{n+1}_{i+\frac{1}{2},j} &= \tilde{u}_{i+\frac{1}{2},j} - \frac{\Delta t}{\rho^n_{i+\frac{1}{2},j} h} (p_{i+1,j} - p_{ij}), \\
v^{n+1}_{i,j+\frac{1}{2}} &= \tilde{v}_{i,j+\frac{1}{2}} - \frac{\Delta t}{\rho^n_{i,j+\frac{1}{2}} h} (p_{i,j+1} - p_{ij}).
\end{aligned}$$

Step 4. Once the updated fluid velocity, $\mathbf{u}^{n+1}$, has been determined, we can find the velocity, $\mathbf{U}^{n+1}$, and then the new position, $\mathbf{X}^{n+1}$, of the immersed boundary points. This is done using a discretization of Eqs. (8.2) and (8.3). That is, for $l = 1, \ldots, M$,

$$\begin{aligned}
\mathbf{U}^{n+1}_l &= \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} \mathbf{u}^{n+1}_{ij} \delta^2_h (\mathbf{x}_{ij} - \mathbf{X}^n_l) h^2, \\
\mathbf{X}^{n+1}_l &= \mathbf{X}^n_l + \Delta t \mathbf{U}^n_l.
\end{aligned} \tag{8.9}$$

This completes the description of the process (*Steps 1-4*, above) by which the quantities $\mathbf{u}$ and $\mathbf{X}$ are updated.

8.2.1. Discretization of the indicator function. In this section, we will present the numerical method to calculate the indicator function. Let the discretization of the right hand side of Eq. (8.4) be $\mathbf{G}^n_{ij}$:

$$\mathbf{G}^n_{ij} = \sum_{l=1}^{M} \mathbf{n}^n_l \delta^2_h (\mathbf{x}_{ij} - \mathbf{X}^n_l) \Delta s_{l+\frac{1}{2}}.$$

For a given interface position $\mathbf{X}_l$, the corresponding unit normal vector $\mathbf{n}_l = (m_l, n_l)$ will be calculated by using three points $\mathbf{X}_{l-1} = (x_{l-1}, y_{l-1})$, $\mathbf{X}_l = (x_l, y_l)$, $\mathbf{X}_{l+1} = (x_{l+1}, y_{l+1})$ with the quadratic polynomial approximation. Let the quadratic polynomial approximation be

$$x(t) = \alpha_1 t^2 + \beta_1 t + \gamma_1 \text{ and } y(t) = \alpha_2 t^2 + \beta_2 t + \gamma_2.$$

And assume $\mathbf{X}_{l-1} = (x(0), y(0))$, $\mathbf{X}_l = (x(\Delta s_l), y(\Delta s_l))$, and $\mathbf{X}_{l+1} = (x(\Delta s_l + \Delta s_{l+1}), y(\Delta s_l + \Delta s_{l+1}))$, then the parameters α_1, β_1, γ_1, α_2, β_2, and γ_2 can be calculated by the following equations.

$$\begin{pmatrix} \alpha_1 \\ \beta_1 \\ \gamma_1 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 1 \\ \Delta s_l^2 & \Delta s_l & 1 \\ (\Delta s_l + \Delta s_{l+1})^2 & \Delta s_l + \Delta s_{l+1} & 1 \end{pmatrix}^{-1} \begin{pmatrix} x(0) \\ x(\Delta s_l) \\ x(\Delta s_l + \Delta s_{l+1}) \end{pmatrix},$$

$$\begin{pmatrix} \alpha_2 \\ \beta_2 \\ \gamma_2 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 1 \\ \Delta s_l^2 & \Delta s_l & 1 \\ (\Delta s_l + \Delta s_{l+1})^2 & \Delta s_l + \Delta s_{l+1} & 1 \end{pmatrix}^{-1} \begin{pmatrix} y(0) \\ y(\Delta s_l) \\ y(\Delta s_l + \Delta s_{l+1}) \end{pmatrix}.$$

Now we get the unit normal vector as

$$\mathbf{n}_l = (m_l, n_l) = \left(\frac{\frac{dy(\Delta s_l)}{dt}}{\sqrt{\left(\frac{dx(\Delta s_l)}{dt}\right)^2 + \left(\frac{dy(\Delta s_l)}{dt}\right)^2}}, \frac{-\frac{dx(\Delta s_l)}{dt}}{\sqrt{\left(\frac{dx(\Delta s_l)}{dt}\right)^2 + \left(\frac{dy(\Delta s_l)}{dt}\right)^2}} \right).$$

Then, we solve the following Poisson equation using the multigrid method.

$$\Delta_d I^n = \nabla_d \cdot \mathbf{G}^n.$$

8.2.2. The area correction algorithm. For a given interface position $\mathbf{X}$, the area and the relative area error are defined as in the following.

$$A(\mathbf{X}) = \frac{1}{2}\sum_{l=1}^{M}(X_l Y_{l+1} - Y_l X_{l+1}),\ A_{error}(\mathbf{X}) = \frac{|A(\mathbf{X}^0) - A(\mathbf{X})|}{A(\mathbf{X}^0)}.$$

For a given tolerance, tol, to find a solution $\bar{\mathbf{X}}^{n+1}$ satisfying $A_{error}(\bar{\mathbf{X}}^{n+1}) < tol$, we take the following algorithm.

1) Update the interface X_l^{n+1}, Y_l^{n+1} according to Eq. (8.9)

$$(X_l^{n+1}, Y_l^{n+1}) = (X_l^n, Y_l^n) + \Delta t(U_l^n, V_l^n).$$

2) Compute the area $A(\mathbf{X}^{n+1})$

Check $A_{error}(\mathbf{X}^{n+1}) < tol$ or not. If not, do the following *3)* and *4)*.

3) Determine the parameter ϵ

$$\epsilon = \frac{-\beta \pm \sqrt{\beta^2 - 4\alpha(A(\mathbf{X}^{n+1}) - A(\mathbf{X}^0))}}{2\alpha}.$$

The parameter ϵ is a root of a quadratic equation from the area correction.

$$A(\mathbf{X}^0) = A(\mathbf{X}^{n+1}) + \alpha\epsilon^2 + \beta\epsilon,$$

where

$$\begin{aligned} \alpha &= \frac{1}{2}\sum_{l=1}^{M}\left(m_l n_{l+1} - n_l m_{l+1}\right), \\ \beta &= \frac{1}{2}\sum_{l=1}^{M}\left(m_l Y_{l+1}^{n+1} + n_{l+1} X_l^{n+1} - (n_l X_{l+1}^{n+1} + m_{l+1} Y_l^{n+1})\right), \end{aligned}$$

where (m_l, n_l) is the unit normal vector at the l-th interface node at time level t^{n+1}.

4) Update the interface $\bar{X}_l^{n+1}$, $\bar{Y}_l^{n+1}$ with area correction

$$(\bar{X}_l^{n+1}, \bar{Y}_l^{n+1}) = (X_l^{n+1}, Y_l^{n+1}) + \epsilon(m_l, n_l).$$

The following two case statements are directly obtainable from the above definitions. Let $\alpha > 0$ and

$$\begin{aligned} \epsilon_1 &= \frac{-\beta + \sqrt{\beta^2 - 4\alpha(A(\mathbf{X}^{n+1}) - A(\mathbf{X}^0))}}{2\alpha}, \\ \epsilon_2 &= \frac{-\beta - \sqrt{\beta^2 - 4\alpha(A(\mathbf{X}^{n+1}) - A(\mathbf{X}^0))}}{2\alpha}. \end{aligned}$$

If $A(\mathbf{X}^{n+1}) > A(\mathbf{X}^0)$, then $\epsilon_2 < \epsilon_1 < 0$ and we choose ϵ_1. If $A(\mathbf{X}^{n+1}) < A(\mathbf{X}^0)$, then $\epsilon_2 < 0 < \epsilon_1$ and we choose ϵ_1. These provided reasons are illustrated in Figs. 8.3 and 8.4. Therefore, we always choose ϵ_1.

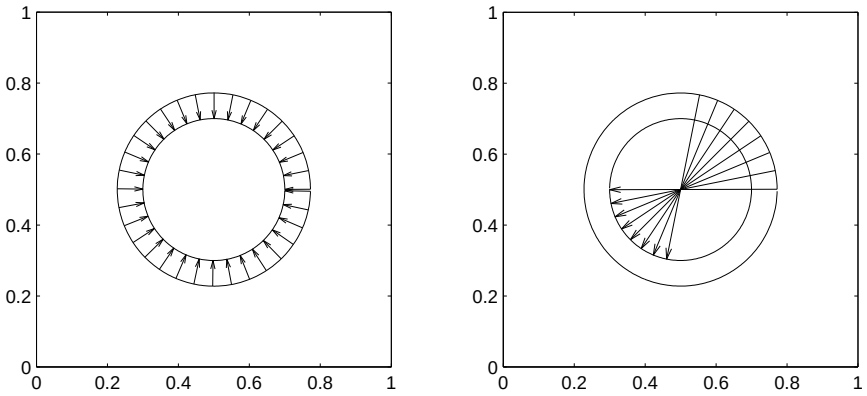


FIGURE 8.3. $\epsilon_2 < \epsilon_1 < 0$. Form left to right: Area correction with ϵ_1 and with ϵ_2, respectively

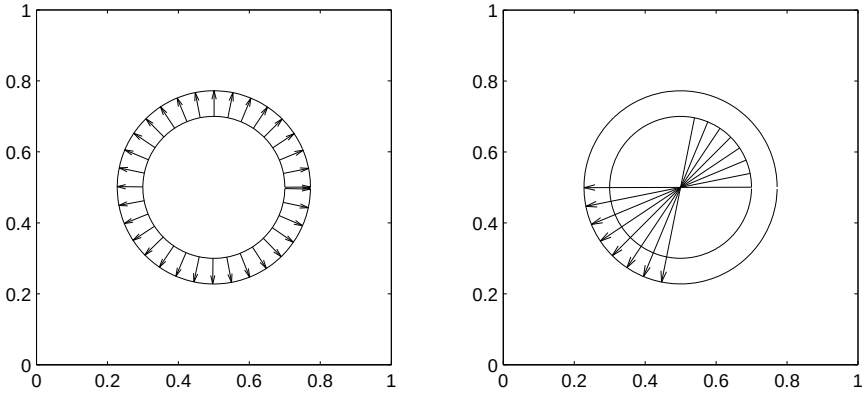


FIGURE 8.4. $\epsilon_2 < 0 < \epsilon_1$. Form left to right: Area correction with ϵ_1 and with ϵ_2, respectively.

8.3. Numerical examples

It is well known that the immersed boundary method does not conserve the area enclosed by the immersed boundary although the velocity field on the Eulerian grid satisfies a discrete divergence free condition [121, 132, 111]. However, we overcome the area loss problem by the area correction algorithm that described in Subsection 8.2.2. In this section, we perform a number of numerical experiments to investigate the effect of our area correction algorithm for the immersed boundary problem.

8.3.1. Relaxation to a circle. The perturbed surface of the droplet is given in polar coordinates (r, θ) and the initial droplet boundary on the computational domain $\Omega = (0, 1) \times (0, 1)$ is

$$\mathbf{x} = (x, y) = (0.5 + r\cos(\theta), 0.5 + r\sin(\theta)),\ 0 \le \theta < 2\pi$$

where $r = 0.25 + 0.1\cos(n\theta)$ and n is the oscillation mode. It is well known that the boundary will relax to a circle with the area unchanged. The computations are carried out for three different modes $n = 3$, 5, and 8 (see the first row of Fig. 8.5). We take $h = 1/128$, $\Delta t = h/64$, $\rho = 1$, $\mu = 0.01$, and $\sigma = 130$ as the initial parameters. As we expected all three models of interfaces relaxed to a circle. We also see that the larger the mode is, the interface relaxes quickly to a circle by the effect of the higher curvature of the interface.

8.3.2. Pressure difference of drop. In this section, we calculate theoretically and numerically the pressure difference $[p]$ with different radii and mesh sizes. In the absence of viscous, gravitational, or other external forces, surface tension causes a static drop to become spherical. Laplace's formula for an infinite cylinder surrounded by a background fluid at zero pressure, Eq. (8.1), gives the internal drop pressure

$$p_{\text{drop}} = \frac{\sigma}{r},$$

where r is the drop radius.

In this experiment, the droplet is placed at the center of the unit square domain, and has three different radii $r = 0.1$, 0.2, and 0.4. A 128×128 mesh, $h = 1/128$, $\Delta t = h/64$, $\rho = 1$, $\sigma = 10$, and $T = h/64$ are employed. Fig. 8.6 (a) and (b) shows the pressure field with $r = 0.2$ and the slice plot of p with three different radii on $y = 0.5$, respectively.

