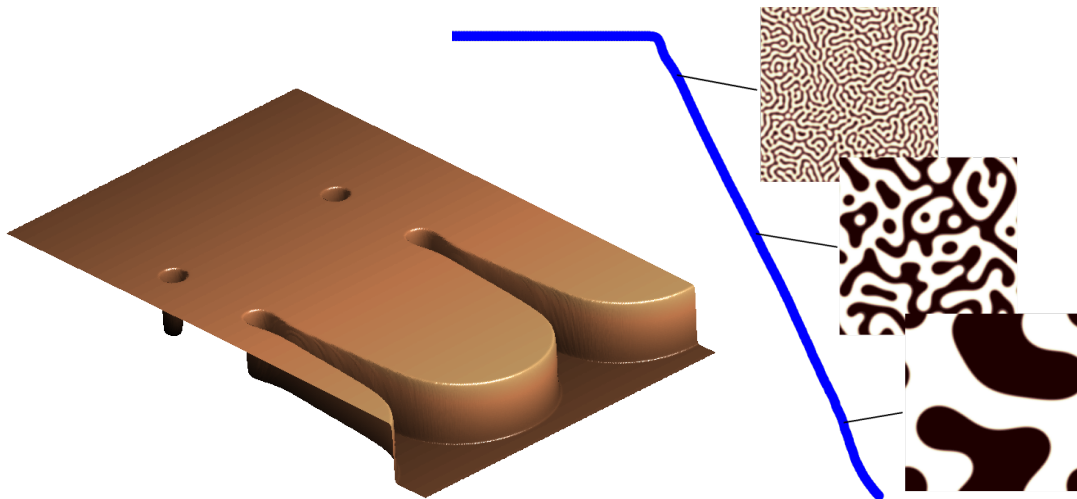


Cahn–Hilliard-type Equations: Robust Discretization and Efficient Implementation



Dissertation

zur Erlangung des Doktorgrades (Dr. rer. nat.)
der Mathematisch–Naturwissenschaftlichen Fakultät
der Rheinischen Friedrich-Wilhelms-Universität Bonn

vorgelegt von
Patrick Penzler
aus Bonn

Bonn, Januar 2009

Angefertigt mit Genehmigung der Mathematisch–Naturwissenschaftlichen Fakultät
der Rheinischen Friedrich-Wilhelms-Universität Bonn

Diese Dissertation ist auf dem Hochschulschriftenserver der ULB Bonn unter
http://hss.ulb.uni-bonn.de/diss_online elektronisch publiziert.

Erscheinungsjahr: 2009

1. Gutachter: Prof. Dr. Felix Otto
 2. Gutachter: Prof. Dr. Martin Rumpf
- Tag der Promotion: 9. April 2009

Zusammenfassung

In der vorliegenden Arbeit stellen wir eine robuste und effiziente Methode vor, um Cahn–Hilliard-artige Gleichungen numerisch zu lösen. Wir betrachten Gleichungen der Gestalt

$$\partial_t \phi + \nabla \cdot J = f \quad \text{mit} \quad J = -M(\phi) \nabla w \quad \text{und} \quad w = -\varepsilon \Delta \phi + \varepsilon^{-1} G'(\phi),$$

also Gleichungen vierter Ordnung mit zwei Nichtlinearitäten: der Mobilität M und dem Potential G . Über die Wahl dieser beiden Funktionen lässt sich das Verhalten der Lösungen beeinflussen und ein großer Bereich von physikalischen Phänomenen abbilden.

Die Gleichung lässt sich formal als Gradientenfluss einer freien Energie schreiben. Wir lassen uns bei der Zeitdiskretisierung von dieser Struktur leiten. Die resultierenden semi-diskreten Gleichungen für den Fluss J sind von der Form $(\text{id} + \varepsilon(\nabla \nabla \cdot)^2 - \varepsilon^{-1} \nabla G''(\phi) \nabla \cdot) J = r$. Durch Einführen einer Hilfsvariablen $K := -\nabla \nabla \cdot J$ können wir die Gleichung in zwei Gleichungen aufteilen, in deren schwacher Version dann nur noch Divergenzen von J und K auftreten. Für die Ortsdiskretisierung wählen wir daher die $H(\nabla \cdot)$ -konformen Raviart–Thomas (RT) Finiten Elemente. Diese sind besonders geeignet für Probleme mit einer Kontinuitätsgleichung, da die Divergenz dreiecksweise erhalten bleibt. Für ϕ und w sind stückweise konstante Elemente die passende Wahl.

Beim Lösen der voll-diskreten Gleichungen tritt die Inverse der RT-Massenmatrix auf. Da es für allgemeine Gitter keine mass-lumping Technik für RT-Elemente gibt, ist die Inverse voll besetzt. Um dieses Problem zu umgehen, schlagen wir vier Möglichkeiten vor und vergleichen diese. Als beste Wahl stellt sich eine Technik heraus, die aus den Gemischten Methoden bekannt ist. Diese Technik konnten wir an unsere Gleichung anpassen. In typischen Simulationen erhalten wir damit eine um einen Faktor 50 schnellere Laufzeit gegenüber einfachen matrixfreien Methoden.

Wir testen unsere Diskretisierung an zwei Anwendungen. Zum einen an Vergrößerungsprozessen und zum anderen an epitaktischem Wachstum. Vergrößerungsprozesse sind eine Herausforderung für die Numerik, da große Systeme und lange Zeiten simuliert werden müssen. Gleichzeitig entwickelt sich die Lösung anfangs auf sehr kurzen Zeitskalen, so dass Zeitadaptivität unerlässlich ist. In beiden Anwendungen müssen Bereiche räumlich aufgelöst werden, deren Längenskala etwa ein Hundertstel der Systemlänge beträgt. Somit muss auch ortsadaptiv gearbeitet werden. Wir vergleichen theoretische Resultate zur Vergrößerungsrate mit den Ergebnissen aus unseren Simulationen und finden gute Übereinstimmung. Im epitaktischen Wachstum dient die Gleichung als diffuse-interface approximation (DIA) an ein freies Randwertproblem. Nachdem wir uns überzeugt haben, dass dies eine gute Approximation ist, vergleichen wir unsere Simulation mit einer Front-Tracking Simulation. Solange beide Modelle gültig sind, sehen wir eine gute Übereinstimmung. Nur die DIA kann über topologische Änderungen hinaus weiter laufen.

Um die Diskretisierung zu implementieren, wurde eine Finite Elemente Toolbox von Grund auf neu entwickelt. Ein Hauptgrund für die Neuentwicklung war, dass für viele Simulationen im Kristallwachstum, insbesondere in dem hier betrachteten Stufenfluss-Regime, eine Unterstützung für so genannte schief-periodische Gitter nötig ist.

Contents

1	Overview	1
2	Cahn–Hilliard-type equations	5
2.1	Dissipative systems	7
2.2	Some spaces of negative order	9
2.3	Gradient flow of a dissipative system	11
2.4	Four examples	12
2.5	Existence of solutions	14
2.6	Morphology of solutions	18
3	Time Discretization	25
3.1	Derivation of the time-discrete equations	26
3.2	Properties of the time-discrete operator	29
3.3	Time-step schemes	34
3.4	Adaptivity	35
4	Spatial Discretization	37
4.1	Idea behind the discretization	38
4.2	Piecewise constant finite elements	41
4.3	Raviart–Thomas finite elements	42
4.4	A discrete gradient	45
4.5	The fully discrete equation	47
4.6	Error estimates	49
4.7	Matrix representation	52
4.8	Solving the fully discrete equation	54
4.9	Adaptivity	72
4.10	Linear solver	81
4.11	Comparison of the methods	82
5	Coarsening	89
5.1	A model for phase segregation	90
5.2	Simulating the equations	93

5.3	Results	97
6	Epitaxial Growth	109
6.1	Molecular beam epitaxy	110
6.2	Instabilities in epitaxial growth	111
6.3	Step-flow model	112
6.4	Diffuse-interface approximation	115
6.5	Numerical tests	118
6.6	Nonlinear instability of a step train	126
7	The finite element software	129
7.1	The math library	130
7.2	The mesh library	131
7.3	The DOF manager	135
7.4	CHEST	136
A	Complements	141
A.1	Gradient-flow structure of the sharp-interface limits	141
A.2	Energy minimization during refinement	142
A.3	Determining the step position	146
A.4	Time adaptivity	150
A.5	Solving Poisson's equation	154
A.6	Computing the negative norm	157
A.7	Eigenvalues of the discrete operator	158
A.8	Number of required Newton steps	158
A.9	Constructing the piecewise constant gradient	159
A.10	Quadrature	160
A.11	Hardware, software versions and compiler options	162
B	Time-step schemes	163
B.1	Euler backward	163
B.2	Crank–Nicolson	163
B.3	Peaceman–Rachford-type	164
B.4	BGP	165
B.5	TR-BDF ₂	166
	Symbols	169
	Bibliography	171

Overview

In this work, we present a robust and efficient numerical method to simulate Cahn–Hilliard-type equations. The discretization is lead by the gradient-flow structure of the equation and the resulting finite-element method uses both time- and space-adaptivity. Our goal was to find a fully practical method that is capable of simulating large systems and long times and is supported by a certain amount of numerical analysis.

We consider nonlinear fourth-order equations of the type

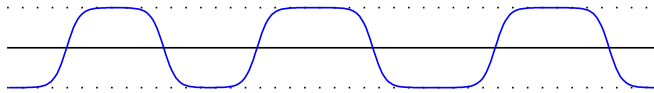
$$\partial_t \phi - \nabla \cdot (M(\phi) \nabla w) = f, \quad (1.1a)$$

$$w = -\varepsilon \Delta \phi + \varepsilon^{-1} G'(\phi). \quad (1.1b)$$

There are two nonlinearities in the equation, namely the mobility $M(\phi)$ and the potential $G(\phi)$. We will see in Section 2.3 that Equation (1.1) is a gradient flow to the free energy

$$E_\varepsilon(\phi) = \int_\Omega \frac{\varepsilon}{2} |\nabla \phi|^2 + \varepsilon^{-1} G(\phi). \quad (1.2)$$

To attain a small energy, ϕ has to take values in the minima of G with few transitions of width $\mathcal{O}(\varepsilon)$ in-between.



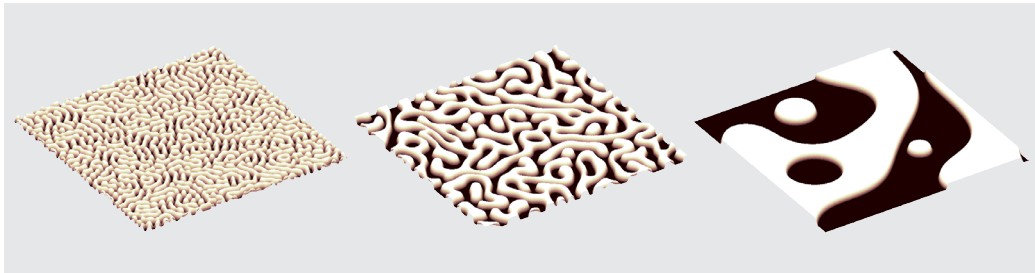
Therefore the long-time behaviour of the solution is influenced by the minima of the chosen potential G .

To understand the role of M , note that the variational derivative of the free energy (1.2) is w , often called the chemical potential. By Fick's law, the mass flux J is given by

$$J = -M(\phi) \nabla w, \quad \text{where} \quad w = \frac{\delta E_\varepsilon}{\delta \phi}(\phi)$$

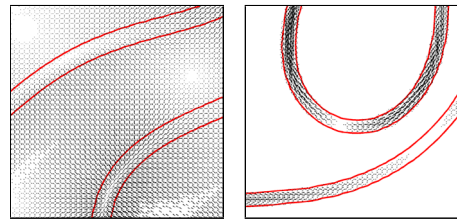
and $M(\phi)$ is a diffusion coefficient. Thus, the choice of $M(\phi)$ influences the flux J and with that the morphology of the solution.

With the choice of mobility and potential, a broad range of phenomena can be attained. In this work, we focus on two applications.

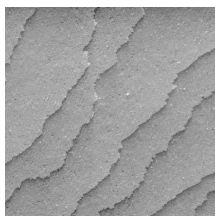


The first application are coarsening processes in binary mixtures, see Chapter 5. Here ϕ denotes the volume fraction of one of the two components. The potential in the energy can be interpreted as the amount of free energy a homogeneous solution would have and the Dirichlet term can be interpreted as surface tension. By using a double-well potential with wells at ± 1 , the mixture will decompose into the two phases. Looking at (1.2), the energy will be lower if there are less transitions between the two phases, so the system will coarsen in time.

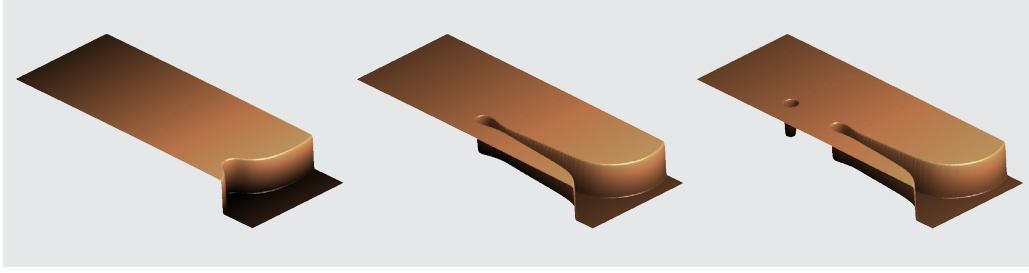
The rate at which the system coarsens depends on the choice of the mobility. We consider both the classical case of constant mobility and the case of a degenerate mobility, where the flux is suppressed in the pure phases and is thus present only along the interfaces.



We chose this application because on the one hand, there are analytical results on the coarsening rate and on the other hand it is a challenging application since it requires large systems and long times to verify the results. At the same time, the system may change on short time-scales, so that time-adaptivity is essential. Furthermore, the transition regions have to be resolved by the mesh and their length-scale makes up only a hundredth of the system size, so that spatial adaptivity is also necessary.



As a second application we consider crystal growth, more precisely epitaxial growth of thin crystalline films. Here, ϕ denotes the height of the film. Since the film can only have heights of a multiple of one atomic height, G is a multi-well potential with wells at the integers. During step-flow growth, the film consists of large terraces with atom-high steps in-between. The mobility is used to create an asymmetry in the attachment rates at the steps, see Chapter 6 for a detailed introduction. The asymmetry induces instabilities, such that an initially straight step may develop overhangs during growth and eventually a vacancy island pinches off.



To simulate this nonlinear instability with reasonable computational costs, our software supports so-called skew-periodic meshes, see Section 7.2.2.

Using our simulation tool, we find good agreement between theory and simulation in both applications. However, our discretization is not limited to these applications. For example, the thin-film equation has also been successfully treated with our method.

To discretize equation (1.1) in time, we use the gradient-flow structure and get a symmetric fourth-order equation for the flux J of the form

$$\left(\frac{1}{\tau} \text{id} + \varepsilon (\nabla \nabla \cdot)^2 - \varepsilon^{-1} \nabla G''(\phi) \nabla \cdot \right) J = r \quad (1.3)$$

in each time step. The value of ϕ is computed using the continuity equation.

For the spatial discretization, we introduce an auxiliary quantity $K := -\nabla \nabla \cdot J$. Then (1.3) becomes

$$\frac{1}{\tau} J - \varepsilon \nabla \nabla \cdot K - \varepsilon^{-1} \nabla G''(\phi) \nabla \cdot J = r. \quad (1.4)$$

In the weak formulation of the equations for J and K , it is now enough for the vector fields to be in $H(\nabla \cdot, \Omega)$. We therefore use the $H(\nabla \cdot)$ -conformal Raviart–Thomas finite elements. These are especially suited for systems which include the continuity equation, since the divergence is retained triangle-wise. Appropriate elements for the functions ϕ and w are piecewise constant elements. When inserting the discrete K back into (1.4), the inverse of the mass matrix appears. Since there are no mass-lumping procedures on general meshes for Raviart–Thomas elements, the inverse would be dense. To bypass this complication, we propose several solutions. They range from simple matrix-free methods to a method known from the area of mixed methods, where inter-element multipliers are introduced to get a block-diagonal mass matrix. We adapted the latter successfully to our situation. The difference in the running times between the different approaches can be a factor of over fifty in typical simulations.

The work is organized as follows: In Chapter 2, we present the type of equation in some more detail and show that it has a natural gradient-flow structure. We review some existence results and explain the typical behaviour of the solutions at the hand of some examples.

In Chapter 3, we discretize the equation in time based on the gradient-flow structure. As a result, we get an equation similar to (1.3) in each time-step. To handle the nonlinearity, we use Newton's method. A simple calculation demonstrates that the time-discrete operator is positive definite if $\tau \lesssim \varepsilon^3$. We show that as soon as the system enters the so-called interfacial

regime, the operator is positive-definite for $\tau \lesssim 1$, independent of ε . Therefore, large time steps are possible. To discretize Cahn–Hilliard-type equations, it is not enough for the time-step scheme to be A-stable. After reviewing several second-order schemes, we choose the less common TR-BDF2 method. Besides providing the required stability for long time steps, it contains an embedded third-order scheme which can be used for time-adaptivity.

The spatial discretization outlined above is presented in Chapter 4: after explaining the idea, we follow another approach which yields the same result but allows to prove existence of the fully discrete solution and error estimates. Then, in Section 4.8, we present four methods to handle the mentioned difficulty with the dense inverse. The methods are compared in Section 4.11. To adapt the mesh, we do not use a general a-posteriori error estimator. Instead, since the solutions have a very special structure, we employ a method tailored to our situation.

In Chapters 5 and 6, the results for the two applications, coarsening processes and epitaxial growth, are presented.

We developed a finite-element toolbox to simulate the equations in two dimensions. Since available toolboxes often do not include the Raviart–Thomas elements and, more importantly, provide no support for skew-periodic meshes, we developed a new software from scratch. It was written in C++ and is not just a reference implementation, but was optimized to be able to handle large system sizes and long simulation times. A brief overview can be found in Section 7.

Cahn–Hilliard-type equations

This chapter gives an overview of the type of equations considered in this work: Let Ω be a domain in \mathbb{R}^d and $f : \mathbb{R} \times \mathbb{R}^d \rightarrow \mathbb{R}$ a given function. We consider equations of the type

$$\partial_t \phi(t, x) - \nabla \cdot \left[M(\phi(t, x)) \nabla \frac{\delta E}{\delta \phi}(\phi(t, x)) \right] = f(t, x) \quad (2.1)$$

with a mobility function $M : \mathbb{R} \rightarrow \mathbb{R}_0^+$ and an energy functional E . The notion of the variational derivative $\delta E / \delta \phi$ is explained in Definition 2.2 on the following page.