In the next experiment, the droplet is placed at the center of the unit square domain, and has a radius of 0.1. The pressure differences are computed on the uniform grids, $h = 1/2^n$, and with corresponding time steps, $\Delta t = h/64$ for $n = 5$, 6, 7, and 8. The calculations are run up to time $T = h/64$, and $\rho = 1$ and $\sigma = 20$ are employed. In this test, the exact pressure difference $[p]$ is 200. As shown in Table 8.1, this sequence of events for the numerical pressure difference $[p]$ is qualitatively in agreement with the theoretical values by refining a mesh.

TABLE 8.1. The pressure difference $[p]$ with $\sigma = 20$ and $r = 0.1$ for different mesh sizes.

Mesh sizes	32×32	64×64	128×128	256×256
$[p]$	201.147	200.673	200.379	200.312

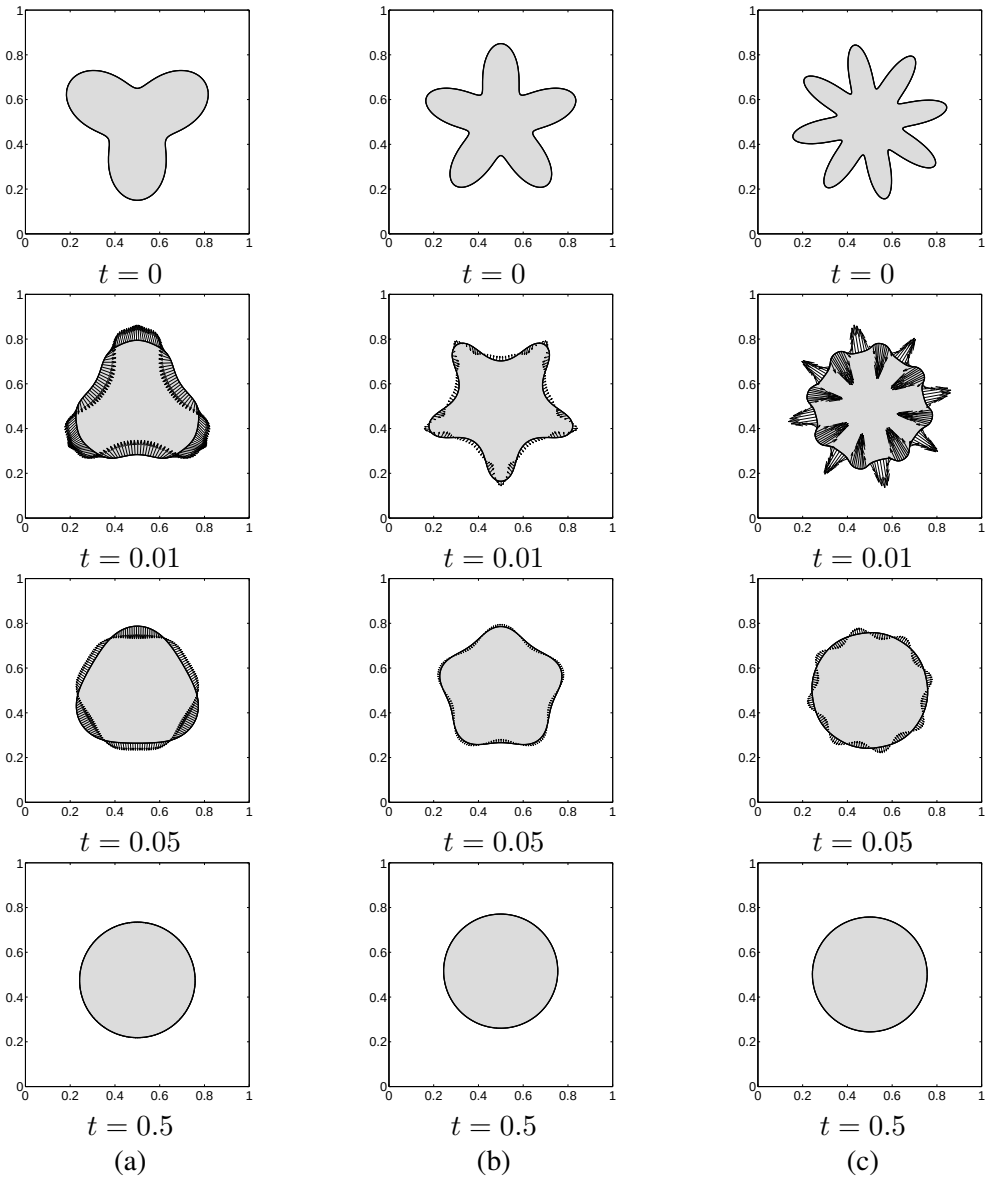


FIGURE 8.5. The evolution of interface for three different modes (a) $n = 3$, (b) $n = 5$, and (c) $n = 8$. The times are shown below each figure.

8.3.3. Area loss by spurious velocity fields induced by the surface tension. The static circular drop is surrounded by a small amplitude velocity field due to the slight unbalance between the stresses at the sites in the interfacial region. Such unphysical flow is a spurious velocity. There are many studies for spurious velocities in incompressible flow problems [40, 66, 57, 65, 107, 128]. In order to estimate the effect of a spurious velocity, we consider the static circular drop with zero initial velocity field. The initial drop has a radius of 0.3 and is centered at $(0.5, 0.5)$ in a unit square domain. The other parameters we choose are $h = 1/128$,

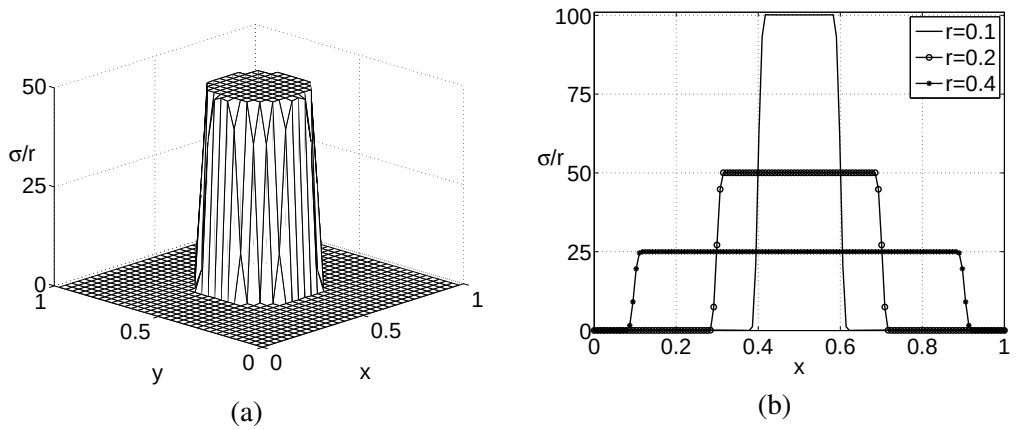


FIGURE 8.6. (a) The pressure field with $r = 0.2$. (b) The slice plot of the pressure field on $y = 0.5$.

$\Delta t = h/64$, $\rho = 1$, $\mu = 0.01$, $\sigma = 130$, and $T = 3$. Fig. 8.7 (a), (b), and (c) shows evolutions of the static circular drop. The computational times are shown below each figure. Spurious velocities near the interface (see amplitudes and directions of arrows in Fig. 8.7) lead to a misinterpretation. As a result, we observe that the area does not conserve as predicted and the area loss is as large as 18.34% when $T = 3$. Fig. 8.7 (d) illustrates the area loss rates with and without correction.

8.3.4. Area change by advection. For an accurate calculation the time step ideally remains zero for all times. But the time step is numerically not zero and this makes the area change. We consider the static circular drop with the nonzero initial velocity field such as

$$\mathbf{u} = (u, v) = (16\pi(y - 0.5), -16\pi(x - 0.5)). \tag{8.10}$$

This velocity field is divergence free. The circle has a radius of 0.25 and is centered at the center of the computational domain $\Omega = (0, 1) \times (0, 1)$. We use the simulation parameters such as $h = 1/128$, $\rho = 1.0$, $\mu = 0.01$, $\sigma = 0$, and time step $\Delta t = h/64$ within the computational time $T = 2$. Fig. 8.8 (a), (b), and (c) show evolutions of the static circular drop following the computational time. As can be seen, the area does not conserve as predicted. The larger the time step is, the area change becomes severe. We will discuss this in Section 8.3.5. Fig. 8.7 (d) illustrates the area gain rates with and without correction. If we would like to solve for the velocity field without correction, then the area gains as large as 36.13%. Even though the velocity field is divergence free, the advected interface does not conserve the volume. The reason is that we use a finite length of time step. In the continuous equation, the temporal evolution of the interface is moved by a infinitesimally small time step.

8.3.5. Rotated circle by advection. We consider the static circular drop with the nonzero initial velocity field such as in Eq. (8.10). The circle has a radius of 0.1 and is centered at $(0.5, 0.8)$ in a unit square domain. We take $h = 1/128$, $\rho = 1.0$, $\mu = 0.01$, $\sigma = 0$, and $T = 1/8$ with different time steps $\Delta t = h/128$ and $\Delta t = h/16$, respectively. For each time step, evolutions are as shown in Fig. 8.9. As can be seen from the area error in Table 8.2, the

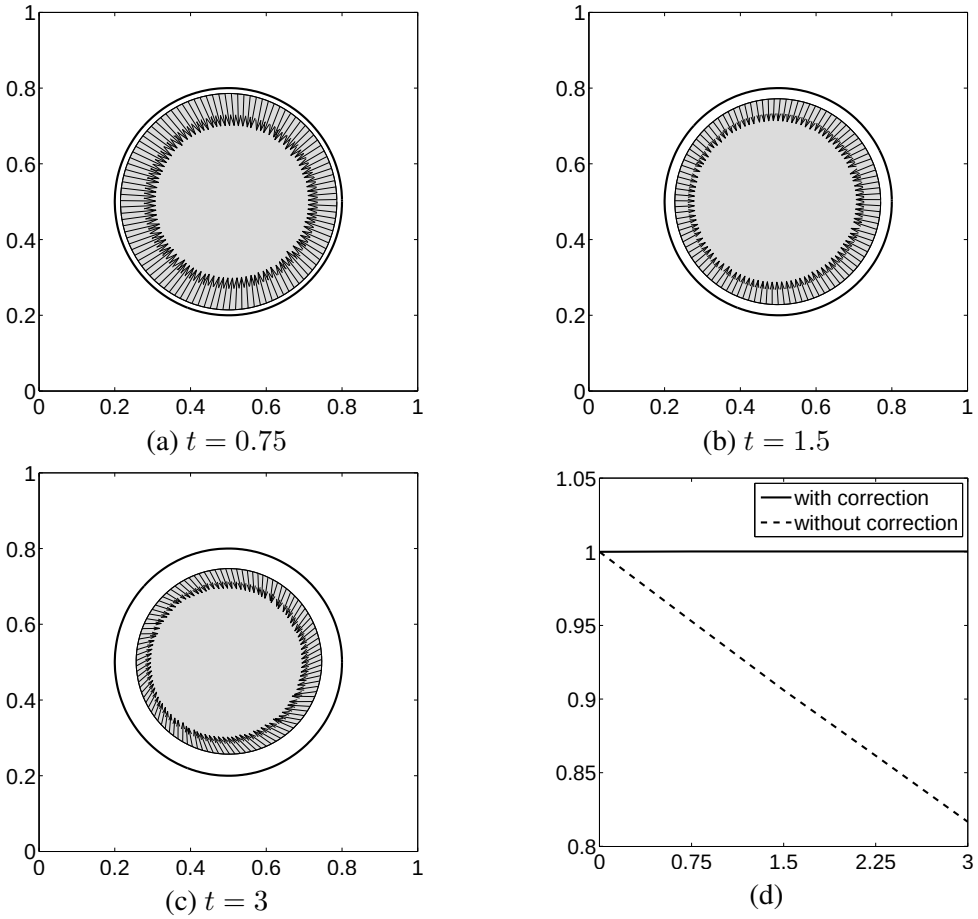


FIGURE 8.7. (a), (b), and (c) show evolutions of the static circular drop with the surface tension. (d) illustrates the area loss rates with and without correction.

area is more conservative when the time step is smaller. While the simulation results for the case with correction algorithm are more qualitatively similar to the theoretical values without area errors.