If $f(t, x) = 0$, we can identify this equation as the gradient flow of a dissipative system, see Sections 2.1 and 2.3. This will help us as a guideline for the discretization later. In Section 2.4, we present four examples for energy and mobility which will appear in the application chapters 5 and 6. In Section 2.5, we give a short overview of existence results relevant for these examples and in Section 2.6 we investigate the typical behaviour of solutions.

In the introductory sections, we will not make any regularity assumptions since we leave the mobility and the energy unspecified at first. Furthermore, we will drop the arguments (t, x) and introduce the *flux* $J : \mathbb{R} \times \mathbb{R}^d \rightarrow \mathbb{R}^d$, so that (2.1) becomes

$$\partial_t \phi + \nabla \cdot J = f, \quad (2.2a)$$

$$\frac{1}{M(\phi)} J = -\nabla \frac{\delta E}{\delta \phi}(\phi). \quad (2.2b)$$

As boundary conditions we use

(BC1) periodic boundary conditions (then $\Omega = [0, L_x] \times [0, L_y]$) or

(BC2) no-flux and natural boundary conditions, i.e. $J \cdot \nu = 0$ and $\nabla \phi \cdot \nu = 0$.

We can easily see two important properties of the equation.

Lemma 2.1. *If $f = 0$, then Cahn–Hilliard-type equations are mass conserving and the energy is a Lyapunov-functional.*

Proof. Both properties follow directly from the definition of the equation. Mass conservation is a consequence of the chosen boundary conditions:

$$\partial_t \int_{\Omega} \phi = \int_{\Omega} \partial_t \phi = - \int_{\Omega} \nabla \cdot J = - \int_{\partial\Omega} J \cdot \nu = 0.$$

A Lyapunov-functional is a non-increasing functional. To see that the energy is non-increasing, write

$$\partial_t E(\phi) = \text{diff } E(\phi) \partial_t \phi = - \int_{\Omega} \frac{\delta E}{\delta \phi}(\phi) (\nabla \cdot J) = \int_{\Omega} \nabla \frac{\delta E}{\delta \phi}(\phi) \cdot J = - \int_{\Omega} \frac{1}{M(\phi)} |J|^2 \leq 0.$$

□

A weak form of (2.2) for either boundary condition is

$$\frac{d}{dt} \int_{\Omega} \phi \zeta + \int_{\Omega} (\nabla \cdot J) \zeta = \int_{\Omega} f \zeta \quad \forall \zeta, \quad (2.3a)$$

$$\int_{\Omega} \frac{1}{M(\phi)} J \cdot \tilde{J} - \int_{\Omega} \frac{\delta E}{\delta \phi}(\phi) (\nabla \cdot \tilde{J}) = 0 \quad \forall \tilde{J}. \quad (2.3b)$$

We have used the definition

Definition 2.2 (Variational derivative). The *variational derivative* or L^2 -*gradient* of E at ϕ is defined through

$$\int_{\Omega} \frac{\delta E}{\delta \phi}(\phi) v = \text{diff } E(\phi) v$$

for all $v \in L^2(\Omega)$ and the *differential* of E at ϕ in turn is defined as

$$\text{diff } E(\phi) v := \left. \frac{d}{d\delta} \right|_{\delta=0} E(\phi + \delta v).$$

◇

Later, we also need

Definition 2.3 (Second variational derivative). The *second variational derivative* of E at ϕ is defined through

$$\int_{\Omega} \frac{\delta^2 E}{\delta \phi^2}(\phi) u v = \text{Hess } E(\phi)(u, v)$$

for all $u, v \in L^2(\Omega)$. The Hessian of E is given by

$$\text{Hess } E(\phi)(u, v) := \left. \frac{d}{d\delta} \frac{d}{d\epsilon} \right|_{\epsilon, \delta=0} E(\phi + \epsilon u + \delta v).$$

◇

2.1 Dissipative systems

We now want to embed the above equation in a more general framework, namely gradient flows of dissipative systems. This will help us as a guideline for the discretization later.

Let $J : \Omega \rightarrow \mathbb{R}^n$ be a vector field fulfilling one of the two boundary conditions and $D(J, J)$ be bilinear in J . Let $H(J)$ be linear in J . Then define the *Rayleigh functional* by

$$\mathcal{R}[J] := \frac{1}{2}D(J, J) + H(J). \quad (2.4)$$

Given a mass conserving system $\partial_t \phi + \nabla \cdot J = 0$ and an energy $E(\phi)$, define

$$H(J) := \int_{\Omega} \nabla \frac{\delta E}{\delta \phi}(\phi) \cdot J = -\text{diff } E(\phi)(\nabla \cdot J) = \text{diff } E(\phi) \partial_t \phi = \frac{d}{dt} E(\phi) \quad (2.5)$$

and think of D as the dissipation of energy generated by friction. The *Rayleigh principle* says

“At any time, the flux J minimizes the Rayleigh functional”.

The Euler–Lagrange equations of the Rayleigh functional \mathcal{R} are

$$D(J_*, \tilde{J}) + H(\tilde{J}) = 0 \iff D(J_*, \tilde{J}) + \int_{\Omega} \nabla \frac{\delta E}{\delta \phi}(\phi) \cdot \tilde{J} = 0 \quad \forall \tilde{J}. \quad (2.6)$$

Taking $\tilde{J} = J_*$, we get with (2.5)

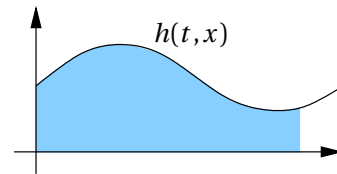
$$\frac{d}{dt} E(\phi) + D(J_*, J_*) = 0,$$

i.e. the minimizing flux balances the change of energy and the dissipation subject to mass conservation.

Example: thin-film equation

As a prototype of a Cahn–Hilliard-type equation, we take a short look at the thin-film equation. For a more detailed presentation, see e.g. Oron et al. [1997].

Starting point are the equations for a viscous, incompressible fluid: the Navier–Stokes equations. Let $h(t, x)$ denote the height of the fluid and denote by H and L the typical height and length of the fluid. Assuming that $H \ll L$, the horizontal component of the fluid’s velocity field is averaged in vertical direction. This averaged quantity is then denoted by u .



The dissipation of a viscous fluid is proportional to the change of the kinetic energy of the fluid. In the lubrication approximation considered here, the dissipation in non-dimensional form is given by

$$\int_{\Omega} h^{2-q} |u|^2, \quad q \in [0, 3].$$

Mass conservation means

$$\partial_t h + \nabla \cdot (hu) = 0, \quad (2.7)$$

so we define $J_{\text{TF}} := hu$. In terms of J_{TF} , the dissipation is given by

$$D_{\text{TF}}(J_{\text{TF}}, J_{\text{TF}}) = \int_{\Omega} \frac{1}{h^q} |J_{\text{TF}}|^2 = \int_{\Omega} \frac{1}{M_{\text{TF}}(h)} |J_{\text{TF}}|^2$$

with $M_{\text{TF}}(h) := h^q$. The energy is determined by the surface tension. Since $H \ll L$, we can use a small-slope approximation:

$$E_{\text{TF}}(h) := \int_{\Omega} \frac{1}{2} |\nabla h|^2.$$

Applying Rayleigh's principle (2.6) yields

$$\frac{1}{M_{\text{TF}}(h)} J_{\text{TF}} = -\nabla \frac{\delta E}{\delta h}(h),$$

which is, together with the continuity equation (2.7), the thin-film equation. Note that it has the form anticipated in (2.2).

Inserting J_{TF} into (2.7) gives the usual form of the thin-film equation:

$$\partial_t h + \nabla \cdot (h^q \nabla \Delta h) = 0.$$

Cahn–Hilliard-type equations as dissipative systems

Motivated by the example above, we use the dissipation

$$D(J, \tilde{J}) := \int_{\Omega} \frac{1}{M(\phi)} J \cdot \tilde{J} \quad (2.8)$$

for all Cahn–Hilliard-type equations. In the same way as in the example, we then get from (2.6)

$$\int_{\Omega} \left(\frac{1}{M(\phi)} J + \nabla \frac{\delta E}{\delta \phi}(\phi) \right) \cdot \tilde{J} = 0 \quad \forall \tilde{J},$$

which yields together with mass conservation the system

$$\begin{aligned} \partial_t \phi + \nabla \cdot J &= 0 \\ \frac{1}{M(\phi)} J &= -\nabla \frac{\delta E}{\delta \phi}(\phi). \end{aligned}$$

This is (2.2) with $f = 0$.

2.2 Some spaces of negative order

Before we can interpret dissipative systems as a gradient flow, we need to define some Hilbert spaces. In this section, we assume Ω to be open and bounded and denote by $\langle \cdot, \cdot \rangle$ the dual pairing between the dual of $H^1(\Omega)$, denoted $H^1(\Omega)'$, and $H^1(\Omega)$ itself. The following theorem can be found e.g. in Adams [2003].

Theorem 2.4 (The Dual of $H^1(\Omega)$). *For every $f \in H^1(\Omega)'$ there exist elements $f_0 \in L^2(\Omega)$, $f_1 \in L^2(\Omega)^2$ such that for all $u \in H^1(\Omega)$ we have*

$$\langle f, u \rangle = \int_{\Omega} u f_0 + \int_{\Omega} \nabla u \cdot f_1. \quad (2.9)$$

Moreover

$$\|f\|_{H^1(\Omega)'}^2 = \min_{(f_0, f_1) \in \mathcal{A}} \left(\|f_0\|_{L^2(\Omega)}^2 + \|f_1\|_{L^2(\Omega)}^2 \right), \quad (2.10)$$

where

$$\mathcal{A} = \left\{ (f_0, f_1) \in L^2(\Omega) \times L^2(\Omega)^2 \mid (2.9) \text{ holds for every } u \in H^1(\Omega) \right\}.$$

The element satisfying (2.9) and (2.10) is unique.

We will define a norm that is easier to handle. First, we define a scalar product on $H^1(\Omega)'$.

Definition 2.5 (H^{-1} scalar product). Given $g_1, g_2 \in H^1(\Omega)'$ with mean zero (in the sense that $\langle g_i, 1 \rangle = 0$), let $w_1, w_2 \in H^1(\Omega)$ be solutions of

$$\begin{aligned} -\Delta w_i &= g_i & \text{in } \Omega, \\ \frac{\partial w_i}{\partial \nu} &= 0 & \text{on } \partial\Omega \end{aligned} \quad (2.11)$$

for $i = 1, 2$. Then define

$$\langle g_1, g_2 \rangle_{H^{-1}(\Omega)} := \int_{\Omega} \nabla w_1 \cdot \nabla w_2.$$

◇

Definition 2.6 (H^{-1} norm). Given $f \in H^1(\Omega)'$ with mean zero, we define

$$\|f\|_{H^{-1}(\Omega)}^2 := \langle f, f \rangle_{H^{-1}(\Omega)}.$$

◇

We now show that this norm is equivalent to the one defined in (2.10):

Lemma 2.7. *The norms $\|\cdot\|_{H^1(\Omega)'}$ and $\|\cdot\|_{H^{-1}(\Omega)}$ are equivalent.*

Proof. Let $w \in H^1(\Omega)$ be a solution of

$$-\Delta w = f \quad \text{in } \Omega \quad (2.12)$$

with Neumann boundary conditions.

$$\textcircled{1} \quad \|\cdot\|_{H^1(\Omega)'} \leq C \|\cdot\|_{H^{-1}(\Omega)}$$

Set $f_0 = w$, $f_1 = \nabla w$. Then by Poincaré's inequality

$$\|f\|_{H^1(\Omega)'}^2 \leq \|w\|_{L^2(\Omega)}^2 + \|\nabla w\|_{L^2(\Omega)}^2 \leq C \|\nabla w\|_{L^2(\Omega)}^2$$

$$\textcircled{2} \quad \|\cdot\|_{H^{-1}(\Omega)} \leq C \|\cdot\|_{H^1(\Omega)'}$$

The weak form of equation (2.12) is

$$\int_{\Omega} \nabla w \cdot \nabla v = \langle f, v \rangle = \int_{\Omega} f_0 v + \int_{\Omega} f_1 \cdot \nabla v \quad \forall v \in H^1(\Omega).$$

Therefore, setting $v = w$, we get with a generic constant C

$$\begin{aligned} \|\nabla w\|_{L^2(\Omega)}^2 &= \int_{\Omega} f_0 w + \int_{\Omega} f_1 \cdot \nabla w \\ &\leq \|f_0\|_{L^2(\Omega)} \|w\|_{L^2(\Omega)} + \|f_1\|_{L^2(\Omega)} \|\nabla w\|_{L^2(\Omega)} \\ &\leq C \|f_0\|_{L^2(\Omega)} \|\nabla w\|_{L^2(\Omega)} + \|f_1\|_{L^2(\Omega)} \|\nabla w\|_{L^2(\Omega)} \\ &\leq C \|\nabla w\|_{L^2(\Omega)} (\|f_0\|_{L^2(\Omega)} + \|f_1\|_{L^2(\Omega)}) \end{aligned}$$

using Hölder and Poincaré. Dividing by $\|\nabla w\|_{L^2(\Omega)}$ yields

$$\|\nabla w\|_{L^2(\Omega)}^2 \leq C (\|f_0\|_{L^2(\Omega)} + \|f_1\|_{L^2(\Omega)})^2 \leq C (\|f_0\|_{L^2(\Omega)}^2 + \|f_1\|_{L^2(\Omega)}^2) = C \|f\|_{H^1(\Omega)'}^2.$$

□

If we use the equation

$$-\nabla \cdot (M(x) \nabla w) = g \quad \text{instead of} \quad -\Delta w = g$$

in (2.11), then there still exists a solution to (2.11) as long as the operator is elliptic, i.e. as long as $M(x) \geq \theta > 0$. Therefore, we can define *weighted* H^{-1} scalar products and the corresponding norms:

Definition 2.8 (weighted H^{-1} scalar product). Given $g_1, g_2 \in H^1(\Omega)'$ with mean zero and a function M with $M(x) \geq \theta > 0$, let $w_1, w_2 \in H^1(\Omega)$ be solutions of

$$-\nabla \cdot (M \nabla w_i) = g_i \quad \text{in } \Omega,$$

for $i = 1, 2$, with Neumann boundary conditions. Then define

$$\langle g_1, g_2 \rangle_{H_M^{-1}(\Omega)} := \int_{\Omega} M \nabla w_1 \cdot \nabla w_2.$$

◇

Definition 2.9 (weighted H^{-1} -norm). Given $f \in H^1(\Omega)'$ with mean zero and a function M with $M(x) \geq \theta > 0$, define

$$\|f\|_{H_M^{-1}(\Omega)}^2 := \langle f, f \rangle_{H_M^{-1}(\Omega)}.$$

◇

2.3 Gradient flow of a dissipative system

Dissipative mass-conserving systems have a natural gradient-flow structure which we outline here. Note that this is only a formal analogy.

The general form of a gradient flow on a manifold \mathcal{M} is

$$\partial_t \phi = -\nabla_g E(\phi), \quad (2.13)$$

where the gradient ∇_g is defined by the metric g on \mathcal{M} through

$$g_\phi(\nabla_g E, v) = \text{diff } E(\phi)v \quad \forall v \in T_\phi \mathcal{M}. \quad (2.14)$$

Here, $T_\phi \mathcal{M}$ is the tangent space in the point ϕ . Inserting (2.13) into (2.14) yields

$$g_\phi(\partial_t \phi, v) + \text{diff } E(\phi)v = 0 \quad \forall v \in T_\phi \mathcal{M}. \quad (2.15)$$

To render the gradient flow meaningful, we have to choose a manifold and a metric on it. Since the system is mass conserving, it is natural to consider the manifold \mathcal{M} of all functions with a constant mass c . The tangent space is then the space of all functions which do not change the mass, i.e. with mean zero.

$$\mathcal{M} = \left\{ \phi \mid \int_\Omega \phi = c \right\}, \quad T_\phi \mathcal{M} = \left\{ v \mid \int_\Omega v = 0 \right\}.$$

As metric, we use the weighted H^{-1} scalar product introduced in the previous section. A motivation for this choice can be found at the end of this section. Then, the gradient flow is

$$\langle \partial_t \phi, v \rangle_{H_M^{-1}} + \text{diff } E(\phi)v = 0 \quad \forall v \in T_\phi \mathcal{M}.$$

By definition, this is

$$\int_\Omega \nabla w \cdot M(\phi) \nabla \tilde{w} + \int_\Omega \frac{\delta E}{\delta \phi}(\phi)v = 0,$$

where w and \tilde{w} are the solutions of

$$\begin{aligned} -\nabla \cdot (M(\phi) \nabla w) &= \partial_t \phi & \text{in } \Omega & & -\nabla \cdot (M(\phi) \nabla \tilde{w}) &= v & \text{in } \Omega \\ \nabla w \cdot \nu &= 0 & \text{on } \partial\Omega, & & \nabla \tilde{w} \cdot \nu &= 0 & \text{on } \partial\Omega. \end{aligned}$$

Now defining

$$J := M(\phi) \nabla w \quad \text{and} \quad \tilde{J} := M(\phi) \nabla \tilde{w},$$

we get

$$\partial_t \phi + \nabla \cdot J = 0 \quad \text{and} \quad v = -\nabla \cdot \tilde{J},$$

therefore obtaining Equation (2.2a). The gradient flow becomes

$$\int_\Omega \frac{1}{M(\phi)} J \cdot \tilde{J} - \int_\Omega \frac{\delta E}{\delta \phi}(\phi) (\nabla \cdot \tilde{J}) = 0, \quad (2.16)$$

which is Equation (2.3b).