TABLE 8.2. Area error without correction

Time step	$\Delta t = h/128$	$\Delta t = h/16$
$A_{error}(\mathbf{X})$	1.94%	16.61%

8.3.6. Buoyancy-driven flow. In this section, we consider the buoyancy-driven flow. The initial bubble with radius $r = 0.1$ is positioned at $(0.5, 0.5)$ in the computational domain $\Omega = (0, 1) \times (0, 2)$ with 128×256 mesh grids. The densities are $\rho_1 = 1000$ and $\rho_2 = 500$

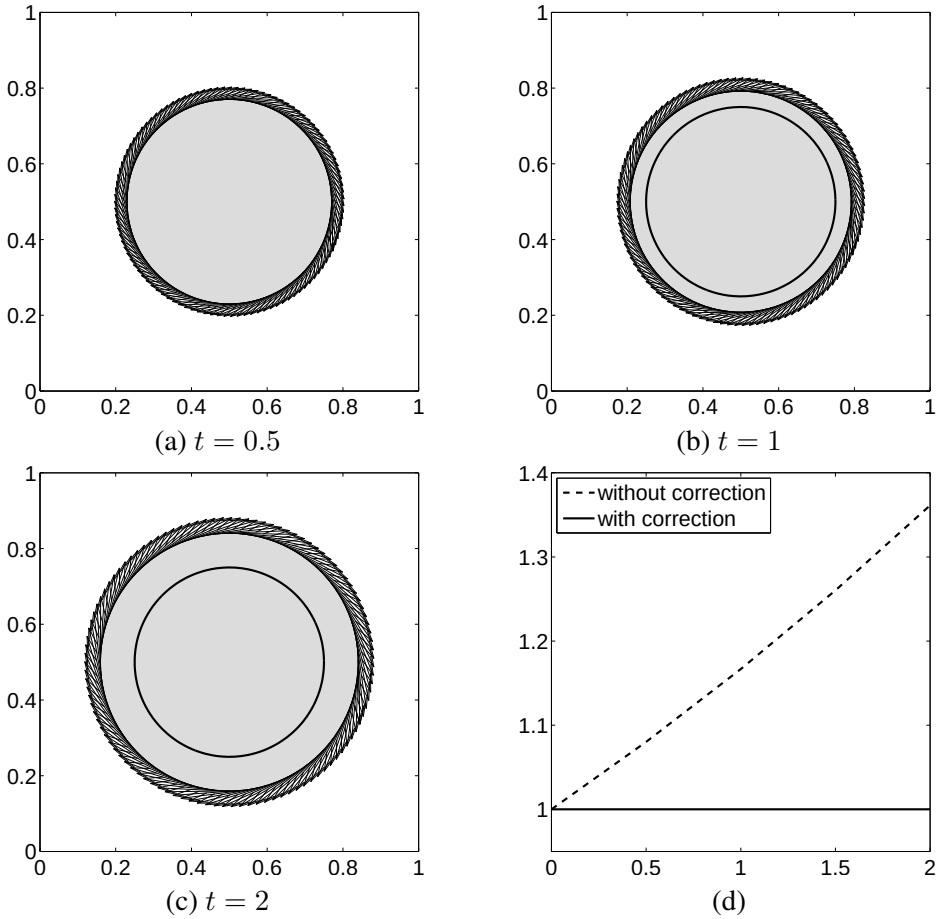


FIGURE 8.8. (a), (b), and (c) show evolutions of the static circular drop with rotation. (d) illustrates the area change rates with and without correction.

(ρ_1 and ρ_2 are the densities outside and inside the bubble, respectively). Other parameters are defined as following: $\mu = 0.01$, $\sigma = 0.1$, $g = 9.81e-3$, $h = 1/128$, and $\Delta t = 0.01$. To solve the buoyancy-driven flow, we use a periodic boundary condition described in above mentioned to vertical boundaries and no slip boundary condition to the top and bottom domain. Therefore,

$$\mathbf{n} \cdot \nabla_d p^{n+1} = \mathbf{n} \cdot \rho^n \mathbf{g}, \text{ i.e., } p_y = -\rho^n g \text{ at } y = 0 \text{ and } y = 2.$$

Fig. 8.10 shows that the bubble starts to rise due to the effect of buoyancy in the cylinder and it eventually deforms to a steady-state shape. The times are shown below each figure and we get the similar results with [135].

8.4. Conclusions

We proposed a simple area preserving correction scheme for the two-phase immiscible incompressible Navier-Stokes flows with an immersed boundary method. The idea is to correct the interface location normal to the interface so that the area remains a constant. Various

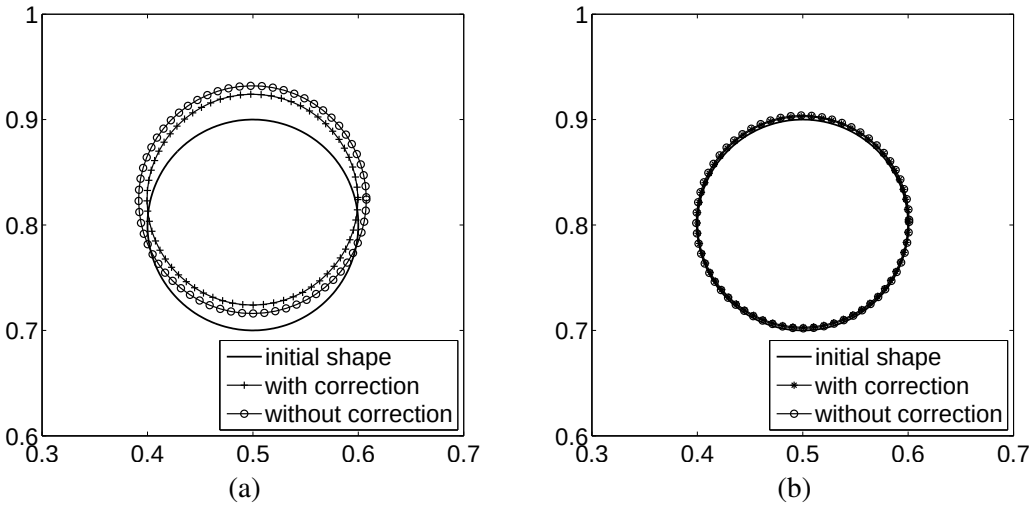


FIGURE 8.9. A rotated circle with a time step (a) $\Delta t = h/16$ and (b) $\Delta t = h/128$.

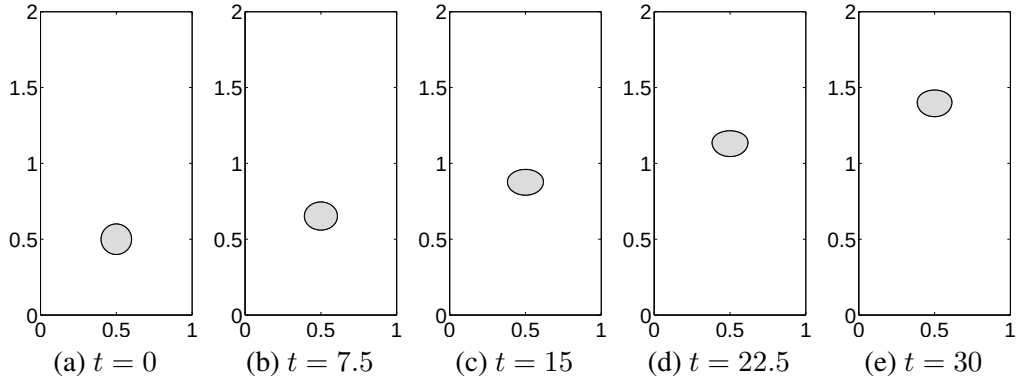


FIGURE 8.10. Instantaneous bubble shapes at different time.

numerical tests were presented to illustrate the efficiency and accuracy of our proposed scheme for two-phase fluid flows.

Chapter 9

An immersed boundary model of the growth and division of cell

9.1. Introduction

During the cell division process, cytokinesis is the final step to create the two daughter cells in the mitosis. Before the mitosis, cell doubles its mass and duplicates its DNA. The mitosis consists of prophase, metaphase, anaphase, and telophase. We mainly focus on the cytokinesis of the eukaryotic cell occurring in the telophase. For the eukaryotic cells, the position of the cleavage is usually decided at the end of the anaphase. Then the cleavage occurs with rings which are known to bisect the cell. The physical division process of the cell is shown in Fig. 9.1.

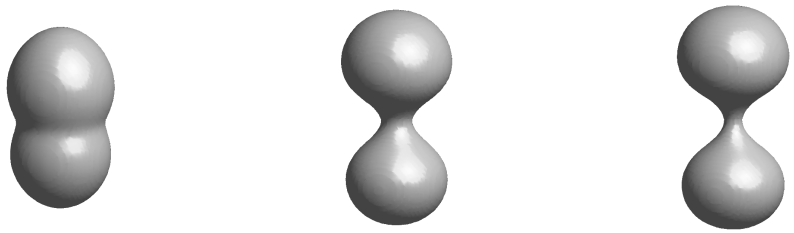


FIGURE 9.1. The evolution of the cytokinesis

In this Chapter, we adapt the polar relaxation theory by means of a difference in surface tension between the equatorial and polar regions using as surface forces [68]. This can be achieved by either increasing the tension at the equator or reducing it at the poles and both possibilities have been postulated for cleavage [133]. The idea expand to White and Borisy [76] as the computer model based on astral relaxation theory. However, it has some restriction that the shape of the cell obeys an inverse power law if it has spherically symmetric shape and the stimulatory effect of the asters ceased by the time that cell deformation began. We propose more realistic stimulus with axisymmetric three dimensional formula. Our model suggests the stimulus as difference between distance from two asters to cell membrane. With the immersed boundary method, numerical simulations show that the polar relaxation theory adapt to not only a spherically symmetric cell shape but also an elliptic cell.

9.2. Mathematical formulation

The material inside and outside the cell is modeled as a homogeneous continuum with the same constant density ρ and viscosity μ. The cell nuclei are represented by a finite set Ξ of

discrete material points $\mathbf{Y}_k$ in which sources of fluid $S_k(t)$ are located if the host cells are growing.

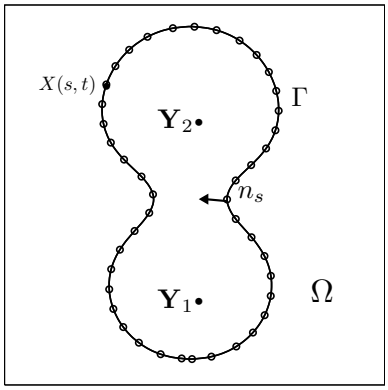


FIGURE 9.2. schematic of computational experiment.

$$\begin{aligned}\rho\left(\frac{\partial \boldsymbol{u}(\boldsymbol{x},t)}{\partial t}+\boldsymbol{u}(\boldsymbol{x},t)\cdot\nabla\boldsymbol{u}(\boldsymbol{x},t)\right) &= -\nabla p(\boldsymbol{x},t)+\mu\Delta\boldsymbol{u}(\boldsymbol{x},t)\\ &+ \left(\zeta+\frac{\mu}{3}\right)\nabla(\nabla\cdot\boldsymbol{u}(\boldsymbol{x},t))+\boldsymbol{F}(\boldsymbol{x},t), \quad (9.1)\end{aligned}$$

where $\mathbf{u} = (u, w)$ is the velocity, p is the pressure, $\mathbf{F} = (F_1, F_2)$ is the external force density. The configuration of the immersed boundary is described by the function $\mathbf{X}(s,t) = (R(s,t), Z(s,t))$, where $0 \le s \le L$ and L is the unstressed length of the boundary . If the compressible effects of the fluid flow are negligible, ζ disappears. And also in [112], aiming to the continuity for the mass of fluid flow, we have

$$\frac{\partial \rho}{\partial t}+\nabla\cdot(\rho\boldsymbol{u}(\boldsymbol{x},t)) = S(\boldsymbol{x},t),$$

where S is the fluid source distribution. Since we assumed ρ is constant, we have

$$\rho\nabla\cdot\boldsymbol{u}(\boldsymbol{x},t) = S(\boldsymbol{x},t).$$

Therefore, the governing equation is described as

$$\rho\left(\frac{\partial \boldsymbol{u}(\boldsymbol{x},t)}{\partial t}+\boldsymbol{u}(\boldsymbol{x},t)\cdot\nabla\boldsymbol{u}(\boldsymbol{x},t)\right)=-\nabla p(\boldsymbol{x},t)+\mu\Delta\boldsymbol{u}(\boldsymbol{x},t)$$

$$+\frac{\mu}{3\rho}\nabla S(\boldsymbol{x},t)+\boldsymbol{F}(\boldsymbol{x},t), \tag{9.2}$$

$$\rho\nabla\cdot\boldsymbol{u}(\boldsymbol{x},t)=S(\boldsymbol{x},t), \tag{9.3}$$

$$S(\boldsymbol{x},t)=\sum_{k\in\Xi}S_k(\boldsymbol{x},t)\delta^2(\boldsymbol{x}-\boldsymbol{Y}_k(t)), \tag{9.4}$$