Remark. In applications, the variational derivative of the energy is often called *chemical potential*, compare Rowlinson [1979]. Using the relation $J = M(\phi)\nabla w$ to replace J in (2.16) and integrating by parts shows

$$w = \frac{\delta E}{\delta \phi}(\phi). \quad (2.17)$$

◇

As a motivation for the choice of the metric, consider the following: Given a change $\nu \in T_\phi \mathcal{M}$ of $\phi \in \mathcal{M}$, we look for the flow J_* realizing ν while dissipating the least energy. The length of ν is then defined as the amount of energy dissipated by J_* :

$$\|\nu\|^2 := \min_{\tilde{J} \in Z_\nu} D(\tilde{J}, \tilde{J}) \stackrel{(2.8)}{=} \min_{\tilde{J} \in Z_\nu} \int_\Omega \frac{1}{M(\phi)} |\tilde{J}|^2 \quad \text{with} \quad Z_\nu = \{\tilde{J} \mid \nu + \nabla \cdot \tilde{J} = 0\}.$$

The optimality conditions are

$$\frac{1}{M(\phi)} J = \nabla w \quad \text{and} \quad \nu + \nabla \cdot J = 0.$$

Thus, we get for the length of ν

$$\|\nu\|^2 = \int_\Omega \nabla w \cdot M(\phi) \nabla w \quad \text{with} \quad -\nabla \cdot (M(\phi) \nabla w) = \nu$$

and can identify the length as

$$\|\nu\|^2 = \|\nu\|_{H_M^{-1}}^2.$$

2.4 Four examples

We will concentrate in the applications on energies of the form

$$E(\phi) = \int_\Omega \frac{1}{2} |\nabla \phi|^2 + G(\phi), \quad (2.18)$$

where $G(\phi) \geq 0$ is a potential. We call this type of energy a *Ginzburg–Landau-type* free energy. Such an energy was already used by Cahn and Hilliard [1958], where it describes the interfacial energy of an isotropic system of nonuniform composition.

The time discretization carried out in Chapter 3 can be done without knowing the precise form of the energy, but the spatial discretization in Chapter 4 will assume a Ginzburg–Landau-type energy, which leaves us with the choice of a mobility $M(\phi)$ and a potential $G(\phi)$. In Chapters 5 and 6 we consider various examples, which we shortly present here. See the respective chapters for details.

Chapter 5 considers a temperature-dependent model for the evolution of a binary mixture of an alloy. Denote the concentration of the first component by $c_1 = (\phi + 1)/2$. Since only $c_1 \in [0, 1]$ make sense, only values $\phi \in [-1, 1]$ have a physical meaning. This model is described by

- the *variable quench* (VQ) equation. It uses the potential $G_{VQ} : [-1, 1] \rightarrow \mathbb{R}$,

$$G_{VQ}(\phi) = \frac{5}{2} (\beta(1 - \phi^2) + (1 + \phi)\log(1 + \phi) + (1 - \phi)\log(1 - \phi)),$$

where β corresponds to the inverse of the temperature. G_{VQ} is continuous and bounded in $[-1, 1]$, but the derivative G'_{VQ} diverges at ± 1 . The mobility $M_{VQ} : [-1, 1] \rightarrow \mathbb{R}_0^+$ is

$$M_{VQ}(\phi) = 1 - \phi^2,$$

i.e. it degenerates at ± 1 . Due to this degeneracy, u will stay in $[-1, 1]$, see Elliott and Garcke [1996].

For $0 < \beta \leq 1$, i.e. at high temperatures, the potential is convex with a minimum at zero. Below the critical temperature $\beta_{\text{crit}} = 1$, i.e. for $\beta > 1$, the potential is a double-well potential. Letting go the temperature to zero ($\beta \gg 1$) or to the critical value ($0 < \beta - 1 \ll 1$) yields, respectively, the

- *deep quench* (DQ) equation with the same degenerate mobility

$$M_{DQ}(\phi) = 1 - \phi^2,$$

but with a concave potential

$$G_{DQ}(\phi) = \frac{15}{2}(1 - \phi^2).$$

Again, since the mobility degenerates at ± 1 , the values of ϕ stay in $[-1, 1]$.

- *shallow quench* (SQ) equation with a constant mobility and a double-well potential $G_{SQ} : \mathbb{R} \rightarrow \mathbb{R}$

$$G_{SQ}(\phi) = \frac{15}{4}(1 - \phi^2)^2.$$

Due to the constant mobility, the values of ϕ may leave the interval $[-1, 1]$ since there is no maximum principle, so G_{SQ} is defined on all of \mathbb{R} .

Finally, we consider a different physical system, namely crystal growth. Here, ϕ does no longer represent volume fractions, but atomic height units of a crystal. Thus, mobility and potential are defined on $[0, 1]$ and then continued periodically. However, for the analysis, one should consider a potential with finitely many wells and then, say, quartic growth, so that G_{EG} is coercive. The equations for

- *epitaxial growth* use the continuously differentiable mobility $M_{EG} : \mathbb{R} \rightarrow \mathbb{R}$

$$M_{EG}(\phi) = (1 + c\phi^{p+2}(1 - \phi)^2)^{-1},$$

which is bounded away from zero: $0 < c(p, \varepsilon, \zeta^-) \leq M_{EG}(\phi) \leq 1$. Here, $p \gg 1$ is a numerical parameter. In $[0, 1]$, the multi-well potential $G_{EG} : \mathbb{R} \rightarrow \mathbb{R}$ is given by

$$G_{EG}(\phi) = 18\phi^2(1 - \phi)^2.$$

The properties of the different mobilities and potentials are summarized in Table 2.1.

Remark 2.10. Due to the periodicity, the potential for epitaxial growth is only in C^2 . As can be seen from the asymptotic expansion in Otto et al. [2004], the potential has to fulfil

- G is a periodic function with period one,
- G vanishes for all integers,
- G is normalized by $\int_0^1 \sqrt{2G(x)} \, dx = 1$.

A simple function with these properties is G_{EG} . If a potential in C^∞ is favoured, another possible choice is

$$\tilde{G}_{\text{EG}}(\phi) = \frac{\pi^2}{16} (1 - \cos(2\pi\phi)).$$

The optimal profile (see Section 2.6.3) is then

$$\phi^*(x) = \frac{2}{\pi} \arctan\left(\exp\left(\frac{\pi^2}{2\varepsilon}x\right)\right).$$

However, in this work and in the numerics, we will use the potential G_{EG} . ◇

2.5 Existence of solutions

We review here some results for the two-dimensional case.

Remark 2.11. In most articles, an energy with a parameter γ in front of the Dirichlet term is assumed,

$$E_\gamma(\phi) = \int_\Omega \frac{\gamma}{2} |\nabla\phi|^2 + G(\phi).$$

For our purposes, set $\gamma = \varepsilon^2$ and rescale time $t = \varepsilon^{-1}\hat{t}$ and energy $E = \varepsilon\hat{E}$. This yields

$$\hat{E}(\phi) = \int_\Omega \frac{\varepsilon}{2} |\nabla\phi|^2 + \frac{1}{\varepsilon}G(\phi), \tag{2.19}$$

which is the energy used in epitaxial growth. For (SQ), (VQ) and (DQ), simply set $\varepsilon = 1$. ◇

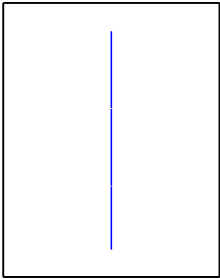
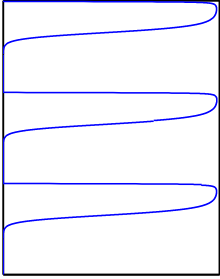
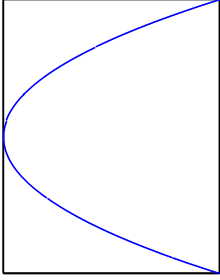
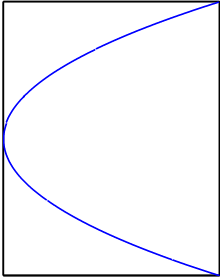
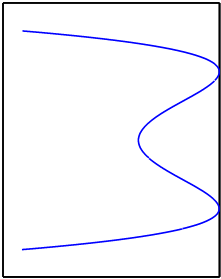
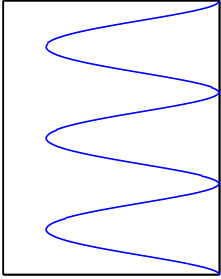
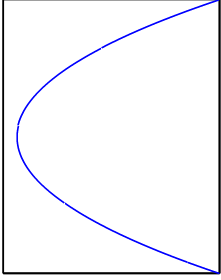
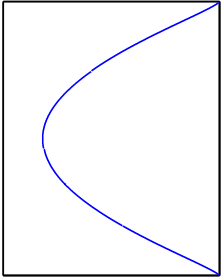
2.5.1 Constant mobility

In the constant mobility case, Equation (2.1) with (BC2) becomes

$$\partial_t\phi + \varepsilon\Delta^2\phi - \varepsilon^{-1}\Delta G'(\phi) = 0.$$

There is a lot of literature on this equation from which we pick the following two results, valid for a double-well potential and therefore covering (SQ):

Table 2.1: Properties of the mobilities and potentials considered in this work. Note that for all mobilities we have $0 \leq M(\phi) \leq 1$ and for all potentials the curvature is bounded from below, $G''(\phi) \geq -\beta$. For (VQ), the wells are hard to see, since they are very close to ± 1 .

	(SQ)	(EG)	(DQ)	(VQ)
Range of ϕ	\mathbb{R}	$[0, 1] + \text{per}$	$[-1, 1]$	$[-1, 1]$
Mobility $M(\phi)$	$\equiv 1$	$1/(1 + c\phi^{22}(1 - \phi)^2)$	$1 - \phi^2$	$1 - \phi^2$
Type	constant	variable, non-degenerate, C^1	degenerate	degenerate
Range	$\equiv 1$	$[\theta, 1], \theta > 0$	$[0, 1]$	$[0, 1]$
				
Potential $G(\phi)$	$\frac{15}{4}(1 - \phi^2)^2$	$18\phi^2(1 - \phi)^2$	$\frac{15}{2}(1 - \phi^2)$	$\frac{5}{2}(4(1 - \phi^2) + \mathfrak{E}(\phi))$
Type	double-well	multi-well, C^2	concave	logarithmic double-well
Range	$[0, \infty)$	$[0, 9/8]$	$[0, 15/2]$	$(3.46, 10]$
				
$G''(\phi) \geq$	-15	-18	-15	-15

Theorem 2.12 (Elliott and Songmu [1986, Remark 2]). *Let Ω be a bounded domain in \mathbb{R}^2 with a smooth boundary Γ and exterior normal ν . Assume $G'(\phi) = \gamma_2\phi^3 + \gamma_1\phi^2 - \phi$ with $\gamma_2 > 0$. For $\phi_0 \in H^2(\Omega)$ with $\partial\phi_0/\partial\nu = 0$ on Γ , there exists a unique global solution $\phi \in H^{4,1}(\Omega \times (0, T))$.*

Theorem 2.13 (Novick-Cohen [1998, Theorem 3.1]). *Let Ω be a bounded domain in \mathbb{R}^2 and assume $G'(\phi) = \phi^3 - \phi$. If $\phi_0 \in (H^1)'(\Omega)$, then there exists a unique solution $\phi(x, t) \in C([0, T], (H^1)'(\Omega))$ such that*

$$\begin{aligned}\phi &\in L^\infty(\delta, \infty; H^1(\Omega)) \cap L^\infty(\delta, \infty; L^4(\Omega)), \\ \partial_t \phi &\in L^2(\delta, \infty; (H^1)'(\Omega)).\end{aligned}$$

2.5.2 Variable mobility

The first to consider a variable mobility in more than one space dimension were Elliott and Garcke [1996], proving existence of weak solutions for both the non-degenerate and the degenerate case. Using (2.17), we write (2.1) as

$$\partial_t \phi - \nabla \cdot M(\phi) \nabla w = 0, \quad (2.20a)$$

$$w = -\varepsilon \Delta \phi + \varepsilon^{-1} G'(\phi) \quad (2.20b)$$

with the boundary conditions (BC2)

$$\nabla \phi \cdot \nu = 0 \quad \text{and} \quad \nabla w \cdot \nu = 0 \quad \text{on} \quad \partial\Omega \times (0, T).$$

In the case of a non-degenerate variable mobility, we have

Theorem 2.14 (Elliott and Garcke [1996, Theorem 2]). *Let Ω be a bounded domain in \mathbb{R}^2 with Lipschitz boundary. Assume that M is continuous on \mathbb{R} and there exist $m_1, M_1 > 0$ such that $m_1 \leq M(\phi) \leq M_1$. Furthermore, assume that $G \in C^1(\mathbb{R})$ and there exist constants $C_1, C_2, C_3, q > 0$ such that*

$$G(\phi) \geq -C_1 \quad \text{and} \quad |G'(\phi)| \leq C_2 |\phi|^q + C_3.$$

If $\phi_0 \in H^1(\Omega)$, then there exists a pair of functions (u, w) such that

$$\begin{aligned}\phi &\in L^\infty(0, T; H^1(\Omega)) \cap C([0, T]; L^2(\Omega)), \\ \partial_t \phi &\in L^2(0, T; (H^1)'(\Omega)), \\ w &\in L^2(0, T; H^1(\Omega)),\end{aligned}$$

which satisfy (2.20) in the following weak sense

$$\int_0^T \langle \zeta, \partial_t \phi \rangle = - \int_{\Omega \times (0, T)} M(\phi) \nabla w \cdot \nabla \zeta$$

for all $\zeta \in L^2(0, T; H^1(\Omega))$ and

$$\int_\Omega w \psi = \varepsilon \int_\Omega \nabla \phi \cdot \nabla \psi + \varepsilon^{-1} \int_\Omega G'(\phi) \psi$$

for all $\psi \in H^1(\Omega)$ and almost all $t \in [0, T]$.

If the initial datum is more regular, one can even get classical solutions for the case of a non-degenerate mobility:

Theorem 2.15 (Liu et al. [2006]). *Let Ω be a bounded domain in \mathbb{R}^2 with smooth boundary. Assume that $M(\phi) \in C^1(\mathbb{R})$ with bounded M' and there exist $m_1, M_1 > 0$ such that $m_1 \leq M(\phi) \leq M_1$. Furthermore, assume that $G \in C^3(\mathbb{R})$ and there exist constants C_1, \dots, C_4 such that*

$$G(\phi) \geq C_1 |\phi|^4 - C_2 \quad \text{and} \quad |G''(\phi)| \leq C_3 |\phi|^2 + C_4.$$

If ϕ_0 is smooth with mean zero and appropriate compatibility conditions, then the problem (2.20) admits a unique classical solution.

These two theorems are also valid for the case $M(\phi) \equiv \text{const}$, thus covering (SQ) and (EG), but do not allow for the logarithmic potential. In contrast, the following theorem allows (and requires) a degenerate mobility and is valid for logarithmic and concave potentials and therefore covers (DQ) and (VQ). Actually, more general potentials including the double-well potential are considered in the article, which we neglect here to reduce notation overhead.

Theorem 2.16 (Elliott and Garcke [1996, Theorem 1]). *Let either $\partial\Omega \in C^{1,1}$ or Ω convex. Assume that G is either the logarithmic potential G_{VQ} or the concave potential G_{DQ} . Suppose furthermore $\phi_0 \in H^1(\Omega)$ with $|\phi_0| \leq 1$ a.e. and*

$$\int_{\Omega} (G(\phi_0) + \Psi(\phi_0)) \leq C, \quad C \in \mathbb{R}^+,$$

where Ψ is defined by

$$\Psi''(\phi) = (1 - \phi^2)^{-1}, \quad \Psi'(0) = 0 \quad \text{and} \quad \Psi(0) = 0.$$

Then there exists a pair (ϕ, J) such that

$$\begin{aligned} \phi &\in L^2(0, T; H^2(\Omega)) \cap L^\infty(0, T; H^1(\Omega)) \cap C([0, T]; L^2(\Omega)), \\ \partial_t \phi &\in L^2(0, T; (H^1)'(\Omega)), \\ |\phi| &\leq 1 \text{ almost everywhere in } \Omega \times (0, T), \\ J &\in L^2(\Omega \times (0, T), \mathbb{R}^2) \end{aligned}$$

which satisfies the equation

$$\begin{aligned} \partial_t \phi + \nabla \cdot J &= 0, \\ J &= -M(\phi) \nabla (-\varepsilon \Delta \phi + \varepsilon^{-1} G'(\phi)) \end{aligned}$$

in the following weak sense

$$\begin{aligned} \int_0^T \langle \zeta(t), \partial_t \phi(t) \rangle &= \int_{\Omega \times (0, T)} J \cdot \nabla \zeta \quad \forall \zeta \in L^2(0, T; H^1(\Omega)) \\ \int_{\Omega \times (0, T)} J \cdot \nu &= - \int_{\Omega \times (0, T)} \varepsilon \Delta \phi \nabla \cdot (M(\phi) \eta) + \varepsilon^{-1} (MG'')(\phi) \nabla \phi \cdot \eta \end{aligned}$$

for all $\eta \in L^2(0, T; H^1(\Omega, \mathbb{R}^2)) \cap L^\infty(\Omega \times (0, T), \mathbb{R}^2)$ which fulfil $\eta \cdot \nu = 0$ on $\partial\Omega \times (0, T)$.

2.6 Morphology of solutions

To illustrate the typical behaviour of a solution to a Cahn–Hilliard-type equation, we return to the variable quench equation.

Consider again the experiment mentioned in Section 2.4, where we had a homogeneous mixture of two components, i.e. $\phi_0 \equiv \text{const}$. When the temperature is lowered to a value $T = 1/\beta < 1$, the potential becomes a double-well (see Figure 5.1 on page 91) and, depending on the volume fractions of the mixture, the homogeneous state may become unstable and then the mixture starts to decompose into its components.

If the temperature, the mixture is quenched to, is close to $T = 1$, we are in the shallow quench regime. If the mixture is quenched to $T = 0$, we are in the deep quench regime. Given that the volume fractions are such that the state is unstable (that is the case if $G''(\phi_0) < 0$, see next section), the development of the initial data can be roughly divided into two parts:

1. Spinodal decomposition: In this initial regime, a most unstable wavelength λ^* develops and then grows in amplitude until it reaches the minima g^\pm of the potential, which are called the bulk equilibrium values.
2. Interfacial regime: After the first phase, we are in a regime of *developed interfaces* where the values of ϕ are mostly at g^\pm with thin interfaces in-between.

We now describe these two regimes in a little more detail.

2.6.1 Spinodal decomposition

To check the stability of a homogeneous mixture, we insert initial data of the form

$$\phi = \phi_0 + \delta h,$$

where ϕ_0 is constant, $\delta \ll 1$ and h is an $\mathcal{O}(1)$ -function, into the Cahn–Hilliard equation with Ginzburg–Landau-type energy, namely

$$\partial_t \phi + \nabla \cdot [M(\phi) \nabla (\Delta \phi - G'(\phi))] = 0, \quad (2.21)$$

where we assume $M(\phi_0) > 0$. This yields

$$\delta \partial_t h + \nabla \cdot [M(\phi_0 + \delta h) \nabla (\delta \Delta h - G'(\phi_0 + \delta h))] = 0.$$

Since

$$\begin{aligned} M(\phi_0 + \delta h) &= M(\phi_0) + \delta M'(\phi_0)h + \mathcal{O}(\delta^2), \\ G'(\phi_0 + \delta h) &= G'(\phi_0) + \delta G''(\phi_0)h + \mathcal{O}(\delta^2), \end{aligned}$$

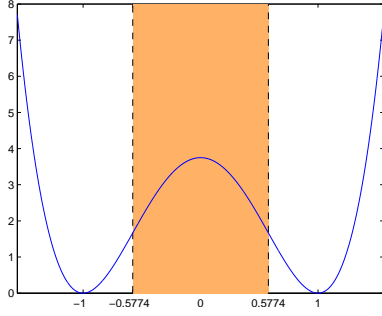


Figure 2.1: Double-well potential for shallow quench. The orange area is the spinodal interval, i.e. initial values $\phi_0 \equiv \text{const}$ in this region are unstable.

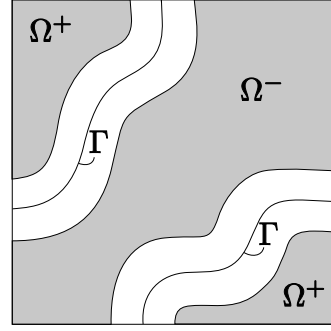


Figure 2.2: At later times, the domain decomposes into areas where $\phi \approx g^\pm$ (grey areas) and transition regions (white areas).

we have

$$\begin{aligned} \delta \partial_t h + \nabla \cdot \left[(M(\phi_0) + \delta M'(\phi_0)h + \mathcal{O}(\delta^2)) \nabla (\delta \Delta h - G'(\phi_0) - \delta G''(\phi_0)h + \mathcal{O}(\delta^2)) \right] &= 0 \\ \iff \delta \partial_t h + \delta \nabla \cdot [M(\phi_0) \nabla (\Delta h - G''(\phi_0)h)] + \mathcal{O}(\delta^3) &= 0. \end{aligned}$$

Therefore, the $\mathcal{O}(\delta)$ -equation is

$$\partial_t h + M(\phi_0) (\Delta^2 h - G''(\phi_0) \Delta h) = 0.$$

Fourier transformation in the space variable yields

$$\frac{d}{dt} \mathcal{F}(h) = -M(\phi_0) (|k|^4 + G''(\phi_0) |k|^2) \mathcal{F}(h)$$

and the solution to this ODE is

$$\mathcal{F}(h) = h_0 \exp(-M(\phi_0) (|k|^4 + G''(\phi_0) |k|^2) t) =: h_0 \exp(\omega(|k|) t)$$

with the dispersion relation

$$\omega(k) = -M(\phi_0) (|k|^4 + G''(\phi_0) |k|^2).$$

So the mixture is stable if $G''(\phi_0) \geq 0$ and unstable for some $|k|$ if $G''(\phi_0) < 0$. The points where $G''(\phi_0) = 0$ are called the spinodal points, and the interval between them, where $G'' < 0$, is the spinodal interval. See Figure 2.1 for a plot in the case of G_{SQ} .

If ϕ_0 is unstable, the most unstable wave number is

$$\omega'(k^*) = 0 \iff k^* = \sqrt{-\frac{1}{2} G''(\phi_0)}. \quad (2.22)$$

Since $G''(\phi_0) < 0$, k^* is well defined. For the most unstable wavelength λ^* , we get

$$\lambda^* = \frac{2\pi}{k^*} = 2\pi \sqrt{\frac{-2}{G''(\phi_0)}}. \quad (2.23)$$

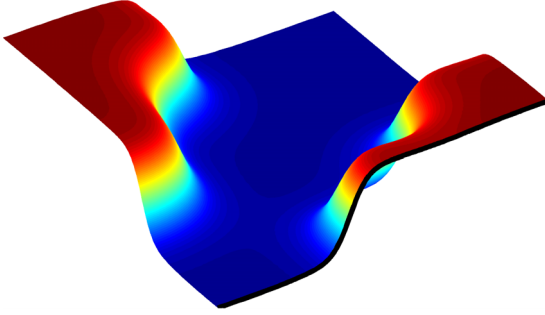


Figure 2.3: On a fast timescale, the transition profile approaches the optimal profile: the thick black line is tanh-like.

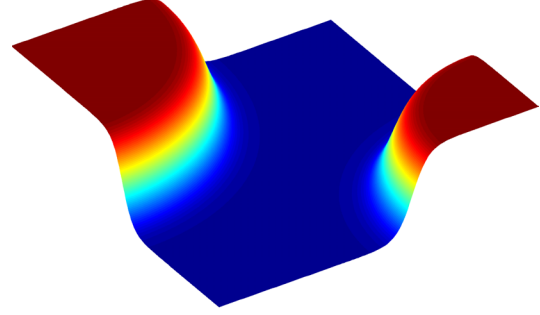


Figure 2.4: On a slow timescale, the interface evolves according to the Mullins–Sekerka free boundary problem: the islands become circular and the larger island grows at the expense of the small one (Ostwald ripening).

2.6.2 Interfacial regime

After the spinodal decomposition, we expect the following structure of the solution: Ω is separated into regions $\Omega^+(t)$ and $\Omega^-(t)$ where the solution is close to the bulk equilibrium values:

$$\phi \approx g^\pm \text{ in } \Omega^\pm(t).$$

In-between these parts are thin transition regions with “centre” $\Gamma(t)$, see Figure 2.2.

Various aspects of the morphology of the solution take place on different time scales, so we rescale time and space by

$$t \mapsto \varepsilon^\alpha t \quad \text{and} \quad x \mapsto \varepsilon x, \quad (2.24)$$

where we assume $\varepsilon \ll 1$.

- For the kind of data just described and the *shallow quench* regime, i.e. for the Cahn–Hilliard equation, Pego [1989] investigated (for $f = 0$) the behaviour using formal asymptotic expansion. He found that on a fast timescale ($\alpha = 0$), the solution in the transition layer approaches the optimal transition profile, see Figure 2.3. This optimal profile will be computed in Section 2.6.3.

On a much slower timescale ($\alpha = 3$), the centre manifold $\Gamma(t)$ evolves to highest order according to the non-local *Mullins–Sekerka* free boundary problem

$$\begin{aligned} -\Delta w^\pm &= 0 && \text{in } \Omega \setminus \Gamma(t), \\ w^\pm &= \sqrt{\frac{10}{3}} \kappa && \text{at } \Gamma(t), \\ \nabla w^\pm \cdot \nu &= 0 && \text{on } \partial\Omega. \end{aligned}$$

The interface $\Gamma(t)$ moves with normal velocity

$$V = \frac{1}{2} \nabla(w^+ - w^-) \cdot \nu,$$

where κ is the curvature of Γ and ν the outer normal.

This was made rigorous by Alikakos, Bates, and Chen [1994], who calculated the error between the true solution of the Cahn–Hilliard equation and the formal expansion, thereby gaining a rigorous proof of the convergence $\varepsilon \rightarrow 0$ under the assumption that a classical solution of the Mullins–Sekerka free-boundary problem exists.

- For the *deep quench* (and *variable quench* for very low temperature), Cahn, Elliott, and Novick-Cohen [1996] showed by formal asymptotic expansion that in the scaling $\alpha = 4$, the sharp-interface problem is *motion by surface diffusion*, i.e. the normal velocity of the sharp interface is given by

$$V \simeq \Delta_s \kappa,$$

where Δ_s is the Laplace–Beltrami operator on Γ . The sign is opposite to Cahn et al. [1996] since their normal has opposite sign.

- Otto, Penzler, Rätz, Rump, and Voigt [2004] showed by formal asymptotic expansion that in the case of *epitaxial growth*, the interface evolves on the timescale $\alpha = 3$ according to a free boundary problem similar to Mullins–Sekerka:

$$\begin{aligned} -\Delta w &= f && \text{in } \Omega \setminus \Gamma(t), \\ w^+ &= \kappa && \text{at } \Gamma(t), \\ w^- + \zeta \frac{\partial w^-}{\partial \nu} &= \kappa && \text{at } \Gamma(t) \end{aligned}$$

with normal velocity

$$V = \nabla(w^+ - w^-) \cdot \nu.$$

See Sections 6.3 and 6.4 for details.

Knowing that these Cahn–Hilliard-type equations have sharp-interface limits, we can deduce their long-time behaviour from the properties of the limit models.

The sharp-interface limits also have a gradient-flow structure (if $f = 0$) and for all of them the length of the interface is the energy functional. Therefore, we infer that the Cahn–Hilliard-type equations reduce the length of the interface over time. One obvious consequence is that if Ω^+ is simply connected, it will become a ball. This is then a steady state.

The gradient-flow structure for the sharp-interface limits is lined out the Section A.1.

2.6.3 Optimal profile

To compute the optimal transition profile, we reduce the problem to a one-dimensional situation. To that end, we introduce a normal–tangential coordinate system (s, r) along the interface, where s measures the position on the interface and r the distance from the interface. Finally, we stretch the variable in normal direction such that the “interface ends at $\pm\infty$ ”.

For a fixed s , we search a function $u^*(z)$ which minimizes the energy subject to the boundary values

$$\lim_{z \rightarrow \pm\infty} u^*(z) = g^\pm.$$

Using the scaling (2.19), the energy to be minimized is

$$E_\varepsilon(\phi) = \int_\Omega \frac{\varepsilon}{2} |\nabla \phi|^2 + \varepsilon^{-1} G(\phi).$$

Following Modica and Mortola [1977], we use Young's inequality to see that

$$E_\varepsilon(\phi) \geq \int_\Omega |\nabla \phi| \sqrt{2G(\phi)}$$

and “=” if and only if

$$\varepsilon |\nabla \phi| = \sqrt{2G(\phi)},$$

which is equipartition of energy. In one space dimension, the above relation is the ordinary differential equation

$$\varepsilon \partial_x \phi = \pm \sqrt{2G(\phi)}.$$

For the potentials of the shallow-quench (SQ) and deep-quench (DQ) Cahn–Hilliard equations, the respective one-dimensional solutions are

$$u_{\text{SQ}}^*(z) = \tanh\left(\sqrt{\frac{15}{2}}z\right) \quad \text{and} \quad u_{\text{DQ}}^*(z) = \begin{cases} -1 & z \leq -\pi/\sqrt{60} \\ \sin(\sqrt{15}z) & |z| \leq \pi/\sqrt{60} \\ 1 & z \geq \pi/\sqrt{60} \end{cases}, \quad (2.25)$$

where we set $\varepsilon = 1$ and selected the solution with $u^*(0) = 0$. For the variable quench, there is no closed form. In Figure 2.5, we plotted these profiles.

Finally, using G_{EG} as potential and requesting $u^*(0) = 1/2$, we obtain

$$u_{\text{EG}}^*(z) = \frac{1}{2} + \frac{1}{2} \tanh(3\varepsilon^{-1}z).$$

The solution has values very close to the minima of G most of the time and has a very thin connecting interface, see Figure 2.6 for a plot of the profile and for the width of the interface. Since the bulk equilibrium values are only attained at $\pm\infty$, the width depends on where one defines the beginning and end of the interface.

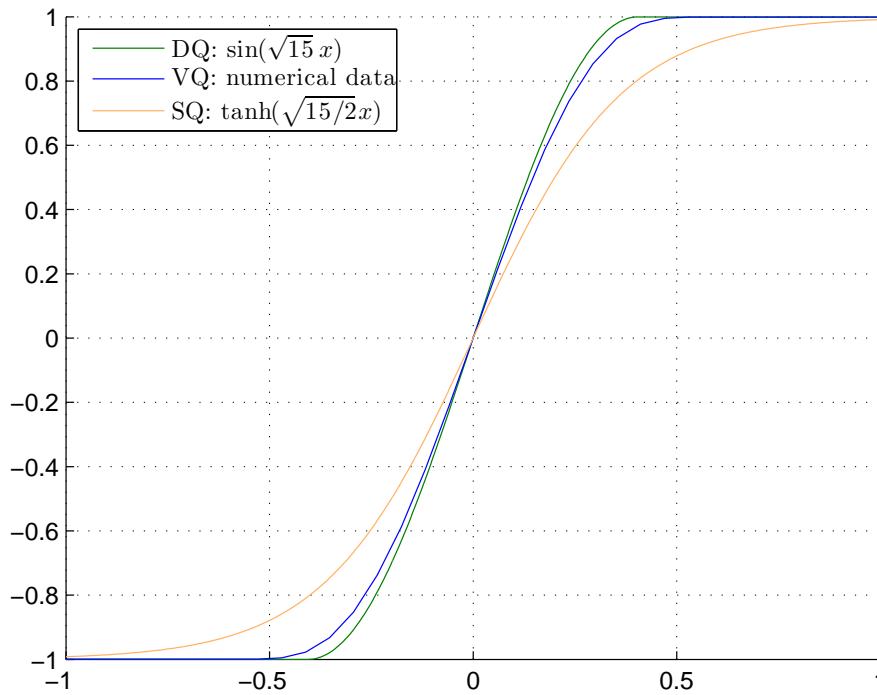
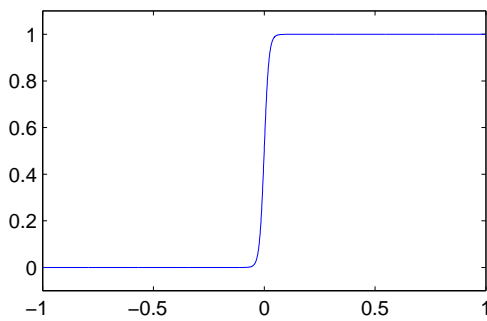


Figure 2.5: The optimal transition profiles of the deep-, variable- and shallow-quench Cahn–Hilliard equations. For variable quench, a value of $\beta = 4$ for the inverse temperature was used.



dist to b.e.v.	Width of EG-interface
$5 \cdot 10^{-2}$	0.98ε
10^{-2}	1.53ε
10^{-3}	2.31ε
10^{-4}	3.07ε

Figure 2.6: Plot of the minimizer u_{EG}^* of E_ε for $\varepsilon = 1/16$. The table shows the width of the interface. The width is measured between the points having the indicated distance to the bulk equilibrium values, which are here zero and one.

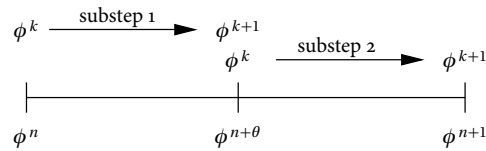
Time Discretization

In the following two chapters, we discretize Equation (2.3). We use Rothe's method and discretize first in time (this chapter) and then in space (Chapter 4), so that we can change the mesh between time steps. The discretization in time yields a symmetric system, which is positive definite for time steps that are small enough. In the worst case, the time steps are bounded by the usual ε^3 . However, in most of the situations the bound is only a constant independent of ε .

First a word on notation. Since we take into consideration a number of time-step schemes, we try to unify the notation. Some of the schemes divide one time step into one or more sub-steps

$$[n\tau, (n+1)\tau] \mapsto [n\tau, (n+\theta)\tau] \cup \dots, \quad \theta \in (0, 1).$$

To distinguish between the time steps of length τ and the sub-steps of length $< \tau$, we enumerate the time steps with n and the sub-steps with k . Here, the sub-steps are counted from the point of view of the current sub-step:



In each of the sub-steps, the schemes again have the form of a one-step scheme (there will be one exception though), so it is enough to consider a one-step method valid for *one sub-interval*.

Summing up, we use a semi-implicit time discretization

$$\frac{1}{a\tau}(\phi^{k+1} - \phi^k) = F(\phi^{k+1}) + \gamma F(\phi^k), \quad (3.1)$$

where $a \in (0, 1)$ and $\gamma \geq 0$ are constants which depend on the choice of the time-step scheme, see Table B.1 on page 163 for an overview.

3.1 Derivation of the time-discrete equations

The guideline for our discretization is the gradient-flow structure outlined in Section 2.3. That is, we discretize the equation

$$\partial_t \phi = -\nabla_g E(\phi).$$

Note that the ideas presented in this section are independent of the concrete form of the energy.