$$\boldsymbol{F}(\boldsymbol{x},t)=\int_\Gamma \boldsymbol{f}(s,t)\delta^2(\boldsymbol{x}-\boldsymbol{X}(s,t))ds, \tag{9.5}$$

$$\boldsymbol{U}(\boldsymbol{X}(s,t),t)=\int_\Omega \boldsymbol{u}(\boldsymbol{x},t)\delta^2(\boldsymbol{x}-\boldsymbol{X}(s,t)), \tag{9.6}$$

$$\frac{\partial \boldsymbol{X}(s,t)}{\partial t}=\boldsymbol{U}(s,t), \tag{9.7}$$

$$\boldsymbol{U}(\boldsymbol{Y}_k(t),t)=\int_\Omega \boldsymbol{u}(\boldsymbol{x},t)\delta^2(\boldsymbol{x}-\boldsymbol{Y}_k(t))d\boldsymbol{x}, \tag{9.8}$$

$$\frac{\partial \boldsymbol{Y}_k(t)}{\partial t}=\boldsymbol{U}(\boldsymbol{Y}_k(t),t), \tag{9.9}$$

$$\boldsymbol{f}(s,t)=\sigma_1\kappa_s\mathbf{n}_s+\sigma_2 d(s)\mathbf{n}_s, \tag{9.10}$$

$$\kappa_s=\frac{1}{2}\left(\frac{R_{ss}Z_s-R_sZ_{ss}}{\sqrt{R_s^2+Z_s^2}^3}-\frac{Z_s}{R\sqrt{R_s^2+Z_s^2}}\right) \tag{9.11}$$

$$d(s)=\frac{1}{\epsilon+\left||\boldsymbol{X}(s,t_0)-\mathbf{Y}_1|-|\boldsymbol{X}(s,t_0)-\mathbf{Y}_2|\right|}, \tag{9.12}$$

where κ_s is mean curvature [100] and $\mathbf{n}_s=(m_s,n_s)$ is the unit normal vector into the cell, $\mathbf{Y}_1$ and $\mathbf{Y}_2$ are these two centers, ϵ is the small positive parameter, σ_1 is a stiffness constant for the curvature force term and σ_2 is the normal curvature force, respectively.

9.3. Numerical method

In this section, we consider only axisymmetric flows for the summary of the immersed boundary method to find a numerical solution to the system of equations (9.2)–(9.10); therefore, there is no flow in the θ (azimuthal) direction, and all θ derivatives are identically zero. Therefore, we consider only two variables, r the radial direction and z the axial direction, in the two-dimensional axisymmetric domain $\Omega=\{(r,z):0<r<R,\ -H/2<z<H/2\}$. We define the fluid velocity by the vector $\mathbf{u}=(u,w)$, where $u=u(r,z)$ is the radial component of the velocity and $w=w(r,z)$ is the component in the axial direction. Let the time proceed in steps of duration Δt, Δr and Δz as the fluid-lattice spacing, Δs as the distance between material points of the immersed boundary.

The fluid equations (9.2) and (9.3) in Eulerian form are discretized on a fixed rectangular lattice at time $t=n\Delta t$: $\boldsymbol{x}_{ik}^n=\boldsymbol{x}(i\Delta r,k\Delta z,n\Delta t)$, where $i=0,\ldots,N_r-1$, $k=0,\ldots,N_z-1$, and $n=0,1,\ldots$. The immersed boundary equations (9.7) and (9.10) in Lagrangian form are discretized on a collection of moving points in the immersed boundary at time $t=n\Delta t$:$\boldsymbol{X}_l^n=$

$\boldsymbol{X}(l\Delta s, n\Delta t)$. Here, we use a set of M Lagrangian points $\mathbf{X}_l^n = (R_l^n, Z_l^n)$ for $l = 1, \ldots, M$ to discretize the immersed boundary with the initial boundary mesh width $\Delta s_{l+1}^n = |\mathbf{X}_{l+1}^n - \mathbf{X}_l^n|$. Our goal is to compute the update $\boldsymbol{u}^{n+1}$, $\boldsymbol{X}^{n+1}$, $\boldsymbol{Y}_k^{n+1}$ from given $\boldsymbol{u}^n$, $\boldsymbol{X}^n$, $\boldsymbol{Y}_k^n$. This is done as follows:

For simplicity, choose $\Delta r = \Delta z = R/N_r = H/N_z = h$.

Step 1. Find the force $\mathbf{f}^n$ on the immersed boundary from the given boundary configuration $\mathbf{X}^n$. For $l = 1, \ldots, M$,

$$\mathbf{f}_l^n = \sigma_1 \kappa_l \mathbf{n}_l + \frac{\sigma_2 \mathbf{n}_l}{\epsilon + \big||\mathbf{X}_l^n - \mathbf{Y}_1^n| - |\mathbf{X}_l^n - \mathbf{Y}_2^n|\big|}, \tag{9.13}$$

$$\kappa_l = \frac{1}{2}\left(\frac{(R_l^n)_{ss}(Z_l^n)_s - (R_l^n)_s(Z_l^n)_{ss}}{\sqrt{(R_l^n)_s^2 + (Z_l^n)_s^2}^3} - \frac{(Z_l^n)_s}{(R_l^n)\sqrt{(R_l^n)_s^2 + (Z_l^n)_s^2}} \right), \tag{9.14}$$

where $\mathbf{n}_l = (m_l, n_l)$ is the unit normal vector. It should be said that we will give a new effective numerical method for computing unit normal vector in later subsection 9.3.2. And also

$$\begin{aligned} (R_l^n)_s &= \left(\frac{\Delta s_l^n (R_{l+1}^n - R_l^n)}{2\Delta s_{l+1}^n} + \frac{\Delta s_{l+1}^n (R_l^n - R_{l-1}^n)}{2\Delta s_l^n} \right) / \Delta s_{l+1/2}^n, \\ (R_l^n)_{ss} &= \left(\frac{R_{l+1}^n - R_l^n}{\Delta s_{l+1}^n} - \frac{R_l^n - R_{l-1}^n}{\Delta s_l^n} \right) / \Delta s_{l+1/2}^n. \end{aligned}$$

Step 2. Spread the boundary force into the nearby lattice points of the fluid.

$$\mathbf{F}_{ik}^n = \sum_{l=1}^{M} \mathbf{f}_l^n \delta_h^2(\mathbf{x}_{ik} - \mathbf{X}_l^n) \Delta s_{l+1/2}^n \text{ for } i = 1, \ldots, N_r \text{ and } k = 1, \ldots, N_z,$$

where δ_h^2 is a smoothed approximation to the two-dimensional Dirac delta function. *Step* 3. Solve the Navier-Stokes equations on the rectangular lattice to get $\boldsymbol{u}^{n+1}$ and p^{n+1} from $\boldsymbol{u}^n$ and $\boldsymbol{F}^n$.

A staggered marker-and-cell (MAC) mesh of Harlow and Welch [71] is used in which pressure is stored at cell-centers and velocities at cell interfaces (see Figure 8.1(a)). Let a computational domain be partitioned in Cartesian geometry into a uniform mesh with mesh spacing h. The center of each cell, Ω_{ik}, is located at $(r_i, z_k) = ((i - 0.5)h, (k - 0.5)h)$ for $i = 1, \cdots, M$ and $k = 1, \cdots, N$. M and N are the numbers of cells in r and z-directions, respectively. The cell vertices are located at $(r_{i+\frac{1}{2}}, z_{k+\frac{1}{2}}) = (ih, kh)$.

At the beginning of each time step, given $\boldsymbol{u}^n$ and $\boldsymbol{F}^n$, we want to find $\boldsymbol{u}^{n+1}$ and p^{n+1} which solve the following implicit first order scheme in time and space:

$$\rho\left(\frac{\boldsymbol{u}^{n+1} - \boldsymbol{u}^n}{\Delta t} + \boldsymbol{u}^n \cdot \nabla_d \boldsymbol{u}^n \right) = -\nabla_d p^{n+1} + \mu \Delta_d \boldsymbol{u}^n + \frac{\mu}{3\rho} \nabla_d S^n + \boldsymbol{F}^n, \tag{9.15}$$

$$\rho \nabla_d \cdot \boldsymbol{u}^{n+1} = S^n. \tag{9.16}$$

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

1. Initialize $\boldsymbol{u}^0$ to be the divergence-free velocity field.

2. Solve an intermediate velocity field, $\tilde{\boldsymbol{u}}$, which does not satisfy the incompressible condition, without the pressure gradient term,

$$\rho\left(\frac{\tilde{\boldsymbol{u}}-\boldsymbol{u}^n}{\Delta t}+\boldsymbol{u}^n\cdot\nabla_d\boldsymbol{u}^n\right)=\mu\Delta_d\boldsymbol{u}^n+\frac{\mu}{3\rho}\nabla_d S^n+\boldsymbol{F}^n.$$

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

$$\begin{aligned}\tilde{u}_{i+\frac{1}{2},k} &= u^n_{i+\frac{1}{2},k}-\Delta t(uu_r+wu_z)^n_{i+\frac{1}{2},k}+\frac{\mu}{3\rho h}(S^n_{i+1,j}-S^n_{i,j})+\frac{\Delta t}{\rho}F^{r-edge}_{i+\frac{1}{2},k}\\ &\quad+\frac{\mu\Delta t}{\rho h^2}\left(\frac{r_{i+\frac{1}{2}}(u^n_{i+\frac{3}{2},k}-u^n_{i+\frac{1}{2},k})-r_{i-\frac{1}{2}}(u^n_{i+\frac{1}{2},k}-u^n_{i-\frac{1}{2},k})}{r_i h^2},\right.\\ &\quad\left.-\frac{u^n_{i+\frac{1}{2},k}}{r_i^2}+\frac{u^n_{i+\frac{1}{2},k+1}-2u^n_{i+\frac{1}{2},k}+u^n_{i+\frac{1}{2},k-1}}{h^2}\right),\end{aligned}\tag{9.17}$$

$$\begin{aligned}\tilde{v}_{i,k+\frac{1}{2}} &= w^n_{i,k+\frac{1}{2}}-\Delta t(uw_r+ww_z)^n_{i,k+\frac{1}{2}}+\frac{\mu}{3\rho h}(S^n_{i,j+1}-S^n_{i,j})+\frac{\Delta t}{\rho}F^{z-edge}_{i,k+\frac{1}{2}}\\ &\quad+\frac{\mu\Delta t}{\rho h^2}\left(\frac{r_{i+\frac{1}{2}}(w^n_{i,k+\frac{3}{2}}-w^n_{i,k+\frac{1}{2}})-r_{i-\frac{1}{2}}(w^n_{i,k+\frac{1}{2}}-w^n_{i,k-\frac{1}{2}})}{r_i h^2},\right.\\ &\quad\left.+\frac{w^n_{i+1,k+\frac{1}{2}}-2w^n_{i,k+\frac{1}{2}}+w^n_{i-1,k+\frac{1}{2}}}{h^2}\right),\end{aligned}\tag{9.18}$$

where the advection terms, $(uu_r+wu_z)^n_{i+\frac{1}{2},k}$ and $(uw_r+vw_z)^n_{i,k+\frac{1}{2}}$, are defined by

$$\begin{aligned}(uu_r+wu_z)^n_{i+\frac{1}{2},k} &= u^n_{i+\frac{1}{2},k}\bar{u}^n_{r_{i+\frac{1}{2},k}}\\ &\quad+\frac{w^n_{i,k-\frac{1}{2}}+w^n_{i+1,k-\frac{1}{2}}+w^n_{i,k+\frac{1}{2}}+w^n_{i+1,k+\frac{1}{2}}}{4}\bar{u}^n_{z_{i+\frac{1}{2},k}},\\ (uw_r+ww_z)^n_{i,k+\frac{1}{2}} &= w^n_{i,k+\frac{1}{2}}\bar{w}^n_{z_{i,k+\frac{1}{2}}}\\ &\quad+\frac{u^n_{i-\frac{1}{2},k}+u^n_{i-\frac{1}{2},k+1}+u^n_{i+\frac{1}{2},k}+u^n_{i+\frac{1}{2},k+1}}{4}\bar{w}^n_{r_{i,k+\frac{1}{2}}}.\end{aligned}$$

The values $\bar{u}^n_{r_{i+\frac{1}{2},k}}$ and $\bar{u}^n_{z_{i+\frac{1}{2},k}}$ are computed using the upwind procedure. The procedure is

$$\bar{u}^n_{r_{i+\frac{1}{2},k}} = \begin{cases}\frac{u^n_{i+\frac{1}{2},k}-u^n_{i-\frac{1}{2},k}}{h} & \text{if } u^n_{i+\frac{1}{2},k}>0\\ \frac{u^n_{i+\frac{3}{2},k}-u^n_{i+\frac{1}{2},k}}{h} & \text{otherwise}\end{cases}$$

and

$$\bar{u}^n_{z_{i+\frac{1}{2},k}} = \begin{cases}\frac{u^n_{i+\frac{1}{2},k}-u^n_{i+\frac{1}{2},k-1}}{h} & \text{if } w^n_{i,k-\frac{1}{2}}+w^n_{i+1,k-\frac{1}{2}}+w^n_{i,k+\frac{1}{2}}+w^n_{i+1,k+\frac{1}{2}}>0\\ \frac{u^n_{i+\frac{1}{2},k+1}-u^n_{i+\frac{1}{2},k}}{h} & \text{otherwise.}\end{cases}$$

The quantities $\bar{w}^n_{r_{i,k+\frac{1}{2}}}$ and $\bar{w}^n_{z_{i,k+\frac{1}{2}}}$ are computed in a similar manner.