Remember that the metric is not constant on \mathcal{M} , but depends on $\phi \in \mathcal{M}$. We use an explicit discretization for this base point and the semi-implicit discretization (3.1) for the gradient:

$$\partial_t^h \phi^{k+1} = -\nabla_{g^k} E(\phi^{k+1}) - \gamma \nabla_{g^k} E(\phi^k), \quad (3.2)$$

where we used the abbreviation

$$\partial_t^h \phi^{k+1} := \frac{1}{a\tau} (\phi^{k+1} - \phi^k).$$

Repeating the calculations of Section 2.3, we get

$$\partial_t^h \phi^{k+1} + \nabla \cdot J_* = 0, \quad (3.3a)$$

$$\int_{\Omega} \frac{1}{M(\phi^k)} J_* \cdot \tilde{J} - \int_{\Omega} \left(\frac{\delta E}{\delta \phi}(\phi^{k+1}) + \gamma \frac{\delta E}{\delta \phi}(\phi^k) \right) (\nabla \cdot \tilde{J}) = 0. \quad (3.3b)$$

To incorporate the source term f , compare these equations to (2.3b). Defining J^{k+1} and J^k by

$$\begin{aligned} \int_{\Omega} \frac{1}{M(\phi^k)} J^{k+1} \cdot \tilde{J} - \int_{\Omega} \frac{\delta E}{\delta \phi}(\phi^{k+1}) (\nabla \cdot \tilde{J}) &= 0, \\ \int_{\Omega} \frac{1}{M(\phi^k)} J^k \cdot \tilde{J} - \int_{\Omega} \frac{\delta E}{\delta \phi}(\phi^k) (\nabla \cdot \tilde{J}) &= 0, \end{aligned}$$

we find

$$J_* = J^{k+1} + \gamma J^k,$$

so that we can conclude that the driving force f has to appear as $f^{k+1} + \gamma f^k$ on the right hand side. We assume for simplicity that f is constant in time, since this is the case in all our examples. Then, the time-discrete equations are

$$\partial_t^h \phi^{k+1} + \nabla \cdot J_* = (1 + \gamma)f, \quad (3.4a)$$

$$\int_{\Omega} \frac{1}{M(\phi^k)} J_* \cdot \tilde{J} - \int_{\Omega} \left(\frac{\delta E}{\delta \phi}(\phi^{k+1}) + \gamma \frac{\delta E}{\delta \phi}(\phi^k) \right) (\nabla \cdot \tilde{J}) = 0. \quad (3.4b)$$

Since the variational derivative of E will be nonlinear in typical applications, we use Newton's method. To find the right form for Newton's method, let us take a look back at the strong version of the equation:

$$\begin{aligned} \frac{1}{a\tau}(\phi^{k+1} - \phi^k) + \nabla \cdot J_* &= (1 + \gamma)f, \\ \frac{1}{M(\phi^k)} J_* &= -\nabla \left(\frac{\delta E}{\delta \phi}(\phi^{k+1}) + \gamma \frac{\delta E}{\delta \phi}(\phi^k) \right). \end{aligned}$$

In order to avoid multiple indices, we just write ϕ instead of ϕ^{k+1} . Inserting the second equation into the first yields

$$\mathcal{F}(\phi) := \frac{1}{a\tau}(\phi - \phi^k) + \nabla \cdot \left[-M(\phi^k) \nabla \left(\frac{\delta E}{\delta \phi}(\phi) + \gamma \frac{\delta E}{\delta \phi}(\phi^k) \right) \right] - (1 + \gamma)f = 0.$$

Then the differential of \mathcal{F} is

$$\text{diff } \mathcal{F}(\phi)\psi = \frac{1}{a\tau}\psi + \nabla \cdot \left[-M(\phi^k) \nabla \frac{\delta^2 E}{\delta \phi^2}(\phi)\psi \right].$$

Therefore, Newton's method

$$\mathcal{F}(\phi^i) + \text{diff } \mathcal{F}(\phi^i)(\phi^{i+1} - \phi^i) = 0$$

amounts to

$$\begin{aligned} \frac{1}{a\tau}(\phi^i - \phi^k) + \frac{1}{a\tau}(\phi^{i+1} - \phi^i) \\ + \nabla \cdot \left[\underbrace{-M(\phi^k) \nabla \left(\frac{\delta E}{\delta \phi}(\phi^i) + \gamma \frac{\delta E}{\delta \phi}(\phi^k) + \frac{\delta^2 E}{\delta \phi^2}(\phi^i)(\phi^{i+1} - \phi^i) \right)}_{=: J_{\text{strong}}^{i+1}} \right] - (1 + \gamma)f = 0. \end{aligned}$$

Let us again mention the notation: we write ϕ^{i+1} for $\phi^{k+1, i+1}$ and have

$$\phi^{k+1, 0} = \phi^k \quad \text{and} \quad \phi^{k+1, I} =: \phi^{k+1},$$

where $I \geq 1$ is the number of Newton steps ($I = 1$ corresponds to the linearization of the nonlinearity). Transferring the above to the weak formulation leads us to

$$\frac{1}{a\tau}(\phi^{i+1} - \phi^k) + \nabla \cdot J_*^{i+1} = (1 + \gamma)f \quad (3.5a)$$

$$\int_{\Omega} \frac{1}{M(\phi^k)} J_*^{i+1} \cdot \tilde{J} - \int_{\Omega} \left(\frac{\delta E}{\delta \phi}(\phi^i) + \gamma \frac{\delta E}{\delta \phi}(\phi^k) + \frac{\delta^2 E}{\delta \phi^2}(\phi^i)(\phi^{i+1} - \phi^i) \right) (\nabla \cdot \tilde{J}) = 0. \quad (3.5b)$$

To be able to insert the first equation into the second, we modify the second equation by writing

$$\phi^{i+1} - \phi^i = (\phi^{i+1} - \phi^k) - (\phi^i - \phi^k).$$

Furthermore, we divide (3.4b) by $a\tau$ and, to simplify notation, define

$$w^* := w^i + \gamma w^k \quad \text{with} \quad w^i := \frac{\delta E}{\delta \phi}(\phi^i), \quad w^k := \frac{\delta E}{\delta \phi}(\phi^k). \quad (3.6)$$

This yields for Equation (3.5b)

$$\begin{aligned} \int_{\Omega} \frac{1}{a\tau M(\phi^k)} J_*^{i+1} \cdot \tilde{J} - \int_{\Omega} \frac{\delta^2 E}{\delta \phi^2}(\phi^i) \frac{1}{a\tau} (\phi^{i+1} - \phi^k) (\nabla \cdot \tilde{J}) \\ = \int_{\Omega} \frac{1}{a\tau} \left(w^* - \frac{\delta^2 E}{\delta \phi^2}(\phi^i) (\phi^i - \phi^k) \right) (\nabla \cdot \tilde{J}). \end{aligned}$$

Now inserting Equation (3.5a) into the above equation, we can eliminate ϕ^{i+1} in the flux-equation and get for the i -th Newton-step the transport- and flux-equations

$$\frac{1}{a\tau} (\phi^{i+1} - \phi^k) + \nabla \cdot J_*^{i+1} = (1 + \gamma)f \quad (3.7a)$$

$$\begin{aligned} \int_{\Omega} \frac{1}{a\tau M(\phi^i)} J_*^{i+1} \cdot \tilde{J} + \int_{\Omega} \frac{\delta^2 E}{\delta \phi^2}(\phi^i) (\nabla \cdot J_*^{i+1}) (\nabla \cdot \tilde{J}) \\ = \int_{\Omega} \left(\frac{1}{a\tau} w^* + \frac{\delta^2 E}{\delta \phi^2}(\phi^i) \left((1 + \gamma)f - \frac{1}{a\tau} (\phi^i - \phi^k) \right) \right) (\nabla \cdot \tilde{J}). \quad (3.7b) \end{aligned}$$

Here we also replaced $M(\phi^k)$ by $M(\phi^i)$, since it seems reasonable to use always the latest available ϕ for the mobility. Putting things together, we get three loops for a solving procedure:

- an outer loop counting the time steps n ,
- an intermediate loop counting the sub-timesteps (one to three depending on the time-step scheme) and
- an inner loop performing the Newton-steps i . As Newton's method converges quadratically when close enough to the solution, it is usually enough to have two or three Newton-steps, see Figure A.6 on page 159 for an example. The inner loop consists of two parts:
 1. Given ϕ^i and ϕ^k , find J_*^{i+1} such that Equation (3.7b) is satisfied.
 2. Use Equation (3.7a) to calculate ϕ^{i+1} .

Remark 3.1. One advantage of the discretization via the flux J is the possibility to use skew-periodic boundary conditions, see Section 7.2.2 for details on this. This is crucial for the simulation of certain phenomena in epitaxial growth, see Chapter 6. \diamond

3.2 Properties of the time-discrete operator

In this section we consider Ginzburg–Landau-type energies $E : H^1(\Omega) \rightarrow \mathbb{R}$,

$$E(\phi) = \frac{\varepsilon}{2} \int_{\Omega} |\nabla \phi|^2 + \varepsilon^{-1} \int_{\Omega} G(\phi)$$

and assume $G''(x) \geq -\beta$. For the mobility, we assume $0 \leq M(\phi) \leq 1$. These assumptions hold for all our applications, compare to Table 2.1 on page 15. Furthermore, we write $\tau_a := a\tau$. The derivatives of the energy are

$$\frac{\delta E}{\delta \phi}(\phi) = -\varepsilon \Delta \phi + \varepsilon^{-1} G'(\phi) \quad \text{and} \quad \frac{\delta^2 E}{\delta \phi^2}(\phi) = -\varepsilon \Delta + \varepsilon^{-1} G''(\phi).$$

So given $\phi^i \in H^1(\Omega)$, we have

$$\frac{\delta E}{\delta \phi}(\phi^i) =: w^i \in H^{-1}(\Omega)$$

and also

$$\frac{\delta^2 E}{\delta \phi^2}(\phi^i) \left((1+\gamma)f - \frac{1}{\tau_a}(\phi^i - \phi^k) \right) =: r_2 \in H^{-1}(\Omega)$$

for given $f \in H^1(\Omega)$. Therefore, setting

$$r := \frac{1}{\tau_a} w^* + r_2, \tag{3.8}$$

where w^* is as defined in (3.6), we have $r \in H^{-1}(\Omega)$. We need the following definition

Definition 3.2 (*H*-grad-div). The space $H(\nabla \nabla \cdot, \Omega)$ is the space of all vector fields $\xi \in L^2(\Omega)^2$ with $\nabla \cdot \xi \in H^1(\Omega)$. The norm is given by

$$\|\xi\|_{H(\nabla \nabla \cdot, \Omega)}^2 := \|\xi\|_{L^2(\Omega)}^2 + \|\nabla \cdot \xi\|_{L^2(\Omega)}^2 + \|\nabla \nabla \cdot \xi\|_{L^2(\Omega)}^2.$$

When the domain Ω is clear from the context, we may just write $H(\nabla \nabla \cdot)$. \diamond

The time-discrete problem (3.7) for Ginzburg–Landau-type energies is then

Given $\phi^k, \phi^i \in H^1(\Omega)$ and $f \in H^1(\Omega)$, find $J_*^{i+1} \in H(\nabla \nabla \cdot)$ such that

$$\int_{\Omega} \frac{1}{\tau_a M(\phi)} J_*^{i+1} \cdot \tilde{J} + \varepsilon \int_{\Omega} (\nabla \nabla \cdot J_*^{i+1}) \cdot (\nabla \nabla \cdot \tilde{J}) + \frac{1}{\varepsilon} \int_{\Omega} G''(\phi) (\nabla \cdot J_*^{i+1}) (\nabla \cdot \tilde{J}) = \int_{\Omega} r(\nabla \cdot \tilde{J}) \tag{3.9}$$

for all $\tilde{J} \in H(\nabla \nabla \cdot)$. Then, calculate $\phi^{i+1} \in H^1(\Omega)$ via

$$\phi^{i+1} = \phi^k + \tau_a \left(-\nabla \cdot J_*^{i+1} + (1+\gamma)f \right).$$

Since $J_*^{i+1} \in H(\nabla\nabla\cdot)$ and therefore $\nabla \cdot J_*^{i+1} \in H^1(\Omega)$, we have indeed $\phi^{i+1} \in H^1(\Omega)$ and so we can iterate the whole process. The boundary conditions for J_*^{i+1} and \tilde{J} are either periodic or $J \cdot \nu = 0$ and $(\nabla\nabla \cdot J) \cdot \nu = 0$ on $\partial\Omega$.

In order to write the time-discrete problem in a shorter way, we define the bilinear form $a : H(\nabla\nabla\cdot, \Omega) \times H(\nabla\nabla\cdot, \Omega) \rightarrow \mathbb{R}$,

$$a_\phi(J, \tilde{J}) = \int_\Omega \frac{1}{\tau_a M(\phi)} J \cdot \tilde{J} + \varepsilon \int_\Omega (\nabla\nabla \cdot J) \cdot (\nabla\nabla \cdot \tilde{J}) + \frac{1}{\varepsilon} \int_\Omega G''(\phi)(\nabla \cdot J)(\nabla \cdot \tilde{J}),$$

and the linear form $b : H(\nabla\nabla\cdot, \Omega) \rightarrow \mathbb{R}$,

$$b(\tilde{J}) = \int_\Omega r(\nabla \cdot \tilde{J}).$$

Using these definitions, Equation (3.9) is just

$$a_\phi(J_*^{i+1}, \tilde{J}) = b(\tilde{J}) \quad \forall \tilde{J} \in H(\nabla\nabla\cdot). \quad (3.10)$$

It remains to show that (3.9) admits a solution.

3.2.1 Existence of time-discrete solutions

To show that Problem (3.9) has a unique solution, we consider the minimization problem

$$\min_{\tilde{J} \in H(\nabla\nabla\cdot, \Omega)} \frac{1}{2} a_\phi(\tilde{J}, \tilde{J}) - b(\tilde{J}), \quad (3.11)$$

which has a unique solution in $H(\nabla\nabla\cdot)$ if a is coercive on $H(\nabla\nabla\cdot, \Omega)$. This solution then satisfies Equation (3.9).

Proposition 3.3. *For any $\gamma \in (0, \varepsilon)$, assume that the time-step size $\tau_a > 0$ is restricted by*

$$\tau_a \leq \frac{1}{\gamma + \frac{(\beta + \varepsilon\gamma)^2}{4(\varepsilon - \gamma)\varepsilon^2}}.$$

Then the bilinear form a_ϕ is coercive on $H(\nabla\nabla\cdot, \Omega)$.

Proof. Since $M(\phi) \leq 1$ and $G''(\phi) \geq -\beta$, we have

$$\begin{aligned} a_\phi(\xi, \xi) &\geq \frac{1}{\tau_a} \int_\Omega |\xi|^2 + \varepsilon \int_\Omega |\nabla\nabla \cdot \xi|^2 - \left(\frac{\beta}{\varepsilon} + \gamma \right) \int_\Omega |\nabla \cdot \xi|^2 + \gamma \int_\Omega |\nabla \cdot \xi|^2 \\ &= \frac{1}{\tau_a} \int_\Omega |\xi|^2 + \varepsilon \int_\Omega |\nabla\nabla \cdot \xi|^2 + \frac{\beta + \varepsilon\gamma}{\varepsilon} \int_\Omega (\nabla\nabla \cdot \xi) \cdot \xi + \gamma \int_\Omega |\nabla \cdot \xi|^2 \end{aligned}$$

and by Young's inequality

$$\begin{aligned}
&\geq \frac{1}{\tau_a} \int_{\Omega} |\xi|^2 + \varepsilon \int_{\Omega} |\nabla \nabla \cdot \xi|^2 + \gamma \int_{\Omega} |\nabla \cdot \xi|^2 \\
&\quad - \frac{\beta + \varepsilon \gamma}{\varepsilon} \left(\frac{\varepsilon(\varepsilon - \gamma)}{\beta + \varepsilon \gamma} \int_{\Omega} |\nabla \nabla \cdot \xi|^2 + \frac{\beta + \varepsilon \gamma}{4\varepsilon(\varepsilon - \gamma)} \int_{\Omega} |\xi|^2 \right) \\
&= \left(\frac{1}{\tau_a} - \frac{(\beta + \varepsilon \gamma)^2}{4(\varepsilon - \gamma)\varepsilon^2} \right) \int_{\Omega} |\xi|^2 + \gamma \int_{\Omega} |\nabla \cdot \xi|^2 + \gamma \int_{\Omega} |\nabla \nabla \cdot \xi|^2
\end{aligned}$$

and due to the restriction on τ_a this is

$$\geq \gamma \|\xi\|_{H(\nabla \nabla \cdot, \Omega)}^2.$$

□

Remark. Using a value of $\gamma = c\varepsilon$ with $0 < c < 1$ yields a restriction of the order ε^3 . The parameter γ used here has nothing to do with the parameter γ in the time-step scheme. ◇

Corollary 3.4 (Existence of a solution). *For $\gamma \in (0, \varepsilon)$ and*

$$\tau_a \leq \frac{1}{\gamma + \frac{(\beta + \varepsilon \gamma)^2}{4(\varepsilon - \gamma)\varepsilon^2}},$$

the problem (3.9) admits a unique solution $J_^{i+1} \in H(\nabla \nabla \cdot)$.*

We summarize the preceding in

Proposition 3.5. *Assume $\phi^0 \in H^1(\Omega)$ and $f \in H^1(\Omega)$. Then for all $i, k \in \mathbb{N}_0$ there exist $\phi^i, \phi^k \in H^1(\Omega)$ and $J_*^{i+1} \in H(\nabla \nabla \cdot)$ such that*

$$\begin{aligned}
\int_{\Omega} \frac{1}{a\tau M(\phi)} J \cdot \tilde{J} + \varepsilon \int_{\Omega} \nabla \nabla \cdot J \cdot \nabla \nabla \cdot \tilde{J} + \frac{1}{\varepsilon} \int_{\Omega} G''(\phi)(\nabla \cdot J)(\nabla \cdot \tilde{J}) &= \int_{\Omega} r(\nabla \cdot \tilde{J}), \\
\phi^{i+1} &= \phi^k + a\tau \left(-\nabla \cdot J_*^{i+1} + (1 + \gamma)f \right),
\end{aligned}$$

for all $\tilde{J} \in H(\nabla \nabla \cdot)$, where $r \in H^{-1}(\Omega)$ is given by

$$r = \frac{1}{a\tau} \left(\frac{\delta E}{\delta \phi}(\phi^i) + \gamma \frac{\delta E}{\delta \phi}(\phi^k) \right) + \frac{\delta^2 E}{\delta \phi^2}(\phi^i) \left((1 + \gamma)f - \frac{1}{a\tau}(\phi^i - \phi^k) \right).$$

3.2.2 Positivity

Later, when the equation is also discretized in space, we only have to deal with finite-dimensional spaces, where every positive bilinear form is elliptic. So from a numerical point of view, an important question is under which restrictions on the time-step size the operator a_ϕ is positive definite. We first look at two extreme situations:

- On the one hand, setting $\gamma = 0$ in Proposition 3.3 yields

$$a_\phi(\xi, \xi) \geq \left(\frac{1}{\tau} - \frac{\beta^2}{4\epsilon^3} \right) \int_\Omega |\xi|^2.$$

Thus, a_ϕ is positive definite if

$$\tau < \frac{4}{\beta^2} \epsilon^3. \quad (3.12)$$

Of course this is only a sufficient condition and is much too restrictive in most cases. However, the constraint can be sharp in some cases. For instance, $G''(\phi) \equiv -\beta$ is such a case, see the numerical example in Section A.7. Note that for the potential in the deep-quench equation, we have $G''(\phi) \equiv -15$ for any ϕ .

- On the other hand, if ϕ^* is a (local) minimizer of the energy E , the Hessian of E is positive semi-definite and so is the conjugate operator

$$-\nabla(\delta^2 E(\phi^*)/\delta\phi^2)\nabla^t.$$

Having the additional L^2 -term, a_{ϕ^*} is positive definite for every $\tau > 0$.

As always, the truth lies in-between the extreme cases. In typical applications we will have, at least after a short time, developed interfaces (see Section 2.6.2). Then, and if the potential G is convex at the bulk equilibrium values, we get a less restrictive constraint on τ . To show this, we quote two theorems from Chen [1994]. We will not state here the full set of assumptions, but refer to (1.9)–(1.15) in Chen's paper. Essentially, a ϕ_ϵ is defined that has the form described in Section 2.6.2. Also, G should be a double-well potential. Then, the Hessian of the energy is bounded from below.

Theorem 3.6 (Chen [1994], Theorem 2.3). *Assume that the above conditions hold. Then there exists $C_1 > 0$ such that for every $\epsilon \in (0, 1]$ and every $\psi \in H^1(\Omega)$*

$$\int_\Omega \epsilon |\nabla\psi|^2 + \epsilon^{-1} G''(\phi_\epsilon)\psi^2 \geq -C_1 \epsilon \int_\Omega \psi^2. \quad (3.13)$$

Theorem 3.7 (Chen [1994], Theorem 3.1). *There exist $\epsilon_* > 0$ and $C_2 > 0$ such that if $\epsilon \in (0, \epsilon_*]$ and $\psi \in H^1(\Omega)$ satisfy*

$$\int_\Omega \epsilon |\nabla\psi|^2 + \epsilon^{-1} G''(\phi_\epsilon)\psi^2 \leq 0,$$

then

$$-C_1 \epsilon \int_\Omega \psi^2 \geq -C_2 \int_\Omega |\nabla w|^2, \quad (3.14)$$

where w is some function with $\Delta w = \psi$.

Note that the right-hand side of (3.14) is the H^{-1} -norm of ψ . With these theorems we can compute the time-step restriction when we are in a regime of developed interfaces.

Lemma 3.8. *Assume ϕ_ε is in a state of developed interfaces. Then a_{ϕ_ε} is positive definite if*

$$\tau \leq C, \quad (3.15)$$

where C is independent of ε .

Proof.

$$a_{\phi_\varepsilon}(\xi, \xi) = \frac{1}{\tau} \int_{\Omega} \frac{1}{M(\phi_\varepsilon)} |\xi|^2 + \varepsilon \int_{\Omega} |\nabla \nabla \cdot \xi|^2 + \frac{1}{\varepsilon} \int_{\Omega} G''(\phi_\varepsilon) |\nabla \cdot \xi|^2$$

and by setting $\psi = \nabla \cdot \xi$

$$\geq \frac{1}{\tau} \int_{\Omega} |\xi|^2 + \int_{\Omega} \left(\varepsilon |\nabla \psi|^2 + \varepsilon^{-1} G''(\phi_\varepsilon) \psi^2 \right)$$

If the second integral is positive, there is no restriction on τ . If not,

$$\begin{aligned} &\stackrel{(3.13)}{\geq} \frac{1}{\tau} \int_{\Omega} |\xi|^2 - C_1 \varepsilon \int_{\Omega} \psi^2. \\ &\stackrel{(3.14)}{\geq} \frac{1}{\tau} \int_{\Omega} |\xi|^2 - C_2 \|\psi\|_{H^{-1}(\Omega)}^2 \\ &= \frac{1}{\tau} \|\xi\|_{L^2(\Omega)}^2 - C_2 \|\nabla \cdot \xi\|_{H^{-1}(\Omega)}^2 \\ &\geq \left(\frac{1}{\tau} - C_2 \right) \|\xi\|_{L^2(\Omega)}^2. \end{aligned}$$

This proves the assertion. For the last inequality, we have used

$$\|\nabla \cdot \xi\|_{H^{-1}(\Omega)} \leq \|\xi\|_{L^2(\Omega)}.$$

To see this, let w be the solution of

$$-\Delta w = \nabla \cdot \xi \quad \text{in } \Omega \quad \text{and} \quad \nabla w \cdot \nu = 0 \quad \text{on } \partial\Omega.$$

Then $\|\nabla \cdot \xi\|_{H^{-1}(\Omega)} = \|\nabla w\|_{L^2(\Omega)}$ by definition of the norm and

$$\|\nabla w\|_{L^2(\Omega)}^2 = \int_{\Omega} |\nabla w|^2 = \int_{\Omega} (-\Delta w) w = \int_{\Omega} (\nabla \cdot \xi) w = - \int_{\Omega} \xi \cdot \nabla w \leq \|\xi\|_{L^2(\Omega)} \|\nabla w\|_{L^2(\Omega)}$$

by Hölder. □

3.3 Time-step schemes

The discretization we performed in the previous sections was independent of the concrete choice of a time-step scheme, so any scheme of the form

$$\frac{1}{a\tau}(\phi^{k+1} - \phi^k) = F(\phi^{k+1}) + \gamma F(\phi^k)$$

can be used. As we want to allow large time steps, especially in the late stages of our simulation, a second-order scheme is preferable.

As was pointed out by Weikard [2002], it is not enough for Cahn–Hilliard-type equations to use A-stable schemes. Indeed, we experienced too that perturbations caused by the initial data were only poorly damped when using Crank–Nicolson. Weikard suggested a scheme by Bristeau et al. [1987], which did work well for our purposes. However, we found that another scheme, namely TR-BDF2, is better suited for our equations while requiring less computational work.

All time-step schemes we considered, together with their stability properties, are presented in Appendix B. The parameters a and γ to be used in Equation (3.7) can be found in Table B.1 on page 163. In the following table we summarize some important properties.

	Euler	CN	PR	BGP	TR-BDF2
Order of accuracy	1	2	2	2	2
Steps per time-step	1	1	2	3	2
Stability (A/A+/L)	L	A	A	A+	L

We touch upon stability concepts below, for a proper introduction see e.g. Deuffhard and Bornemann [1994, Chapter 6].

Stability concepts are used to answer the question for which time steps τ the discrete evolution inherits stability properties of the continuous evolution. This can be made explicit for the model problem

$$\partial_t x = \mathbf{A}x, \tag{3.16}$$

where \mathbf{A} is a linear operator, and the corresponding discrete evolution

$$x^{n+1} = R(\tau\mathbf{A})x^n. \tag{3.17}$$

where the rational function R is given by the time-step scheme.

A time-step scheme is called A-stable if $|R(z)| \leq 1$ for all $z \in \mathbb{C}$ with non-positive real part. Then, given a stable evolution (3.16), the discrete evolution (3.17) is also stable for all $\tau > 0$. We denote a scheme strongly A-stable (A+) if $|R(z)|$ is bounded away from one. This is necessary for Cahn–Hilliard-type equations. Finally, a scheme is called L-stable if $R(\infty) = 0$, thus providing better control of large time steps.

3.4 Adaptivity

As we have seen in Section 2.6, the solution of Cahn–Hilliard-type equations develops on different time scales. To capture both fast and slow changes, we need an adaptive time-step size.

We first derive an error estimator for the TR-BDF2 scheme and then argue that we can use a quite naïve estimator for the other schemes. To derive the error estimator we use that TR-BDF2 contains an embedded third-order scheme. By comparison of the solutions using the second- and third-order schemes, we get an effective and robust error estimator. Here, embedded means that it is not necessary to solve the whole system again to obtain the third-order solution. Details can be found in Section A.4.1.

For the other time-step schemes, we simply use the difference in the L^2 -norm of the solution at two successive time steps as an error indicator. Numerical simulations have shown that this yields comparable results, see Section A.4.3.

To set the time-step sizes we use an explicit strategy in the sense that we first compute the solution for the time-step and if the error was too large, we repeat the time-step. The value from which on we repeat the time-step is an empirical value. Additionally, an upper limit for the step size is set.

An example of time-step sizes generated with the error estimator for a coarsening problem (see Chapter 5) is shown in Figure 3.1.

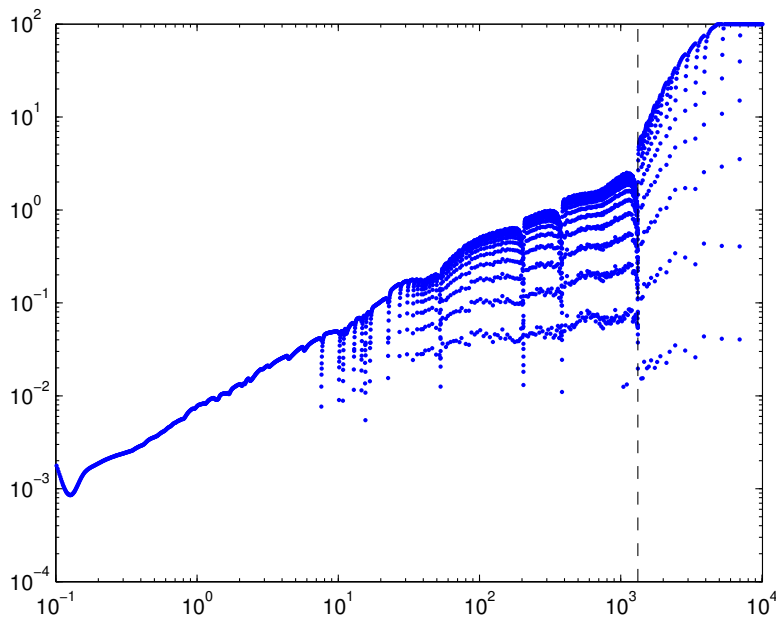


Figure 3.1: Example for the development of the time-step sizes. The simulation was for spinodal decomposition. At the time marked with the dashed line, only one island is left. Before that, the step sizes approximately grow as $\tau \sim t^{0.8}$. Note that they vary across five orders of magnitude.

Spatial Discretization

In this chapter, we spatially discretize the time-discrete equations derived in the previous chapter using a finite-element method.

As a first test for our time discretization, we used a finite difference scheme for the spatial discretization, see Otto, Penzler, and Rump [2005]. The guiding principles were conservation of mass and thermodynamic consistency of the discretization. The discretization was based on a finite-volume-type Ansatz, where functions live on the volumes and the components of the flux live on the edges of the mesh. The mesh was non-adaptive and equidistant, which made the computations for large domains very expensive.

As we have seen in Section 2.6, solutions of Cahn–Hilliard-type equations have a special form, namely essentially large and flat terraces with a thin and steep transition in-between. Therefore we can save a lot of computational work by reducing the number of unknowns on the terraces while still resolving the steps. To realize such an adaptive mesh, we use a finite-element method. The strategy for the adaptive refinement is presented in Section 4.9.

The energy used from now on for the rest of this work is a Ginzburg–Landau-type energy as introduced in Section 2.6:

$$E(\phi) = \int_{\Omega} \frac{\varepsilon}{2} |\nabla \phi|^2 + \varepsilon^{-1} G(\phi), \quad (4.1)$$

with a potential G depending on the actual application.

The idea we had in mind when developing the discretization is presented in Section 4.1. However, since this approach comes neither from a minimization problem nor from a saddle point problem, the standard methods for error estimation cannot be used. Thus, we use another approach in form of a non-conforming discretization of the minimization problem (3.11), see Section 4.4. The resulting fully discrete equations are equivalent to those derived in Section 4.1. Now we can easily get error estimates, see Section 4.6.

It turned out that the solution of the fully discrete equations is by no means straightforward. In Section 4.8, we present a number of possibilities to solve the equations. The different methods are compared in Section 4.11.

4.1 Idea behind the discretization

We have seen in Section 3.2 that the problem to be solved in every Newton-step is: Given $\phi^i, \phi^k \in H^1(\Omega)$, find $J_*^{i+1} \in H(\nabla\nabla\cdot, \Omega)$ such that

$$\int_{\Omega} \frac{1}{a\tau M(\phi^i)} J_*^{i+1} \cdot \tilde{J} + \varepsilon \int_{\Omega} (\nabla\nabla\cdot J_*^{i+1}) \cdot (\nabla\nabla\cdot \tilde{J}) + \frac{1}{\varepsilon} \int_{\Omega} G''(\phi^i)(\nabla\cdot J_*^{i+1})(\nabla\cdot \tilde{J}) = \int_{\Omega} r(\nabla\cdot \tilde{J}) \quad (4.2a)$$

for all $\tilde{J} \in H(\nabla\nabla\cdot)$, where $r \in H^{-1}(\Omega)$ as defined in (3.8). Then, calculate $\phi^{i+1} \in H^1(\Omega)$ via

$$\phi^{i+1} = \phi^k + \tau_a (-\nabla\cdot J_*^{i+1} + (1+\gamma)f). \quad (4.2b)$$

If we now require more regularity for ϕ , namely $\phi^i, \phi^k \in H^2(\Omega)$, we get $r \in L^2(\Omega)$ and so by weak regularity theory $\nabla\nabla\cdot J_*^{i+1}$ is not only in $L^2(\Omega)^2$, but in $H(\nabla\cdot, \Omega)$, which is defined as follows.

Definition 4.1. The space $H(\nabla\cdot, \Omega)$ is defined as

$$H(\nabla\cdot, \Omega) = \{J \in L^2(\Omega)^2 \mid \nabla\cdot J \in L^2(\Omega)\}$$

and is equipped with the norm

$$\|J\|_{H(\nabla\cdot, \Omega)}^2 = \|J\|_{L^2(\Omega)}^2 + \|\nabla\cdot J\|_{L^2(\Omega)}^2.$$

◇

Therefore, we can apply partial integration in the fourth-order term of the left-hand side of (4.2a) and get

$$\int_{\Omega} \frac{1}{a\tau M(\phi^i)} J_*^{i+1} \cdot \tilde{J} - \varepsilon \int_{\Omega} (\Delta\nabla\cdot J_*^{i+1})(\nabla\cdot \tilde{J}) + \frac{1}{\varepsilon} \int_{\Omega} G''(\phi^i)(\nabla\cdot J_*^{i+1})(\nabla\cdot \tilde{J}) = \int_{\Omega} r(\nabla\cdot \tilde{J}) \quad (4.3)$$

If we now set

$$K := -\nabla\nabla\cdot J_*^{i+1},$$

then (K, J_*^{i+1}) is a solution of the problem:

Given $\phi^i, \phi^k \in H^2(\Omega)$, find $(K, J_*^{i+1}) \in H(\nabla\cdot, \Omega) \times H(\nabla\cdot, \Omega)$ such that

$$\int_{\Omega} \frac{1}{a\tau M(\phi^i)} J_*^{i+1} \cdot \tilde{J} + \varepsilon \int_{\Omega} (\nabla\cdot K)(\nabla\cdot \tilde{J}) + \varepsilon^{-1} \int_{\Omega} G''(\phi^i)(\nabla\cdot J_*^{i+1})(\nabla\cdot \tilde{J}) = \int_{\Omega} r(\nabla\cdot \tilde{J}) \quad (4.4a)$$

$$\int_{\Omega} K \cdot \tilde{K} = \int_{\Omega} (\nabla\cdot J_*^{i+1})(\nabla\cdot \tilde{K}), \quad (4.4b)$$

for all $\tilde{J} \in H(\nabla\cdot, \Omega)$, $\tilde{K} \in H(\nabla\cdot, \Omega)$. Then, find $\phi^{k+1} \in L^2(\Omega)$ such that

$$\int_{\Omega} \frac{1}{a\tau} (\phi^{i+1} - \phi^k) \zeta + \int_{\Omega} (\nabla\cdot J_*^{i+1}) \zeta = (1+\gamma) \int_{\Omega} f \zeta \quad \forall \zeta \in L^2(\Omega). \quad (4.4c)$$

By (4.4b), we have that $\nabla\nabla\cdot J_*^{i+1} \in H(\nabla\cdot, \Omega)$ and thus $\nabla\cdot J_*^{i+1} \in H^2(\Omega)$. Provided $f \in H^2(\Omega)$, we have $\phi^{i+1} \in H^2(\Omega)$ and so we can iterate.

Note that it is enough for the vector fields K and J_*^{i+1} to be in $H(\nabla\cdot, \Omega)$.

Proposition 4.2. *The problems (4.2a) and (4.4) are equivalent.*

Proof. It remains to show that a solution of (4.4) provides a solution of (4.2a). Indeed, if $(K, J) \in H(\nabla\cdot) \times H(\nabla\cdot)$ is a solution of (4.4), then Equation (4.4b) tells us that $J \in H(\nabla\nabla\cdot)$ with the gradient given by $K = -\nabla\nabla\cdot J$. When we now restrict \tilde{J} to $H(\nabla\nabla\cdot)$, we can apply partial integration in the second term of (4.4b) and get

$$\int_{\Omega} (\nabla\cdot K)(\nabla\cdot \tilde{J}) = - \int_{\Omega} K \cdot \nabla\nabla\cdot \tilde{J} = \int_{\Omega} \nabla\nabla\cdot J \cdot \nabla\nabla\cdot \tilde{J},$$

so J is a solution of (4.2a). □

Together with Proposition 3.5, we get

Corollary 4.3. *For τ small enough, Problem (4.4) has a unique solution.*

Remark 4.4. Problem (4.4) may look like a saddle point problem, but, although it is indeed indefinite, it is *not* a saddle point problem in J and K . To see this, write

$$\mathcal{L}(\tilde{J}, \tilde{K}) := \frac{1}{2}a(\tilde{J}, \tilde{J}) + b(\tilde{J}, \tilde{K}) - \frac{1}{2}c(\tilde{K}, \tilde{K}) - f(\tilde{J}),$$

where

$$\begin{aligned} a(J, \tilde{J}) &= \int_{\Omega} \frac{1}{\tau M(\phi^i)} J \cdot \tilde{J} + \varepsilon^{-1} \int_{\Omega} G''(\phi^i)(\nabla\cdot J)(\nabla\cdot \tilde{J}), \\ b(\tilde{J}, \tilde{K}) &= \varepsilon \int_{\Omega} (\nabla\cdot \tilde{J})(\nabla\cdot \tilde{K}), \quad c(K, \tilde{K}) = \varepsilon \int_{\Omega} K \cdot \tilde{K}, \quad f(\tilde{J}) = \int_{\Omega} r(\nabla\cdot \tilde{J}). \end{aligned}$$

Every critical point of \mathcal{L} is a solution of (4.4), which in this notation is

$$\begin{aligned} a(J, \tilde{J}) + b(K, \tilde{J}) &= f(\tilde{J}) & \forall \tilde{J} \in H(\nabla\cdot, \Omega), \\ b(J, \tilde{K}) - c(K, \tilde{K}) &= 0 & \forall \tilde{K} \in H(\nabla\cdot, \Omega). \end{aligned}$$

Now let (J, K) be such a solution. The saddle point property we have to check is

$$\mathcal{L}(J, \tilde{K}) \leq \mathcal{L}(J, K) \leq \mathcal{L}(\tilde{J}, K) \quad \forall \tilde{J}, \tilde{K} \in H(\nabla\cdot).$$

The first inequality is indeed true

$$\textcircled{1} \quad \mathcal{L}(J, \tilde{K}) \leq \mathcal{L}(J, K) \quad \forall \tilde{K} \in H(\nabla\cdot)$$

$$\begin{aligned} \frac{1}{2}a(J, J) + \underbrace{b(J, \tilde{K})}_{=c(K, \tilde{K})} - \frac{1}{2}c(\tilde{K}, \tilde{K}) - f(J) &\leq \frac{1}{2}a(J, J) + \underbrace{b(J, K)}_{=c(K, K)} - \frac{1}{2}c(K, K) - f(J) \\ \iff c(K, \tilde{K}) - \frac{1}{2}c(\tilde{K}, \tilde{K}) &\leq c(K, K) - \frac{1}{2}c(K, K) \\ \iff 0 &\leq c(K, K) - 2c(K, \tilde{K}) + c(\tilde{K}, \tilde{K}) \\ \iff 0 &\leq c(K - \tilde{K}, K - \tilde{K}), \end{aligned}$$

which is true due to the positivity of c . However, the second inequality is wrong

$$\textcircled{2} \quad \mathcal{L}(J, K) \not\leq \mathcal{L}(\tilde{J}, K) \quad \forall \tilde{J} \in H(\nabla \cdot)$$

$$\begin{aligned} \frac{1}{2}a(J, J) + \underbrace{b(J, K) - f(J)}_{=-a(J, J)} - \frac{1}{2}c(K, K) &\leq \frac{1}{2}a(\tilde{J}, \tilde{J}) + \underbrace{b(\tilde{J}, K) - f(\tilde{J})}_{=-a(J, \tilde{J})} - \frac{1}{2}c(K, K) \\ \iff -\frac{1}{2}a(J, J) &\leq \frac{1}{2}a(\tilde{J}, \tilde{J}) - a(J, \tilde{J}) \\ \iff 0 &\leq a(J, J) - 2a(J, \tilde{J}) + a(\tilde{J}, \tilde{J}) \\ \iff 0 &\leq a(J - \tilde{J}, J - \tilde{J}). \end{aligned}$$

Since $G''(\phi)$ might be negative and we don't have the fourth-order term to compensate, $a(J - \tilde{J}, J - \tilde{J})$ might be either positive or negative.