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

$$\rho\frac{\boldsymbol{u}^{n+1}-\tilde{\boldsymbol{u}}}{\Delta t} = -\nabla_d p^{n+1}, \tag{9.19}$$

$$\rho\nabla_d\cdot\boldsymbol{u}^{n+1} = S^n. \tag{9.20}$$

With application of the divergence operator to (9.19), we find that the Poisson equation for the pressure at the advanced time $(n+1)$.

$$\Delta_d p^{n+1} = \frac{1}{\Delta t}\left(\rho\nabla_d\cdot\tilde{\boldsymbol{u}} - S^n\right), \tag{9.21}$$

where we have made use of the Eq. (9.20). The terms are defined as in the following.

$$\Delta_d p_{ik}^{n+1} = \frac{r_{i+\frac{1}{2}}(p_{i+1,k}^{n+1}-p_{ik}^{n+1}) - r_{i-\frac{1}{2}}(p_{i,k}^{n+1}-p_{i-1,k}^{n+1})}{r_i h^2} + \frac{p_{i,k+1}^{n+1}-2p_{i,k}^{n+1}+p_{i,k-1}^{n+1}}{h^2},$$

$$\nabla_d\cdot\tilde{\boldsymbol{u}}_{ik} = \frac{r_{i+\frac{1}{2}}\tilde{u}_{i+\frac{1}{2},k} - r_{i-\frac{1}{2}}\tilde{u}_{i-\frac{1}{2},k}}{r_i h} + \frac{\tilde{w}_{i,k+\frac{1}{2}}-\tilde{w}_{i,k-\frac{1}{2}}}{h}.$$

The resulting linear system of (9.21) is solved using a multigrid method, specifically, V-cycles with a Gauss-Seidel relaxation. Then the updated velocities u^{n+1} and w^{n+1} are defined by

$$\boldsymbol{u}^{n+1} = \tilde{\boldsymbol{u}} - \frac{\Delta t}{\rho}\nabla_d p^{n+1}, \text{ i.e.,}$$

$$u_{i+\frac{1}{2},k}^{n+1} = \tilde{u}_{i+\frac{1}{2},k} - \frac{\Delta t}{\rho h}(p_{i+1,k}-p_{ik}),\ w_{i,k+\frac{1}{2}}^{n+1} = \tilde{w}_{i,k+\frac{1}{2}} - \frac{\Delta t}{\rho h}(p_{i,k+1}-p_{ik}).$$

These complete the one time step.

Step 4. Once the updated fluid velocity, $\boldsymbol{u}^{n+1}$, has been determined, we can find the velocity, $\boldsymbol{U}^{n+1}$, and then the new immersed boundary points, $\boldsymbol{X}^{n+1}$, and the new center position, $\boldsymbol{Y}^{n+1}$. The difference approximations to the interpolation equation and no-slip condition are expressed as follows:

$$\boldsymbol{U}_l^{n+1} = \sum_{i=0}^{N-1}\sum_{k=0}^{N-1}\boldsymbol{u}_{ik}^{n+1}\delta_h^2(\boldsymbol{x}_{ik}-\boldsymbol{X}_l^n)h^2, \tag{9.22}$$

$$\boldsymbol{X}_l^{n+1} = \boldsymbol{X}_l^n + \Delta t\boldsymbol{U}_l^{n+1},\ \text{for } l=1,\ldots,M, \tag{9.23}$$

$$\boldsymbol{U}_k^{n+1} = \sum_{i=0}^{N-1}\sum_{k=0}^{N-1}\boldsymbol{u}_{ik}^{n+1}\delta_h^2(\boldsymbol{x}_{ik}-\boldsymbol{Y}_k^n)h^2, \tag{9.24}$$

$$\boldsymbol{Y}_k^{n+1} = \boldsymbol{Y}_k^n + \Delta t\boldsymbol{U}_k^{n+1},\ \text{for } k=1,2. \tag{9.25}$$

Note that we use the same delta function in Eq.9.22 and Eq.9.24 as the one in the interaction equation for the force term Eq.9.15. This completes the description of the process (Steps 1-4, above) by which the quantities $\boldsymbol{u}$, $\boldsymbol{X}$ and $\boldsymbol{Y}_k$ are updated.

9.3.1. Boundary conditions. We next specify the boundary conditions. Due to the symmetry of the flow, at the column axis $(r=0),\ u(0,z)=0,\ w_r(0,z)=0$. At the rigid wall, $r=R$, the no-slip conditions are applied, i.e., $u(R,z)=w(R,z)=0$. We will denote the domain containing the two materials as Ω and its boundary as $\partial\Omega$.

9.3.2. Discretization of unit normal vector. In this section, we present the numerical method to calculate the unit normal vector. For a given interface position $\mathbf{X}_l$, the unit normal vector $\mathbf{n}_l = (m_l, n_l)$ will be calculated by using three points $\mathbf{X}_{l-1}$, $\mathbf{X}_l$, and $\mathbf{X}_{l+1}$ with the quadratic polynomial approximation. Let the quadratic polynomial approximation be

$$R(t) = \alpha_1 t^2 + \beta_1 t + \gamma_1 \text{ and } Z(t) = \alpha_2 t^2 + \beta_2 t + \gamma_2.$$

And assume $\mathbf{X}_{l-1} = (R(0), Z(0))$, $\mathbf{X}_l = (R(\Delta s_l), Z(\Delta s_l))$, and $\mathbf{X}_{l+1} = (R(\Delta s_l + \Delta s_{l+1}), Z(\Delta s_l + \Delta s_{l+1}))$, then the parameters α_1, β_1, γ_1, α_2, β_2, and γ_2 can be calculated by the following equations:

$$\begin{pmatrix} \alpha_1 \\ \beta_1 \\ \gamma_1 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 1 \\ \Delta s_l^2 & \Delta s_l & 1 \\ (\Delta s_l + \Delta s_{l+1})^2 & \Delta s_l + \Delta s_{l+1} & 1 \end{pmatrix}^{-1} \begin{pmatrix} R(0) \\ R(\Delta s_l) \\ R(\Delta s_l + \Delta s_{l+1}) \end{pmatrix},$$
$$\begin{pmatrix} \alpha_2 \\ \beta_2 \\ \gamma_2 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 1 \\ \Delta s_l^2 & \Delta s_l & 1 \\ (\Delta s_l + \Delta s_{l+1})^2 & \Delta s_l + \Delta s_{l+1} & 1 \end{pmatrix}^{-1} \begin{pmatrix} Z(0) \\ Z(\Delta s_l) \\ Z(\Delta s_l + \Delta s_{l+1}) \end{pmatrix}.$$

Now we get the unit normal vector as

$$\mathbf{n}_l = (m_l, n_l) = \left(\frac{\frac{dZ(\Delta s_l)}{dt}}{\sqrt{\left(\frac{dR(\Delta s_l)}{dt}\right)^2 + \left(\frac{dZ(\Delta s_l)}{dt}\right)^2}}, \frac{-\frac{dR(\Delta s_l)}{dt}}{\sqrt{\left(\frac{dR(\Delta s_l)}{dt}\right)^2 + \left(\frac{dZ(\Delta s_l)}{dt}\right)^2}} \right).$$

9.3.3. Discretization of interpolation. In this section, we present the numerical method to calculate the unit normal vector. For a given interface position $\mathbf{X}_l$ which satisfied that $\Delta s_{l+1} > 0.5h$, the wanted $\bar{\mathbf{X}}_l = 0.5(\bar{\mathbf{X}}_1 + \bar{\mathbf{X}}_2)$. $\bar{\mathbf{X}}_1$ will be calculated by using three points $\mathbf{X}_{l-1}$, $\mathbf{X}_l$, $\mathbf{X}_{l+1}$ with the quadratic polynomial approximation. And in the same way, $\bar{\mathbf{X}}_2$ will be calculated by using three points $\mathbf{X}_l$, $\mathbf{X}_{l+1}$, $\mathbf{X}_{l+2}$. For simplicity of exposition, we only give the form of computing the first part $\bar{\mathbf{X}}_1$ and $\bar{\mathbf{X}}_2$ will computed with the same method. Let the quadratic polynomial approximation be

$$R(t) = \alpha_1 t^2 + \beta_1 t + \gamma_1 \text{ and } Z(t) = \alpha_2 t^2 + \beta_2 t + \gamma_2.$$

And assume $\mathbf{X}_{l-1} = (R(0), Z(0))$, $\mathbf{X}_l = (R(\Delta s_l), Z(\Delta s_l))$, and $\mathbf{X}_{l+1} = (R(\Delta s_l + \Delta s_{l+1}), Z(\Delta s_l + \Delta s_{l+1}))$, then the parameters α_1, β_1, γ_1, α_2, β_2, and γ_2 can be calculated by the following equations:

$$\begin{pmatrix} \alpha_1 \\ \beta_1 \\ \gamma_1 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 1 \\ \Delta s_l^2 & \Delta s_l & 1 \\ (\Delta s_l + \Delta s_{l+1})^2 & \Delta s_l + \Delta s_{l+1} & 1 \end{pmatrix}^{-1} \begin{pmatrix} R(0) \\ R(\Delta s_l) \\ R(\Delta s_l + \Delta s_{l+1}) \end{pmatrix},$$
$$\begin{pmatrix} \alpha_2 \\ \beta_2 \\ \gamma_2 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 1 \\ \Delta s_l^2 & \Delta s_l & 1 \\ (\Delta s_l + \Delta s_{l+1})^2 & \Delta s_l + \Delta s_{l+1} & 1 \end{pmatrix}^{-1} \begin{pmatrix} Z(0) \\ Z(\Delta s_l) \\ Z(\Delta s_l + \Delta s_{l+1}) \end{pmatrix}.$$

Now we get

$$\bar{\mathbf{X}}_1 = \left(\frac{dr(\Delta s_l + 0.5\Delta s_{l+1})}{dt}, \frac{dz(\Delta s_l + 0.5\Delta s_{l+1})}{dt} \right),$$

with the same method, we compute these two polynomial approximations by assuming $\mathbf{X}_l = (R(0), Z(0))$, $\mathbf{X}_l = (R(\Delta s_{l+1}), Z(\Delta s_{l+1}))$, and $\mathbf{X}_{l+2} = (R(\Delta s_{l+1}+\Delta s_{l+2}), Z(\Delta s_{l+1}+\Delta s_{l+2}))$ and get

$$\bar{\mathbf{X}}_2 \quad = \quad \left(\frac{dR(0.5\Delta s_{l+1})}{dt}, \frac{dZ(0.5\Delta s_{l+1})}{dt} \right).$$

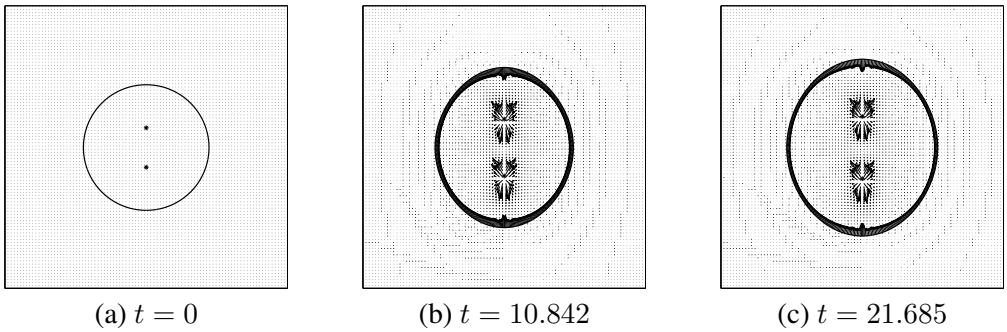
Then the wanted position $\bar{\mathbf{X}}_l = 0.5(\bar{\mathbf{X}}_1 + \bar{\mathbf{X}}_2)$ will be offered. And also the unit normal vector for wanted position is provided as $\bar{\mathbf{n}}_l = 0.5(\mathbf{n}_l + \mathbf{n}_{l+1})$.