◇

As a consequence, we cannot use this idea to get existence results or error estimates. We therefore use another approach for spatial discretization, see Section 4.5, which leads to the same discrete system. Nevertheless, we will use finite-dimensional subspaces of $H(\nabla \cdot)$ for the discretization and one should have in mind the splitting (4.4).

Finite-dimensional subspaces

To discretize Equation (4.4), we use the following finite-dimensional subspaces of $L^2(\Omega)$ and $H(\nabla \cdot, \Omega)$:

- $\mathcal{L}_0(\mathcal{T}_h) = \{\text{piecewise constant functions}\} \subset L^2(\Omega)$ for the functions and
- $\mathcal{RT}_0(\mathcal{E}_h) = \{\text{lowest-order Raviart-Thomas elements}\} \subset H(\nabla \cdot, \Omega)$ for the vector fields.

Before we define the finite-dimensional subspaces and their properties, let us first fix some notation. Assume that $\Omega \subset \mathbb{R}^2$ is a polygonal domain and consider a triangulation of Ω consisting of

- N_T triangles $\{T_0, \dots, T_{N_T-1}\} =: \mathcal{T}_h$ with volumes $|T_k|$ and outer normals ν_{T_k} ,
- N_E edges $\{E_0, \dots, E_{N_E-1}\} =: \mathcal{E}_h$ with lengths $|E_i|$ and normals ν_i ,
- N_V vertices.

In the case of Neumann boundary conditions (BC2), we distinguish between boundary edges ($E \subset \partial\Omega$) and inner edges (all others). The number of inner edges is denoted by N_E^0 . In the case of periodic boundary conditions, we consider all edges to be inner edges, and so $N_E = N_E^0$.

We will use the following notation in this chapter: $J(x)$ is a continuous vector field, $J_h(x)$ is a vector field in the (mesh-dependent) finite-dimensional subspace and \underline{J} is the coefficient vector for J_h with respect to a basis.

Furthermore, we use the indices i, j for edge-related quantities and k, l for triangle-related quantities where possible.

Finally, we use L^2 -orthogonal projections $P_h^T : L^2(\Omega) \rightarrow \mathcal{L}_0(\mathcal{T}_h)$ and $P_h^E : H(\nabla \cdot, \Omega) \rightarrow \mathcal{RT}_0(\mathcal{E}_h)$.

4.2 Piecewise constant finite elements

As finite-dimensional subspace of $L^2(\Omega)$, we use the space of lowest possible order: the space of functions being constant on each triangle

$$\mathcal{L}_0(\mathcal{T}_h) := \{\phi_h \in L^2(\Omega) \mid \phi_h|_T = \text{const} \quad \forall T \in \mathcal{T}_h\}.$$

The simplest basis is given by functions which are one on Triangle T_k and zero everywhere else:

$$Z_k(x) := \begin{cases} 1 & \text{for } x \in T_k \\ 0 & \text{else} \end{cases} \quad k = 0, \dots, N_T - 1. \quad (4.5)$$

It remains to define a restriction operator $L^2(\Omega) \rightarrow \mathcal{L}_0(\mathcal{T}_h)$. The essential property of this operator is to conserve mass.

Definition 4.5. Let $u \in L^2(\Omega)$. The restriction operator $R_h^T : L^2(\Omega) \rightarrow \mathcal{L}_0(\mathcal{T}_h)$ is given by

$$R_h^T(u) := \sum_{k=0}^{N_T-1} \left(\int_{T_k} u \right) Z_k \quad \iff \quad R_h^T(u)|_T := \int_T u. \quad (4.6)$$

◇

Lemma 4.6 (Properties of R_h^T). For any $u \in L^2(\Omega)$ we have for all $T \in \mathcal{T}_h$

$$\int_T R_h^T(u) = \int_T u.$$

Furthermore, R_h^T is the L^2 -orthogonal projection of u onto $\mathcal{L}_0(\mathcal{T}_h)$,

$$R_h^T(u) = P_h^T(u).$$

Proof. The first assertion is true by definition. For the second one define $\tilde{Z}_k(x) := |T_k|^{-1/2} Z_k$. Then the \tilde{Z}_k form an L^2 -orthonormal basis and

$$R_h^T(u) = \sum_{k=0}^{N_T-1} \left(\int_T u \right) Z_k = \sum_{k=0}^{N_T-1} \left(\int_\Omega u \tilde{Z}_k \right) \tilde{Z}_k = P_h^T(u).$$

□

4.3 Raviart–Thomas finite elements

As subspace for $H(\nabla \cdot, \Omega)$, we use the lowest-order Raviart–Thomas (RT) finite elements, see Raviart and Thomas [1977], which compose a conformal subspace of $H(\nabla \cdot, \Omega)$. On each triangle $T \in \mathcal{T}_h$, they are of the form

$$J_h(x)|_T = ax + b, \quad (4.7)$$

where $b \in \mathbb{R}^2$ and $a \in \mathbb{R}$. We will need such vector fields later, so we give the set a name

$$\mathcal{RT}_{-1}(\mathcal{T}_h) := \left\{ J_h \in L^2(\Omega)^2 \mid J_h(x)|_T = ax + b \quad \forall T \in \mathcal{T}_h \right\}. \quad (4.8)$$

Note that a is a scalar, so the $\mathcal{RT}_{-1}(\mathcal{T}_h)$ finite elements are only a subset of the affine vector fields. Next, we state some basic properties of these vector fields:

Proposition 4.7. *Let J_h be in $\mathcal{RT}_{-1}(\mathcal{T}_h)$, then*

1. *On each triangle $T \in \mathcal{T}_h$ we have*

$$\nabla \cdot J_h|_T = \text{const} =: \omega|_T. \quad (4.9)$$

2. *Given a straight line $g := \{x \in \mathbb{R}^2 \mid x \cdot \nu = \text{const}\}$ with normal ν , we have for all $x \in g \cap T$*

$$J_h(x) \cdot \nu = \text{const}.$$

3. *J_h is in $H(\nabla \cdot, \Omega)$ if and only if*

$$\sum_{k=0}^{N_T-1} \int_{\partial T_k} \eta(J_h \cdot \nu_{T_k}) = 0 \quad \forall \eta \in C^\infty(\Omega),$$

i.e. if $J_h \cdot \nu$ is continuous across edges.

Proof. The first two assertions are obvious. For the third one, take a function $\eta \in C^\infty(\Omega)$. Then

$$\begin{aligned} \int_{\Omega} \eta \omega &= \sum_{k=0}^{N_T-1} \int_{T_k} \eta(\nabla \cdot J_h) = \sum_{k=0}^{N_T-1} \left(- \int_{T_k} \nabla \eta \cdot J_h + \int_{\partial T_k} \eta(J_h \cdot \nu_{T_k}) \right) \\ &= - \int_{\Omega} \nabla \eta \cdot J_h + \sum_{k=0}^{N_T-1} \int_{\partial T_k} \eta(J_h \cdot \nu_{T_k}). \end{aligned} \quad (4.10)$$

So if the normal component of J_h is continuous across the edges, then $\omega \in \mathcal{L}_0(\mathcal{T}_h) \subset L^2(\Omega)$ is the weak divergence of J_h and so $J_h \in H(\nabla \cdot, \Omega)$. \square

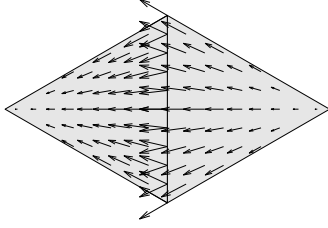


Figure 4.1: Basis element of the space $\mathcal{RT}_0(\mathcal{E}_h)$. The normal component is continuous across the centre edge and zero on the outer edges.

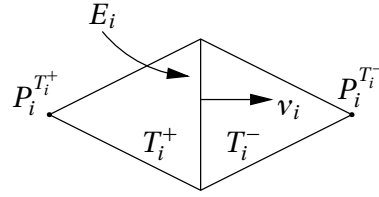


Figure 4.2: Notations for triangles and edges. $\mu_i^T \in \{\pm 1\}$ is defined such that $\mu_i^T v_i$ is the outer normal to T .

Note that we took $\eta \in C^\infty(\Omega)$ and not $\eta \in C_0^\infty(\Omega)$ in (4.10). This ensures $J_h \cdot \nu = 0$ on $\partial\Omega$ in the case of non-periodic boundary conditions.

Now we define the space of the lowest-order Raviart–Thomas finite elements by

$$\mathcal{RT}_0(\mathcal{E}_h) := \left\{ J_h \in \mathcal{RT}_{-1}(\mathcal{T}_h) \mid \sum_{T \in \mathcal{T}_h} \int_{\partial T} \eta (J_h \cdot \nu_T) = 0 \quad \forall \eta \in C^\infty(\Omega) \right\} \quad (4.11)$$

In order to define a basis of $\mathcal{RT}_0(\mathcal{E}_h)$, we notice that prescribing the values of $J_h \cdot \nu$ on the (inner) edges \mathcal{E}_h defines a unique vector field in $\mathcal{RT}_0(\mathcal{E}_h)$. Indeed, this makes J_h continuous across the (inner) edges and as we have three degrees of freedom on each triangle, the vector field is unique. This is also the reason why we write $\mathcal{RT}_0(\mathcal{E}_h)$ and not $\mathcal{RT}_0(\mathcal{T}_h)$.

Now we seek a basis $\{\Psi_i\}_{i=0, \dots, N_E^0 - 1}$ with $\Psi_i(x) \cdot \nu_j = \delta_{ij}$ for $x \in E_j$. Such a basis is given by

$$\Psi_i(x) := \begin{cases} \mu_i^T \frac{|E_i|}{2|T_i^\pm|} (x - P_i^{T_i^\pm}) & x \in T_i^\pm, \\ 0 & \text{else,} \end{cases} \quad i = 0, \dots, N_E - 1, \quad (4.12)$$

see Figure 4.1 for the plot of a basis element. P_i^T is the vertex of triangle T which does not belong to E_i and $\mu_i^T \in \{\pm 1\}$ is a sign such that $\mu_i^T v_i$ is the outer normal to T . By T_i^\pm we denote the two triangles adjacent to E_i with the sign corresponding to μ_i^T , see Figure 4.2.

Proposition 4.8. *For the basis vector fields $\Psi_i(x)$, we have*

$$\Psi_i(x) \cdot \nu_j = \delta_{ij} \quad \text{for } x \in E_j.$$

Proof. ① If E_j is not adjacent to T_i^\pm , then $\Psi_i(x) \equiv 0$ on E_j .

② If E_j is adjacent to T_i^\pm and $i \neq j$, then $x \in E_j$ and $P_i^{T_i^\pm}$ lie both on the edge E_j and therefore $x - P_i^{T_i^\pm}$ is orthogonal to ν_j , which in turn gives $\Psi_i(x) \cdot \nu_j = 0$.

③ For $i = j$ and $T \in T_i^\pm$ we have, since $\Psi_i(x) \cdot \nu_i = \text{const}$ on E_i by Proposition 4.7,

$$\begin{aligned} \Psi_i(x) \cdot \nu_i &= \frac{1}{|E_i|} \int_{E_i} \Psi_i(x) \cdot \nu_i \stackrel{\textcircled{2}}{=} \frac{1}{|E_i|} \sum_{j \in \mathcal{E}(T)} \int_{E_j} \Psi_i(x) \cdot \nu_j \\ &= \frac{1}{|E_i|} \int_{\partial T} \mu_i^T \Psi_i(x) \cdot \nu_T(x) = \frac{1}{|E_i|} \int_T \mu_i^T \nabla \cdot \Psi_i(x) \\ &= \frac{1}{|E_i|} \int_T \frac{|E_i|}{|T|} = 1. \end{aligned}$$

□

What is left is the definition of a restriction operator. The essential property here is to retain the divergence.

Definition 4.9. The restriction operator $R_h^E : H(\nabla \cdot, \Omega) \cap L^s(\Omega)^2 \rightarrow \mathcal{RT}_0(\mathcal{E}_h)$, $s > 2$ fixed, is given by

$$R_h^E(J) = \sum_{j=0}^{N_E^0-1} J_j \Psi_j(x) \quad \text{with} \quad J_j := \frac{1}{|E_j|} \int_{E_j} J \cdot \nu_j.$$

◇

The higher regularity for the vector fields is necessary since the trace of a vector field in $H(\nabla \cdot, T)$ need not to be in $H^{1/2}(\partial T)$. In abuse of notation, we will nevertheless write sometimes $H(\nabla \cdot, \Omega)$.

Lemma 4.10 (Properties of R_h^E). *For any $J \in H(\nabla \cdot, \Omega)$ and $T_k \in \mathcal{T}_h$, the restriction operator and the divergence commute in the sense*

$$\nabla \cdot R_h^E(J) = R_h^T(\nabla \cdot J).$$

Furthermore, the divergence on each triangle is conserved

$$\int_{T_k} \nabla \cdot J = \int_{T_k} \nabla \cdot R_h^E(J)$$

Proof. For the first assertion, write

$$\begin{aligned} R_h^T(\nabla \cdot J)|_T &= \frac{1}{|T|} \int_T \nabla \cdot J = \frac{1}{|T|} \int_{\partial T} J \cdot \nu_T = \frac{1}{|T|} \sum_{j \in \mathcal{E}_h(T)} \mu_j^T \int_{E_j} J \cdot \nu_j \\ &= \sum_{j \in \mathcal{E}_h(T)} \mu_j^T \frac{|E_j|}{|T|} \frac{1}{|E_j|} \int_{E_j} J \cdot \nu_j = \sum_{j \in \mathcal{E}_h(T)} (\nabla \cdot \Psi_j) J_j = \nabla \cdot \left(\sum_{j=0}^{N_E-1} J_j \Psi_j|_T \right) \\ &= \nabla \cdot R_h^E(J)|_T. \end{aligned}$$

The second assertion follows from the first with Lemma 4.6. □

Concerning the approximation estimates for the restriction operator, we quote the following Proposition from Brezzi and Fortin [1991, Prop. III.3.9]

Proposition 4.11. *There exists a constant c independent of h such that*

$$\|J - R_h^E(J)\|_{L^2(\Omega)} \leq ch |J|_{H^1(\Omega)}.$$

Moreover,

$$\|\nabla \cdot (J - R_h^E(J))\|_{L^2(\Omega)} \leq ch |\nabla \cdot J|_{H^1(\Omega)}.$$

To express this in a more concise form, we generalize Definition 4.1.

Definition 4.12 (H^m -div). The spaces $H^m(\nabla \cdot, \Omega)$, $m \geq 0$, are defined as

$$H^m(\nabla \cdot, \Omega) = \{J \in H^m(\Omega)^2 \mid \nabla \cdot J \in H^m(\Omega)\}$$

and are equipped with the norm

$$\|J\|_{H^m(\nabla \cdot, \Omega)}^2 = \|J\|_{H^m(\Omega)}^2 + \|\nabla \cdot J\|_{H^m(\Omega)}^2.$$

We furthermore define the semi-norm

$$|J|_{H^m(\nabla \cdot, \Omega)}^2 = |J|_{H^m(\Omega)}^2 + |\nabla \cdot J|_{H^m(\Omega)}^2.$$

For $m = 0$, that are the $L^2(\Omega)^2$ vector fields with divergence in $L^2(\Omega)$, we omit the superscript and write $H(\nabla \cdot, \Omega)$ for $H^0(\nabla \cdot, \Omega)$. \diamond

Now Proposition 4.11 yields

$$\|J - R_h^E(J)\|_{H(\nabla \cdot, \Omega)} \leq ch |J|_{H^1(\nabla \cdot, \Omega)}. \quad (4.13)$$

4.4 A discrete gradient

As motivated in Section 4.1, we want to use a finite-dimensional subspace of $H(\nabla \cdot)$ for the discretization. If J_*^{i+1} from Equation (4.2a) is in $H(\nabla \cdot)$, then by definition $\nabla \cdot J_*^{i+1}$ is in $L^2(\Omega)$. The discrete flux then would be in $\mathcal{RT}_0(\mathcal{E}_h)$ and its divergence in $\mathcal{L}_0(\mathcal{T}_h)$. To make sense of the fourth-order term

$$\int_{\Omega} \nabla \nabla \cdot J_*^{i+1} \cdot \nabla \nabla \cdot \tilde{J},$$

we have to define a discrete gradient ∇_h for piecewise constant functions. It helps us later and requires no extra effort to define the discrete gradient for all L^2 -functions.

Definition 4.13 (Discrete gradient). Let $u \in L^2(\Omega)$. The *discrete gradient* $\nabla_h u \in \mathcal{RT}_0(\mathcal{E}_h)$ is defined by duality

$$\int_{\Omega} \nabla_h u \cdot \tilde{K}_h = - \int_{\Omega} u (\nabla \cdot \tilde{K}_h) \quad \forall \tilde{K}_h \in \mathcal{RT}_0(\mathcal{E}_h). \quad (4.14)$$

\diamond

By Riesz, there exists a unique discrete gradient for all $u \in L^2(\Omega)$.