9.4. Numerical examples

In this section, we perform a number of numerical experiments to investigate the effect of our algorithm for simulating the growth of cell with the following stopping algorithm. We stop the numerical computations when the distance between the two nearest boundary positions becomes less than a given tolerance, tol. The termination criterion algorithm is listed as follows:

Set a maximum iteration number N, a tolerance tol, and $n = 1$.
While $(n \leq N)$ do *Steps 1-2*
Step 1 Do the above main procedures and compute $\boldsymbol{X}^n$.
Step 2 Get the nearest two positions named α and β, respectively.
If $||X_\alpha - X_\beta|| < tol$, then stop the calculation.
Else $n = n + 1$.

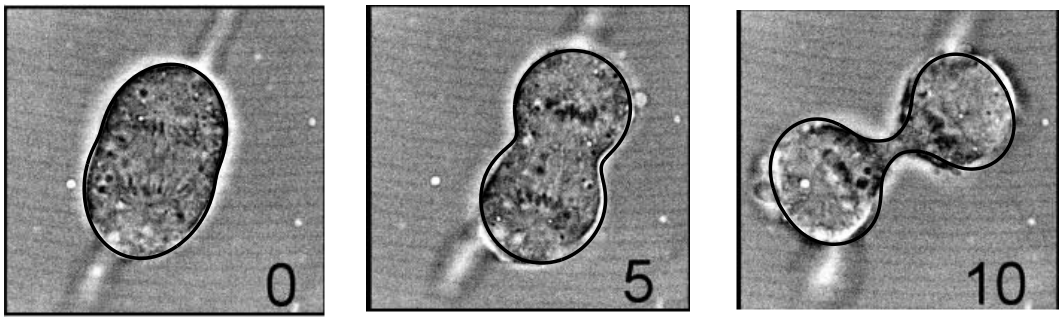
9.4.1. Cell proliferation. A process of cell proliferation is modeled by introducing two point sources inside the cell that correspond to two sets of sister chromatids (Fig. 9.3 (a)). These sources are located along the cell's longest axis. Fluid created by both sources causes cell growth by pushing cell boundaries and by increasing the cell area (Fig. 9.3 (b) and (c)). The sources are deactivated when the cell area is doubled. At this time, two opposite points on the cell boundary are selected in such a way that a line provided between them separates the cell into two parts of approximately equal areas, each with its own nucleus (former point sources). The contractile ring is created by attaching the contractile forces to the opposite points on the cell boundary. This results in a formation of the contractile furrow (Fig. 9.3 (d) and (e)) and causes division of the cell into two daughter cells (Fig. 9.3 (f)). We use the simulation parameters computational domain $(0, \pi) \times (-\pi, \pi)$ with 64×128 mesh grids, time step $\Delta t = h^2$, $\sigma_1 = 0.05$, $\sigma_2 = 0.1$, and $\epsilon = 0.01$, respectively.



(a) $t = 0$ (b) $t = 10.842$ (c) $t = 21.685$

(d) $t = 22.889$ (e) $t = 24.094$ (f) $t = 26.503$

FIGURE 9.3. Main phases of the cell cycle: (a) cell ready to proliferate, (b) and (c) cell growth and elongation, (d) and (e) formation of the contractile ring, (f) cellular division.



(a) t=1.204 (b) t=7.228 (c) t=22.890

FIGURE 9.4. The top row plots experimental data from [13]. In the second, the numerical results are superposed with experimental data.

9.4.2. Comparison with the experimental data. Next, we compare our numerical simulation results with experimental data by E. Boucrot et al. [13]. We set $\sigma_1 = 5$, $\sigma_2 = 0.1$, $\epsilon = 0.1$, and $Re = 0.5$ on the computational domain $\Omega = (0, \pi) \times (-\pi, \pi)$ with a 64×128 mesh. Calculation is run up to time $T = 0.854$ with a time step $\Delta t = 0.1h^2$. For this initial shape, we use the technology of image segmentation to get the edge of cell as the initial shape shown in the figure of 9.4(a). As can be seen, these results show that our computational results are in qualitative agreement with experimental data.

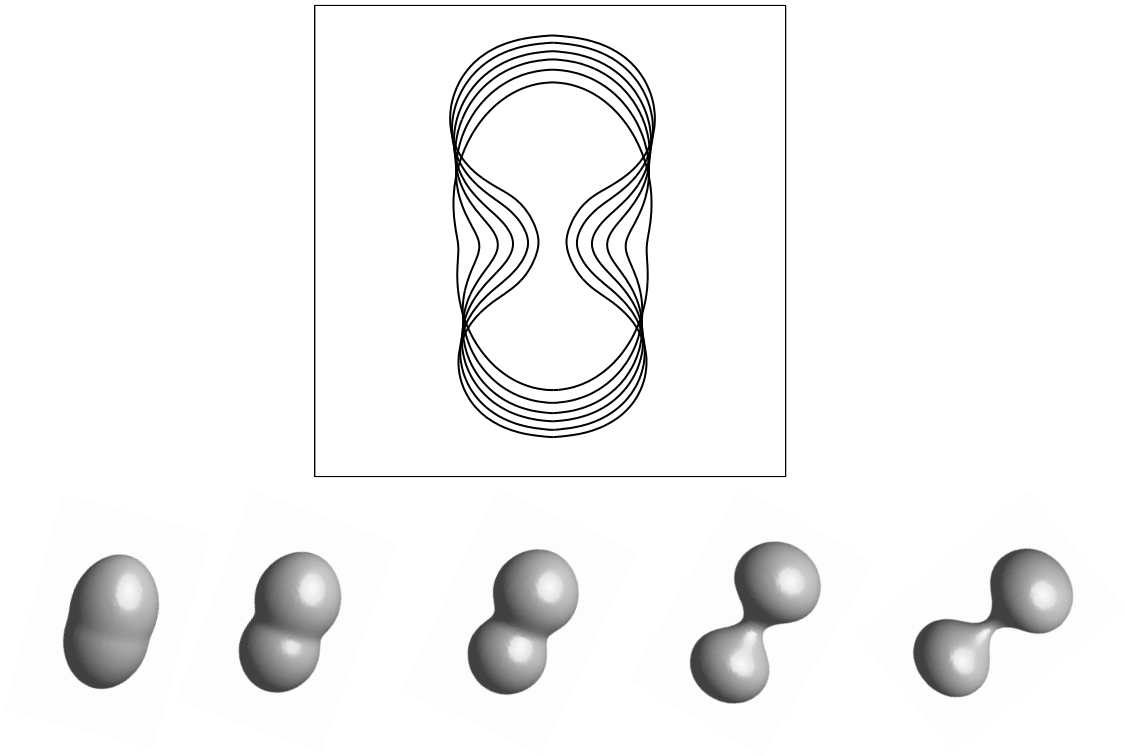


FIGURE 9.5. Evolution of the immersed boundary. The first row is the simulation shown in two dimensions and the others is shown in three dimensions.

Chapter 10

Conclusions

This thesis described various numerical methods for hybrid phase-field and immersed boundary methods. In the first part of this thesis, we present an unconditionally stable second-order hybrid numerical method for solving the Allen-Cahn equation representing a model for antiphase domain coarsening in a binary mixture. Then, we applied this hybrid method to the application of binary image segmentation, geometric image segmentation, multiphase image segmentation, and the simulation of crystal growth. In the second part, we considerd the simulations of thin film based on Navier-Stokes flows by implicit ENO type scheme with a good stability property. Secondly, IBM for two-phase fluid flows was considered. Since the interface between two fluids is moved in a discrete manner, this can result in a lack of volume conservation. We proposed a volume correction scheme. The idea of area preserving correction scheme is to correct the interface location normally to the interface so that the area remains constant. Finally, we considered the cytokinesis of an animal cell using the IBM.

Bibliography

[1] S.M. Allen and J.W. Cahn, A microscopic theory for antiphase boundary motion and its application to antiphase domain coarsening, Acta Metall. 27 (1979) 1085–1095.

[2] G.R. Arce, J.L. Paredes, and J. Mullan, Nonlinear Filtering for Image Analysis and Enhancement, in Handbook of Image and Video Processing, Academic Press, 2000.

[3] B. Appleton and H. Talbot, Globally optimal geodesic active contours, Journal of Mathematical Imaging and Vision, 23 (2005) 67–86.

[4] N. Al-Rawahi and G. Tryggvason, Numerical simulation of dendritic solidification with convection: two-dimensional geometry, J. Comput. Phys. 180 (2002) 471–496.

[5] W. Bao, Approximation and comparison for motion by mean curvature with intersection points, Comput. Math. App. 46 (2003) 1211–1228.

[6] A.L. Bertozzi and M.P. Brenner, Linear stability and transient growth in driven contact lines, Phys. Fluids, (1997) 530–539.

[7] M. Beneš, V. Chalupecký, and K. Mikula, Geometrical image segmentation by the Allen-Cahn equation, Appl. Numer. Math. 51 (2004) 187–205.

[8] R.L. Burden and J.D. Faires, Numerical Analysis, Thomson, 2004.

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

[10] R.L. Burden and Faires, Numerical Analysis (4th Edition ed.) Prindle, Weber & Schmidt, London, UK.

[11] W.L. Briggs, A multigrid tutorial, SIAM, Philadelphia, PA, 1987.

[12] M. Burger, L. He, and C. Schöenlieb, Cahn-Hilliard inpainting and a generalization for grayvalue images, UCLA CAM report 08, 2008.

[13] E. Boucrot and K. Tomas, Endosomal recycling controls plasma membrane area during mitosis, 104 (19) 2007 7939–7944.

[14] V.G. Brunet and B. Lameyre, Object recognition and segmentation in videos by connecting heterogeneous visual features, Comput. Vis. Image Underst. 111 (1) (2008) 86–109.

[15] M. Beneš and K. Mikula, Simulation of anisotropic motion by mean curvature-comparison of phase field and sharp interface approaches, Acta Math. Univ. Comenian. 67 (1998) 17-42.

[16] A.L. Bertozzi, A. Münch, X. Fanton, and A.M. Cazabat, Contact line stability and "Undercompressive shocks" in driven thin film flow. Physical Review Letters, 81 (1998) 5169–5172.

[17] A.L. Bertozzi, A. Münch, and M. Shearer, Undercompressive shocks in thin film flows, Physica D, 134 (1999) 431–464.

[18] W.L. Briggs, A Multigrid Tutorial, SIAM, Philadelphia, 1987.

[19] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, Image inpainting, Proceedings of Siggraph 2000,New Orleans (2000).

[20] G. Caginalp, Stefan and Hele-Shaw type models as asymptotic limits of the phase-field equations, Phys. Rev. A 39 (1989) 5887–5896.

[21] V. Caselles, F. Catté, T. Coll, and F. Dibos, A geometric model for active contours in image processing, Numerische Mathematik, 66 (1993) 1–31.

[22] T.F. Chan, S. Esedoḡlu, and M. Nikolova, Algorithms for finding global minimizers of image segmentation and denoising models, SIAM J. Appl. Math. 66 (5) (2006) 1632–1648.

[23] J.-W. Choi, H.G. Lee, D. Jeong, and J. Kim, An unconditionally gradient stable numerical method for solving the Allen-Cahn equation, Phys. A. 388 (2009) 1791–1803.

[24] Y. Chang, T. Hou, B. Merriman, and S. Osher, A level set formulation of Eulerian interface capturing methods for incompressible fluid flows, J. Comput. Phys. 124 (1996) 449–464.
[25] V. Caselles, R. Kimmel, and G. Sapiro, Geodesic active contours, International journal of computer vision, 22 (1) (1997) 61–79.
[26] V. Cristini and J. Lowengrub, Three-dimensional crystal growth-II: nonlinear simulation and control of the Mullins-Sekera instability, J. Crystal Growth. 266 (2004) 552-567.
[27] S. Chen, B. Merriman, S. Osher, and P. Smereka, A simple level set method for solving Stefan problem, J. Comput. Phys. 135 (1997) 8–29.
[28] S.M. Cox and P.C. Matthews, Exponential time differencing for stiff systems, J. Comput. Phys. 176 (2002) 430-455.
[29] C. Cowan, The Cahn-Hilliard equation as a gradient flow, Simon Fraser University, Canada, 2005.
[30] C.I. Christov, J. Pontes, D. Walgraef, and M.G. Velarde, Implicit time-splitting for fourth-order parabolic equations. Computer Methods in Applied Mechanics and Engineering, 148(1997)209–224.
[31] M. Cheng and A.D. Rutenberg, Maximally fast coarsening algorithms, Phys. Rev. E 72 (2005) 055701(R).
[32] T. Chinyoka, Y.Y. Renardy, M. Renardy, and D.B. Khismatullin, Two-dimensional study of drop deformation under simple shear for Oldroyd-B liquids, J. Non-Newtonian Fluid Mech. 130 (2005) 45-56.
[33] T.F. Chan and J. Shen, Mathematical models for local non-texture inpaintings, SIAM J. Appl. Math. 62 (2001) 1019–1043.
[34] C. Cowan and M.S. Thesis, Simon Fraser University, Canada, 2005.
[35] C.C. Chen and Y.L. Tsai, C.W. Lan, Adaptive phase field simulation of dendritic crystal growth in a forced flow: 2D vs. 3D morphologies, Int. J. Heat Mass Transfer 52 (2009) 1158–1166.
[36] T. Chan and L. Vese, Active contours without edges, IEEE Trans. Image Process. 10 (2) (2001) 266–277.
[37] T.F. Chan and L.A. Vese, A multiphase level set framework for image segmentation using the Mumford and Shah model, Int. J. Comput. Vis., 50 (3) (2002) 271–293.
[38] M. Cheng and J.A. Warren, An efficient algorithm for solving the phase field crystal model, J. Comput. Phys. 227 (2008) 6241-6248.
[39] P.-R. Cha, D.-H. Yeon, and S.-H. Chung, Phase-field study for the splitting mechanism of coherent misfitting precipitates in anisotropic elastic media, Scripta Mater. 52 (2005) 1241–1245.
[40] O. Dorok, Eine stabilisierte Finite-Elemente-Methode zur Lösung der Boussinesq-Approximation der Navier-Stokes-Gleichungen, Ph.D. thesis, Otto-von-Guericke-Universitat, 1995.
[41] J.A. Dobrosotskaya and A.L. Bertozzi, A Wavelet-Laplace variational technique for image deconvolution and inpainting, IEEE. Trans. Imag. Proc. 17 (2008) 657–663.
[42] N. Daniels, P. Ehret Gaskel, P.H. Thompson, and H.M. Decré M. Multigrid methods for thin liquid film spreading flows. Proceedings of the first international conference on computational fluid dynamics, Springer, (2001) 279–284.
[43] J.A. Diez and L. Kondic, Contact line instabilities of thin liquid films.Physical Review Letters 86 (2001) 632–635.
[44] J.-M. Debierre, A. Karma, F. Celestini, and R. Guérin, Phase-field approach for faceted solidification, Phys. Rev. E 68 (2003) 041–604.
[45] D.J. Duffy, Finite difference methods in financial engineering: A partial differential equation approach, Wiley Finance, West Sussex, England, 2006.
[46] Q. Du and W. Zhu, Stability analysis and applications of the exponential time differencing schemes, J. Comput. Math. 22 (2004) 200–209.
[47] D.J. Eyre, Computational and mathematical models of microstructural evolution, The Materials Research Society, Warrendale, 1998.
[48] D.J. Eyre, http://www.math.utah.edu/˜eyre/research/methods/stable.ps
[49] L.C. Evans, H.M. Soner, and P.E. Souganidis, Phase transitions and generalized motion by mean curvature, Comm. Pure Appl. Math. 45 (1992) 1097–1123.
[50] S. Esedo$\bar{g}$lu and Y.H.R. Tsai, Threshold dynamics for the piecewise constant Mumford-Shah functional, J. Comput. Phys. 211 (1) (2006) 367–384.
[51] P.C. Fife, Dynamics of internal layers and diffusive interfaces, SIAM, Philadelphia, PA, 1988.
[52] X. Feng and A. Prohl, Numerical analysis of the Allen-Cahn equation and approximation for mean curvature flows, Numer. Math. 94 (2003) 33–65.

[53] M. Francois and W. Shyy, Computations of drop dynamics with the immersed boundary method, Part 1: Numerical algorithm and buoyancy-induced effect, Numer. Heat Transfer B 44 (2003) 101–118.
[54] M. Francois, E. Uzgoren, J. Jackson, and W. Shyy, Multigrid computations with the immersed boundary technique for multiphase flows, Int. J. Numer. Meth. Heat Fluid Flow 14 (2004) 98–115.
[55] X. Feng and H.-J. Wu, A posteriori error estimates and an adaptive finite element method for the Allen-Cahn equation and the mean curvature flow, J. Sci. Comput. 24 (2005) 121–146.
[56] W.M. Feng, P. Yu, S.Y. Hu, Z.K. Liu, Q. Du, and L.Q. Chen, Spectral implementation of an adaptive moving mesh method for phase-field equations, J. Comput. Phys. 220 (2006) 498–510.
[57] J.-F. Gerbeau, C. le Bris, and M. Bercovier, Spurious velocities in the steady flow of an incompressible fluid subjected to external forces, Int. J. Numer. Meth. Fluids 25 (1997) 679–695.
[58] F. Gibou, R. Fedkiw, R. Caflisch, and S. Osher, A level set approach for the numerical simulation of dendritic growth, J. Sci. Comput. 19 (2002) 183–199.
[59] J. Grooss and J. Hesthaven, A level set discontinuous Galerkin method for free surface flows, Comput. Methods Appl. Mech. Engrg. 195 (2006) 3406–3429.
[60] E. Guyon, J.-P. Hulin, L. Petit, and C.D. Mitescu, Physical Hydrodynamics, Oxford University Press, 2001.
[61] J. Glimm, J. Grove, X. Li, K.-M. Shyue, Q. Zhang, and Y. Zeng, Three-dimensional front tracking, SIAM. J. Sci. Comput. 19 (1998) 703–727.
[62] P.H. Gaskell, P.K. Jimack, M. Sellier, and H.M. Thompson, Efficient and accurate time adaptive multigrid simulations of droplet spreading. International Journal for Numerical Methods in Fluids (45)(2004) 1161–1186.
[63] P.H. Gaskell, P.K. Jimack, M. Sellier, M.C.T..Wilson, and H.M. Thompson, Gravity-driven flow of continuous thin liquid films on non-porous substrates with topography. Journal of Fluid Mechanics (509) (2004) 253–280.
[64] P.H. Gaskell, P.K. Jimack, M. Sellier, and H.M. Thompson, Flow of evaporating, gravity-driven thin liquid films over topography. Physics of Fluids (18)(2006) 031601-1–031601-14.
[65] P. Gresho, R. Lee, S. Chan, and J. Leone, A new finite element for incompressible or Boussinesq fluids, in: Proc. Third Int. Conf. on Finite Elements in Flow Problems, Banff, Canada, 1980, 204–215.
[66] S. Ganesan, G. Matthies, and L. Tobiska, On spurious velocities in incompressible flow problems with interfaces, Comput. Methods Appl. Mech. Engrg. 196 (2007) 1193–1202.
[67] A. Goshtasby and M. Satter, An adaptive window mechanism for image smoothing, Comput. Vis. Image Underst, 111 (2) (2008) 155–169.
[68] Y. Hiramoto, Mechanical properties of sea urchin eggs. Exp. Cell Res. 32 (1963) 5-75.
[69] Y. Ha, Y.-J. Kim, and T.G. Myers, On the numerical solution of a driven thin film equation, J. Comput. Phys. 227 (2008) 7246–7263.
[70] J. Hahn and C-O. Lee, Geometric attraction-driven flow for image segmentation and boundary detection, Journal of Visual Communication and Image Image Representation, 21 (2010), 56–66.
[71] F. Harlow and J. Welch, Numerical calculations of time dependent viscous incompressible flow with free surface, Phys. Fluids 8 (1965) 2182–2189.
[72] C.H. Ho and Y.C. Tai, Micro-electro-mechanical-systems (MEMS) and fluid flows, Annu. Rev. Fluid Mech. 30 (1998) 579–612.
[73] T. Ihle, Competition between kinetic and surface tension anisotropy in dendritic growth, Eur. Phys. J. B 16 (2000) 337–344.
[74] T. Ilmanen, Convergence of the Allen-Cahn equation to Brakke's motion by mean curvature, J. Diff. Geom. 38 (1993) 417–461.
[75] D. Juric and G. Tryggvason, A front-tracking method for dendritic solidification, J. Comput. Phys. 123 (1996) 127-148.
[76] J. G. White and G. G. Borisy, On the Mechanisms of Cytokinesis in Animal Cells, J. theor. Biol. (101) (1983) 289–316.
[77] J.-H. Jeong and N. Goldenfeld, J.A. Dantzig, Phase field model for three-dimensional dendritic growth with fluid flow, Phys. Rev. E 64 (2001) 041–602.
[78] Y.M. Jung, S.H. Kang, and J. Shen, Multiphase image segmentation via Modica-Mortola phase transition, SIAM Appl. Math., 67(2007) 1213–1232.
[79] A. Jacot and M. Rappaz, A pseudo-front tracking technique for the modelling of solidification microstructures in multi-component alloys, Acta Mater. 50 (2002) 1909–1926.

[80] J.S. Kim and H.-O. Bae, An unconditionally stable adaptive mesh refinement for Cahn-Hilliard equation, in review, 2007.
[81] J. Kim, Adaptive mesh refinement for thin-film equations. Journal of the Korean Physical Society, (49) (2006)1903–1907.
[82] J. Kim, A continuous surface tension force formulation for diffuse-interface models, J. Comput. Phys. 204 (2005) 784-804.
[83] S. Kang, Uniform-shear flow over a circular cylinder at low Reynolds numbers, J. Fluids Struct. 22 (2006) 541-555.
[84] Y.-T. Kim, N. Goldenfeld, and J. Dantzig, Computation of dendritic microstructures using a level set method, Phys. Rev. E 62 (2000) 2471-2474.
[85] S. Kichenassamy, A. Kumar, P. Olver, A. Tannenbaum, and A. Yezzi, Conformal curvature flows: From phase transitions to active vision, Archive for Rational Mechanics and Analysis, 134 (1996) 275–301.
[86] R.E. Khayat, K-T. Kim, and S. Delosquer, Influence of inertia, topography and gravity on transient axisymmetric thin-film flow. International Journal for Numerical Methods in Fluids. (45) (2004) 391–419.
[87] M. Katsoulakis, G.T. Kossioris, and F. Reitich, Generalized motion by mean curvature with Neumann conditions and the Allen-Cahn model for phase transitions, J. Geom. Anal. 5 (1995) 255-279.
[88] K.H. Karlsen and K.-A. Lie, An unconditionally stable splitting for a class of nonlinear parabolic equations, IMA J. Numer. Anal. 19 (1999) 609-635.
[89] A. Karma, Y.H. Lee, and M. Plapp, Three-dimensional dendrite-tip morphology at low undercooling, Phys. Rev. E 61 (2000) 3996-4006.
[90] R. Kobayashi, Modeling and numerical simulations of dendritic crystal growth, Phys. D 63 (1993) 410-423.
[91] Y. Kim and C. Peskin, 3-D Parachute simulation by the immersed boundary method, Comput. Fluids 38 (2009) 1080-1090.
[92] Y. Kim and C. Peskin, Numerical study of incompressible fluid dynamics with nonuniform density by the immersed boundary method, Phys. Fluids 20 (2008) 062101.
[93] A. Karma and W.-J. Rappel, Quantitative phase-field modeling of dendritic growth in two and three dimensions, Phys. Rev. E 57 (1998) 4323-4349.
[94] A. Karma and W.-J. Rappel, Phase-field method for computationally efficient modeling of solidification with arbitrary interface kinetics, Phys. Rev. E 53 (1996) 3017–3020.
[95] J. Kim and J.Sur, A hybrid method for higher-order nonlinear diffusion equations. Communications of the Korean Mathematical Society (20) (2005) 179–193.
[96] D.E. Kataoka and S.M. Troian, A theoretical study of instabilities at the advancing front of thermally driven coating films. Journal of Colloid and Interface Science (192) (1997) 350–362.
[97] J.S. Langer, Directions in Condensed Matter (World Scientific, Singapore, 1986).
[98] C. Li, C. Xu, C. Gu, and M.D. Fox, Level set evolution without re-initialization: a new variational formulation, IEEE International Conference on Computer Vision and Pattern Recognition, San Diego, (2005) 430–436.
[99] Y. Li, H.G. Lee, D. Jeong, and J.S. Kim, An unconditionally stable hybrid numerical method for solving the Allen-Cahn equation, Comput. Math. Appl., 60 (2010) 1591-1606.
[100] M-C. Lai C-Y. Huang and Y-M. Huang, Simulating the axisymmetric interfacial flows with insoluble surfactant by immersed boundary, International Journal Numerical Analysis and Modeling Computing and Information, 1 (1) (2004)1–18
[101] D.S. Lee, M.Y. Ha, H.S. Yoon, and S. Balachandar, A numerical study on the flow patterns of two oscillating cylinders, J. Fluids Struct. 25 (2009) 263-283.
[102] S. Li, J.S. Lowengrub, and P.H. Leo, Nonlinear morphological control of growing crystals, Phys. D 208 (2005) 209–219.
[103] R. Leveque and Z. Li, Immersed interface methods for Stokes flow with elastic boundaries or surface tension, SIAM J. Sci. Comput. 18 (1997) 709–735.
[104] J. Lie, M. Lysaker, and X.C. Tai, A binary level set model and some applications for Mumford-Shah image segmentation, IEEE Trans. Image Process. 15 (4) (2006) 1171–1181.
[105] J. Lie, M. Lysaker, and X.C. Tai, A variant of the level set method and applications to image segmentation, Math. Comp., 75 (2006) 1155-1174.
[106] D. Li, R. Li, and P. Zhang, A cellular automaton technique for modelling of a binary dendritic growth with convection, Appl. Math. Modelling 31 (2007) 971-982.

[107] B. Lafaurie, C. Nardone, R. Scardovelli, S. Zaleski, and G. Zanetti, Modelling merging and fragmentation in multiphase flows with SURFER, J. Comput. Phys. 113 (1994) 134-147.
[108] B.P. Vollmayr-Lee and A.D. Rutenberg, Fast and accurate coarsening simulation with an unconditionally stable time step, Phys. Rev. E 68 (2003) 066–703.
[109] Y.C. Lee, H.M. Thompson, and P.H. Gaskell, A parallel algorithm designed for the efficient and accurate computation of thin film flow on functional surfaces containing micro-structure. Computer Physics Communications (180) (2009) 2634–2649.
[110] Y.C. Lee, H.M. Thompson, and P.H. Gaskell, An efficient adaptive multigrid algorithm for predicting thin film flow on surfaces containing localised topographic features. Computers & Fluids (36)(2007) 838–855.
[111] M.-C. Lai, Y.-H. Tseng, and H. Huang, An immersed boundary method for interfacial flows with insoluble surfactant, J. Comput. Phys. 227 (2008) 7279–7293.
[112] E. Guyon, J.P. Hulin, and L. Petit, hysical hydrodynamics Oxford University Press (2001).
[113] E.V.L. Melloa, Otton Teixeira da Silveira Filho, Numerical study of the Cahn-Hilliard equation in one, two and three dimensions, Phys. A. 347 (2005) 429–443.
[114] T.G. Myers, J.P.F. Charpin, and S.J. Chapman, The flow and solidification of a thin fluid film on an arbitrary three-dimensional surface. Physics of Fluids (24) (2002) 2788–2803.
[115] D.I. Meiron, Boundary integral formulation of the two-dimensional symmetric model of dendritic growth, Phys. D 23 (1986) 329–339.
[116] D.F. Martin, P. Colella, M. Anghel, F.L. Alexander, Adaptive mesh refinement for multiscale nonequilibrium physics, Comp. Sci. Eng. 7 (2005) 24–31.
[117] K.W. Morton, D.E, Mayers, Numerical Solution of Partial Differential Equations, Cambridge University Press, University of Cambridge, 1996.
[118] D. Mumford, J. Shah, Optimal approximation by piecewise smooth functions and associated variational problems, Commun. Pure Appl. Math. 14 (1989) 577–685.
[119] J.M. Morel and S. Solimini, Variational methods in image segmentation, Progress in Nonlinear Differential Equations and Their Applications, Birkhdiuser Boston, Cambridge, MA, 1995.
[120] I.S. McKinley and S.K. Wilson, and B.R. Duffy, Spin coating and air-jet blowing of thin viscous drops. Physics of Fluids (11) (1999) 30–47.
[121] E. Newren, A. Fogelson, R. Guy, and R. Kirby, Unconditionally stable discretizations of the immersed boundary equations, J. Comput. Phys. 222 (2007) 702–719.
[122] T. Ohtsuka, Motion of interfaces by an Allen-Cahn type equation with multiple-well potentials, Asymptotic Anal. 56 (2008) 87–123.
[123] S. Osher, J.A. Sethian, Fronts propagating with curvature-dependent speed: Algorithms based on hamilton-Jacobi formulations, J. Comput. Phys. 79 (1) (1988) 12–49.
[124] N. Provatas, N. Goldenfeld, and J. Dantzig, Adaptive mesh refinement computation of solidification microstructures using dynamic data structures, J. Comput. Phys. 148 (1999) 265–290.
[125] N. Provatas, N. Goldenfeld, and J. Dantzig, Efficient computation of dendritic microstructures using adaptive mesh refinement, Phys. Rev. Lett. 80 (1998) 3308–3311.
[126] C. Peskin, Numerical analysis of blood flow in the heart, J. Comput. Phys. 25 (1977) 220–252.
[127] C. Peskin, The immersed boundary method, Acta Numerica 11 (2002) 1–39.
[128] D. Pelletier, A. Fortin, and R. Camarero, Are FEM solutions of incompressible flows really incompressible (Or how simple flows can cause headaches), Int. J. Numer. Meth. Fluids 9 (1989) 99–112.
[129] W.H. Press, B.P. Flemming, S.A. Teukolsky and W.T. Vetterling, Numerical Recipes, The Art of Scientific Computing, 2nd edn., Cambridge University Press, Cambridge, 1989.
[130] M. Plapp and A. Karma, Multiscale finite-difference-diffusion-Monte-Carlo method for simulating dendritic solidification, J. Comput. Phys. 165 (2000) 592–619.
[131] C. Peskin and D. Mcqueen, A general method for the computer simulation of biological systems interacting with fluids, Symp. Soc. Exp. Biol. 49 (1995) 265–276.
[132] C. Peskin and B. Printz, Improved volume conservation in the computation of flows with immersed elastic boundaries, J. Comput. Phys. 105 (1993) 33–46.
[133] Rappaport, R., Int. Rev. Cytol. (31) (1971) 169.
[134] J.C. Ramirez, C. Beckermann, A. Karma, and H.-J. Diepers, Phase-field modeling of binary alloy solidification with coupled heat and solute diffusion, Phys. Rev. E 69 (2004) 051607.

[135] M. Raessi, M. Bussmann, and J. Mostaghimi, A semi-implicit finite volume implementation of the CSF method for treating surface tension in interfacial flows, Int. J. Numer. Meth. Fluids 59 (2009) 1093–1110.
[136] J. Rosam, P.K. Jimack, and A. Mullis, A fully implicit, fully adaptive time and space discretisation method for phase-field simulation of binary alloy solidification, J. Comput. Phys. 225 (2007) 1271–1287.
[137] V. Rutka and Z. Li, An explicit jump immersed interface method for two-phase Navier-Stokes equations with interfaces, Comput. Methods Appl. Mech. Engrg. 197 (2008) 2317-2328.
[138] C. Reisinger and G. Wittum, On multigrid for anisotropic equations and variational inequalities, Pricing multi-dimensional European and American options, Computing and Visualization in Science, (7)(2004) 189–197.
[139] J. Sur, A.L. Bertozzi, and R.P. Behringer, Reverse undercompressive shock structures in driven thin film flow. Physical Review Letters (90) (2003) 126105-1–126105-4.
[140] T.P. Schulze, Simulation of dendritic growth into an undercooled melt using kinetic Monte Carlo techniques, Phys. Rev. E 78 (2008) 020601(R).
[141] J.A. Sethian and J. Strain, Crystal growth and dendlritic solidification, J. Comput. Phys. 98 (1992) 231–253.
[142] C. Samson, L. Blanc-Feraud, G. Aubert, and J. Zerubia, A level set model for image classification, Inte. J. Comput. Vis., 40 (3) (2000) 187–197.
[143] A. Stuart and A.R. Humphries, Dynamical system and numerical analysis, Cambridge University Press, Cambridge, 1998.
[144] C.J. Shih, M.H. Lee, and C.W. Lan, A simple approach toward quantitative phase field simulation for dilute-alloy solidification, J. Cryst. Growth 282 (2005) 515–524.
[145] M. Sellier,Y.C. Lee, H.M. Thompson, and P.H. Gaskell, Thin film flow on surfaces containing arbitrary occlusions. Computers & Fluids, (38) (2009) 171–182.
[146] C.W. Shu and S. Osher, Efficient implementation of essentially non-oscillatory shock capturing schems II. Journal of Computational Physics (83)(1989) 32–78.
[147] M. Sellier and S. Panda, Beating capillarity in thin film flows. International Journal for Numerical Methods in Fluids, (63) (2010) 431–448.
[148] L.W. Schwartz and R.V. Roy, Theoretical and numerical results for spin coating of viscous liquids. Physics of Fluids (16) (2004) 569–584.
[149] J. Strain, A boundary integral approach to unstable solidification, J. Comput. Phys. 85 (1989) 342–389.
[150] S.M. Troian, E. Herbolzheimer, S.A. Safran, and J.F. Joanny, Fingering instability of driven spreading films. Europhysics Letters (10) (1989)25–30.
[151] G. Tryggvason, B. Bunner, A. Esmaeeli, D. Juric, N. Al-Rawahi, W. Tauber, J. Han, S. Nas, and Y.-J. Jan, A front-tracking method for the computations of multiphase flow, J. Comput. Phys. 169 (2001) 708-759.
[152] X. Tong, C. Beckermann, and A. Karma, Q. Li, Phase-field simulations of dendritic crystal growth in a forced flow, Phys. Rev. E 63 (2001) 061601.
[153] U. Trottenberg, C. Oosterlee, and A. Schüller, Multigrid, Academic Press, USA, 2001.
[154] L.A. Vese and T.F. Chan, A multiphase level set framevork for image segmentation using the mumford and shah model, Int. J. Comput. Vis. 50 (3) (2002) 271–293.
[155] S. Veremieiev, H.M. Thompson, Y.C, Lee, and P.H. Gaskell. Inertial thin film flow on planar surfaces featuring topography. Computers & Fluids (39)(2010) 431–450.
[156] J.A. Warren and W.J. Boettinger, Prediction of dendritic growth and microsegregation patterns in a binary alloy using the phase-field method, Acta Metall. Mater. 43 (1995) 689-703.
[157] T.P, Witelski and M. Bowen, ADI schemes for higher-order nonlinear diffusion equations. Applied Numerical Mathematics, 45 (2003)331–351.
[158] A.A. Wheeler, W.J. Boettinger, and G.B. McFadden, Phase-field model for isothermal phase transitions in binary alloys, Phys. Rev. A 45 (1992) 7424–7439.
[159] K. Wang, A. Chang, L.V. Kale, and J.A. Dantzig, Parallelization of a level set method for simulating dendritic growth, J. Parallel Distrib. Comput. 66 (2006) 1379–1386.
[160] A.A. Wheeler, W.J. Boettinger, and G.B. Mcfadden, Phase-field model for isothermal phase transitions in binary alloys, Phys. Rev. A 45 (1992) 7424–7439.
[161] S.-L. Wang and R.F. Sekerka, Algorithms for phase field computation of the dendritic operating state at large supercoolings, J. Comput. Phys. 127 (1996) 110–117.

[162] Y. Xu, J.M. McDonough, and K.A. Tagavi, A numerical procedure for solving 2D phase-field model problems, J. Comput. Phys. 218 (2006) 770-7-93.
[163] H. Yin and S.D. Felicelli, A cellular automaton model for dendrite growth in magnesium alloy AZ91, Modelling Simul. Mater. Sci. Eng. 17 (2009) 075011.
[164] X. Yang, J.J. Feng, C. Liu, and J. Shen, Numerical simulations of jet pinching-off and drop formation using an energetic variational phase-field method, J. Comput. Phys. 218 (2006) 417–428.
[165] A. Yezzi, S. Kichenassamy, A. Kumar, P Olver, and A. Tannenbaum, A geometric snake model for segmentation of medical imagery, IEEE Transaction on Medical Image, 16 (2)(1997) 199–209.
[166] Y.-l. Wang, J. D. Silverman, and L.-G. Cao, Single Particle Tracking of Surface Receptor Movement during Cell Division, The Journal of Cell Biology,1994.
[167] M.F. Zhu and C.P. Hong, A modified cellular automaton model for the simulation of dendritic growth in solidification of alloys, ISIJ Int. 41 (2001) 436–445.
[168] P. Zhao, J.C. Heinrich, and D.R. Poirier, Fixed mesh front-tracking methodology for finite element simulations, Int. J. Numer. Meth. Engng. 61 (2004) 928–948.
[169] M.F. Zhu, S.Y. Lee, and C.P. Hong, Modified cellular automaton model for the prediction of dendritic growth with melt convection, Phys. Rev. E 69 (2004) 061–610.
[170] P. Zhao, J.C. Heinrich, and D.R. Poirier, Dendritic solidification of binary alloys with free and forced convection, 49 (2005) 233–266.
[171] Z.C. Zheng and N. Zhang, Frequency effects on lift and drag for flow past an oscillating cylinder, J. Fluids Struct. 24 (2008) 382–399.