Proposition 4.14 (Properties of the discrete gradient).

(G.i) For any u_h that is piecewise constant on the triangles, $u_h \in \mathcal{L}_0(\mathcal{T}_h)$, we have

$$\int_{\Omega} \nabla_h u_h \cdot R_h^E(\tilde{K}) = - \int_{\Omega} u_h (\nabla \cdot \tilde{K}) \quad \forall \tilde{K} \in H(\nabla \cdot).$$

(G.ii) Let $\{e_i\}_{i=0, \dots, N_E-1}$ be an L^2 -orthonormal basis of $\mathcal{RT}_0(\mathcal{E}_h)$. Then the discrete gradient can be written as

$$\nabla_h u = \sum_{i=0}^{N_E-1} \mu_i e_i \quad \text{with} \quad \mu_i = - \int_{\Omega} u (\nabla \cdot e_i).$$

(G.iii) If $u \in H^1(\Omega)$, then $\nabla_h u$ is the L^2 -orthogonal projection of ∇u onto $\mathcal{RT}_0(\mathcal{E}_h)$

$$\nabla_h u = P_h^E(\nabla u).$$

(G.iv) For any $u \in L^2(\Omega)$, we have

$$\nabla_h R_h^T(u) = \nabla_h u.$$

Proof. To see (G.i), write

$$- \int_{\Omega} (\nabla \cdot \tilde{K}) u_h = - \sum_{k=0}^{N_T-1} \int_{T_k} (\nabla \cdot \tilde{K}) u_h = - \sum_{k=0}^{N_T-1} \underline{u}_k \int_{T_k} (\nabla \cdot \tilde{K}).$$

By Lemma 4.10 this is

$$\begin{aligned} &= - \sum_{k=0}^{N_T-1} \underline{u}_k \int_{T_k} (\nabla \cdot R_h^E(\tilde{K})) = - \sum_{k=0}^{N_T-1} \int_{T_k} (\nabla \cdot R_h^E(\tilde{K})) u_h \\ &= - \int_{\Omega} (\nabla \cdot R_h^E(\tilde{K})) u_h \stackrel{(4.14)}{=} \int_{\Omega} \nabla_h u_h \cdot R_h^E(\tilde{K}). \end{aligned}$$

For (G.ii), write $\nabla_h u \in \mathcal{RT}_0(\mathcal{E}_h)$ as $\nabla_h u = \sum_i \mu_i e_i$. Then

$$\mu_i = \sum_{j=0}^{N_E-1} \mu_j \int_{\Omega} e_i \cdot e_j = \int_{\Omega} e_i \cdot \left(\sum_{j=0}^{N_E-1} \mu_j e_j \right) = \int_{\Omega} e_i \cdot \nabla_h u \stackrel{(4.14)}{=} - \int_{\Omega} u (\nabla \cdot e_i).$$

If $u \in H^1(\Omega)$, we can perform partial integration and get (G.iii). To see (G.iv), remember from Proposition 4.7 that the divergence of a Raviart–Thomas vector field is piecewise constant on the triangles. Therefore

$$\int_{\Omega} u (\nabla \cdot e_i) = \sum_{k=0}^{N_T-1} \int_{T_k} u (\nabla \cdot e_i) = \sum_{k=0}^{N_T-1} (\nabla \cdot e_i)|_{T_k} \int_{T_k} u$$

and by definition of R_h^T

$$= \sum_{k=0}^{N_T-1} (\nabla \cdot e_i) \Big|_{T_k} \int_{T_k} R_h^T(u) = \sum_{k=0}^{N_T-1} \int_{T_k} R_h^T(u) (\nabla \cdot e_i) = \int_{\Omega} R_h^T(u) (\nabla \cdot e_i).$$

Using representation (G.ii) then yields the assertion. \square

Together with Lemma 4.10, the last two properties yield the following commutative diagram

$$\begin{array}{ccccc} H(\nabla \nabla \cdot) & \xrightarrow{\nabla \cdot} & H^1(\Omega) & \xrightarrow{\nabla} & L^2(\Omega)^2 \\ \downarrow R_h^E & & \downarrow P_h^T & & \downarrow P_h^E \\ \mathcal{RT}_0(\mathcal{E}_h) & \xrightarrow{\nabla \cdot} & \mathcal{L}_0(\mathcal{T}_h) & \xrightarrow{\nabla_h} & \mathcal{RT}_0(\mathcal{E}_h) \end{array} \quad (4.15)$$

Corollary 4.15. *The discrete gradient is exact for linear functions in the sense that $\nabla_h u = \nabla u$ for affine u .*

Proof. If u is affine, then $\nabla u \in \mathcal{RT}_0(\mathcal{E}_h)$, therefore $\nabla u = P_h^E(\nabla u) = \nabla_h u$. \square

4.5 The fully discrete equation

First a remark on notation: to reduce overhead, we drop the “ $i + 1$ ” at J_h and use in the following

$$J_h^* \quad \text{instead of} \quad (J_*^{i+1})_h = (J_*^{k+1, i+1})_h.$$

We now spatially discretize the time-discrete minimization problem

$$\min_{\tilde{J} \in H(\nabla \nabla \cdot)} \int_{\Omega} \frac{1}{a\tau M(\phi)} |\tilde{J}|^2 + \varepsilon^{-1} \int_{\Omega} G''(\phi) |\nabla \cdot \tilde{J}|^2 + \varepsilon \int_{\Omega} |\nabla \nabla \cdot \tilde{J}|^2 - \int_{\Omega} r(\nabla \cdot \tilde{J}),$$

using the discrete gradient. In the notation of Section 3.2, the above problem is

$$\min_{\tilde{J} \in H(\nabla \nabla \cdot)} a_{\phi}^h(\tilde{J}, \tilde{J}) - f(\tilde{J}).$$

We define the discrete bilinear form $a_{\phi}^h : H(\nabla \cdot) \times H(\nabla \cdot) \rightarrow \mathbb{R}$ as

$$a_{\phi}^h(J, \tilde{J}) := \int_{\Omega} \frac{1}{a\tau M(\phi)} J \cdot \tilde{J} + \frac{1}{\varepsilon} \int_{\Omega} G''(\phi) (\nabla \cdot J) (\nabla \cdot \tilde{J}) + \varepsilon \int_{\Omega} \nabla_h \nabla \cdot J \cdot \nabla_h \nabla \cdot \tilde{J}. \quad (4.16)$$

For the right-hand side, we define $f_h : H(\nabla \cdot) \rightarrow \mathbb{R}$ by

$$f_h(\tilde{J}) := \int_{\Omega} r_h(\nabla \cdot \tilde{J})$$

where r_h is defined below in (4.18). Then the fully discrete minimization problem is:

$$\min_{\tilde{J}_h \in \mathcal{RT}_0(\mathcal{E}_h)} a_{\phi_h^i}^h(\tilde{J}_h, \tilde{J}_h) - f_h(\tilde{J}_h). \quad (4.17)$$

Since $\mathcal{RT}_0(\mathcal{E}_h) \not\subset H(\nabla \nabla \cdot)$, this is a *non-conforming* discretization. However, note that a^h is defined on $H(\nabla \cdot)$ and both $H(\nabla \nabla \cdot)$ and $\mathcal{RT}_0(\mathcal{E}_h)$ are subsets of $H(\nabla \cdot)$.

Lemma 4.16 (Positivity). *Let the time-step size be small enough, namely*

$$\tau < \frac{4}{\beta^2} \varepsilon^3,$$

where $\beta > 0$ is such that $G''(\phi) \geq -\beta$. Then

$$a_{\phi}^h(\tilde{J}, \tilde{J}) > 0$$

for all $\tilde{J} \neq 0$, $\tilde{J} \in \mathcal{RT}_0(\mathcal{E}_h)$.

Proof. By the definition of the discrete gradient, we can use partial integration. Therefore, the proof is the same as for Proposition 3.5. \square

Since $\mathcal{RT}_0(\mathcal{E}_h)$ is finite-dimensional, we get immediately

Corollary 4.17 (Existence of a solution). *The minimization problem (4.17) admits a unique solution $J_h^* \in \mathcal{RT}_0(\mathcal{E}_h)$.*

The optimality conditions of (4.17) yield the fully discrete equations:

Given $\phi_h^i, \phi_h^k \in \mathcal{L}_0(\mathcal{T}_h)$, find $J_h^* \in \mathcal{RT}_0(\mathcal{E}_h)$ such that

$$\begin{aligned} \int_{\Omega} \frac{1}{a\tau M(\phi_h^i)} J_h^* \cdot \tilde{J}_h + \varepsilon^{-1} \int_{\Omega} G''(\phi_h^i) (\nabla \cdot J_h^*) (\nabla \cdot \tilde{J}_h) + \varepsilon \int_{\Omega} \nabla_h \nabla \cdot J_h^* \cdot \nabla_h \nabla \cdot \tilde{J}_h \\ = \int_{\Omega} r_h(\nabla \cdot \tilde{J}_h) \quad \forall \tilde{J}_h \in \mathcal{RT}_0(\mathcal{E}_h) \end{aligned} \quad (4.18a)$$

where

$$r_h = \frac{1}{a\tau} w_h^* + \left(-\varepsilon \Delta_h + \varepsilon^{-1} G''(\phi_h^i) \right) \left((1 + \gamma) f_h - \frac{1}{a\tau} (\phi_h^i - \phi_h^k) \right) \in \mathcal{L}_0(\mathcal{T}_h),$$

$w_h^* = w_h^i + \gamma w_h^k$, but bear in mind (B.5) for TR-BDF2,

$$w_h^i = -\varepsilon \Delta_h \phi_h^i + \varepsilon^{-1} G'(\phi^i),$$

and $\Delta_h := \nabla \cdot \nabla_h$ is the divergence of the discrete gradient. Then find $\phi_h^{i+1} \in \mathcal{L}_0(\mathcal{T}_h)$ satisfying

$$\frac{1}{a\tau} \int_{\Omega} (\phi_h^{i+1} - \phi_h^k) \zeta_h + \int_{\Omega} (\nabla \cdot J_h^*) \zeta_h = \int_{\Omega} (1 + \gamma) f_h \zeta_h \quad \forall \zeta_h \in \mathcal{L}_0(\mathcal{T}_h). \quad (4.18b)$$

By Proposition 4.7, we have

$$J_h \in \mathcal{RT}_0(\mathcal{E}_h) \implies \nabla \cdot J_h \in \mathcal{L}_0(\mathcal{T}_h),$$

so Equation (4.18b) is, due to the choice of the basis (4.5), just an element-wise evaluation:

$$\phi_h^{i+1}|_T = \phi_h^k|_T + a\tau \left((1+\gamma)f_h|_T - \nabla \cdot J_h^*|_T \right) \quad \forall T \in \mathcal{T}_h. \quad (4.19)$$

Finally, we establish the connection to the discretization idea presented in Section 4.1.

Proposition 4.18. *Problem (4.18) is equivalent to the following problem:*

Given $\phi_h^i, \phi_h^k \in \mathcal{L}_0(\mathcal{T}_h)$, find $(J_h^*, K_h) \in \mathcal{RT}_0(\mathcal{E}_h) \times \mathcal{RT}_0(\mathcal{E}_h)$ such that

$$\int_{\Omega} K_h \cdot \tilde{K}_h = \int_{\Omega} (\nabla \cdot J_h^*)(\nabla \cdot \tilde{K}_h) \quad \forall \tilde{K}_h \in \mathcal{RT}_0(\mathcal{E}_h) \quad (4.20a)$$

$$\begin{aligned} \int_{\Omega} \frac{1}{a\tau M(\phi_h^i)} J_h^* \cdot \tilde{J}_h + \varepsilon \int_{\Omega} (\nabla \cdot K_h)(\nabla \cdot \tilde{J}_h) + \varepsilon^{-1} \int_{\Omega} G''(\phi_h^i)(\nabla \cdot J_h^*)(\nabla \cdot \tilde{J}_h) \\ = \int_{\Omega} r_h(\nabla \cdot \tilde{J}_h) \quad \forall \tilde{J}_h \in \mathcal{RT}_0(\mathcal{E}_h). \end{aligned} \quad (4.20b)$$

where r_h and $\phi_h^{i+1} \in \mathcal{L}_0(\mathcal{T}_h)$ are as in (4.18).

Proof. Defining $K_h := -\nabla_h \nabla \cdot J_h^*$, we have

$$\varepsilon \int_{\Omega} \nabla_h \nabla \cdot J_h^* \cdot \nabla_h \nabla \cdot \tilde{J}_h \stackrel{(4.14)}{=} \varepsilon \int_{\Omega} (-\nabla \cdot \nabla_h \nabla \cdot J_h^*)(\nabla \cdot \tilde{J}_h) = \varepsilon \int_{\Omega} (\nabla \cdot K_h)(\nabla \cdot \tilde{J}_h).$$

and therefore we get the second equation. The first equation is just the definition of the weak gradient. \square

Looking at these equations, we can see that we realized the idea presentend in Section 4.1. Note furthermore that (4.4) was solved in $H(\nabla \cdot)$ and that $\mathcal{RT}_0(\mathcal{E}_h)$ is a subspace of $H(\nabla \cdot)$.

This formulation will again become important in Section 4.8.3.

4.6 Error estimates

We establish below some simple error estimates for the discrete problem. These are by no means exhaustive, in particular we assume $\phi_h = \phi$, but should give us a hint that our discretization is reasonable.

The (mesh-dependent) norm we use for the error estimates is

$$\|J\|_h^2 := \|J\|_{H(\nabla\cdot, \Omega)}^2 + \|\nabla_h \nabla \cdot J\|_{L^2(\Omega)}^2. \quad (4.21)$$

Note that both the norm and the discrete bilinear form (4.16) are defined on $H(\nabla\cdot, \Omega)$ and therefore are valid both for vector field in $H(\nabla\cdot, \Omega)$ and in $\mathcal{RT}_0(\mathcal{E}_h)$.

Since the space $\mathcal{RT}_0(\mathcal{E}_h)$ is finite-dimensional, we readily get

- Ellipticity, $a_\phi^h(J_h, J_h) \geq \alpha \|J_h\|_h^2$ for all $J_h \in \mathcal{RT}_0(\mathcal{E}_h)$.
- Continuity, $|a_\phi^h(J, J_h)| \leq C \|J\|_h \|J_h\|_h$ for all $J \in H(\nabla\cdot, \Omega)$, $J_h \in \mathcal{RT}_0(\mathcal{E}_h)$

with α, C independent of h . Therefore, we can use Strang's lemma.

Lemma 4.19 (Second Strang lemma). *Let $J \in H(\nabla\cdot, \Omega)$ and $J_h \in \mathcal{RT}_0(\mathcal{E}_h)$ be solutions of the problems (3.9) and (4.18), respectively. Then there exists a constant c independent of the subspace $\mathcal{RT}_0(\mathcal{E}_h)$ such that*

$$\|J - J_h\|_h \leq c \left(\inf_{\tilde{J}_h \in \mathcal{RT}_0(\mathcal{E}_h)} \|J - \tilde{J}_h\|_h + \sup_{\tilde{J}_h \in \mathcal{RT}_0(\mathcal{E}_h)} \frac{|a_\phi^h(J, \tilde{J}_h) - b_h(\tilde{J}_h)|}{\|\tilde{J}_h\|_h} \right). \quad (4.22)$$

In the following two lemmata, we estimate the terms in (4.22).

Lemma 4.20 (Best approximation). *Let $J \in H(\nabla\cdot, \Omega)$ be given. Then*

$$\inf_{\tilde{J}_h \in \mathcal{RT}_0(\mathcal{E}_h)} \|J - \tilde{J}_h\|_h \leq ch |J|_{H^1(\nabla\cdot, \Omega)}.$$

Proof. From the possible choices, we use $\tilde{J} = R_h^E(J)$. First, we note that

$$\nabla_h \nabla \cdot R_h^E(J) = \nabla_h R_h^T(\nabla \cdot J) = \nabla_h \nabla \cdot J$$

by Lemma 4.10 and (G.iv). Therefore

$$\|\nabla_h \nabla \cdot J - \nabla_h \nabla \cdot R_h^E(J)\|_{L^2(\Omega)} = 0$$

and thus

$$\|J - R_h^E(J)\|_h = \|J - R_h^E(J)\|_{H(\nabla\cdot, \Omega)}.$$

Now we can use Proposition 4.11 to see

$$\begin{aligned} \inf_{\tilde{J}_h \in \mathcal{RT}_0(\mathcal{E}_h)} \|J - \tilde{J}_h\|_h &\leq \|J - R_h^E(J)\|_h = \|J - R_h^E(J)\|_{H(\nabla\cdot, \Omega)} \\ &\leq ch |J|_{H^1(\nabla\cdot, \Omega)}. \end{aligned}$$

□

Lemma 4.21. *Let $J \in H(\nabla\cdot, \Omega)$ and $J_h \in \mathcal{RT}_0(\mathcal{E}_h)$ be solutions of the problems (3.9) and (4.18), respectively, and assume $\nabla \nabla \cdot J \in H(\nabla\cdot, \Omega)$. Then there exists a constant c independent of h such that*

$$|a_\phi^h(J, \tilde{J}_h) - b(\tilde{J}_h)| \leq ch |\nabla \nabla \cdot J|_{H^1(\Omega)} \|\nabla_h \nabla \cdot \tilde{J}_h\|_{L^2(\Omega)}.$$

Proof. Integration by parts in $a_\phi(J, \tilde{J})$ shows that it is enough for \tilde{J} to be in $H(\nabla \cdot, \Omega)$, see (4.3). Therefore we can choose $\tilde{J} = \tilde{J}_h$ and get

$$\int_{\Omega} \frac{1}{a\tau M(\phi)} J \cdot \tilde{J}_h + \frac{1}{\varepsilon} \int_{\Omega} G''(\phi)(\nabla \cdot J)(\nabla \cdot \tilde{J}_h) = \varepsilon \int_{\Omega} (\Delta \nabla \cdot J)(\nabla \cdot \tilde{J}_h) + \int_{\Omega} r(\nabla \cdot \tilde{J}_h).$$

We use this identity to replace the first part of $a_\phi^h(J, \tilde{J}_h)$.

$$\begin{aligned} a_\phi^h(J, \tilde{J}_h) &= \int_{\Omega} \frac{1}{a\tau M(\phi)} J \cdot \tilde{J}_h + \frac{1}{\varepsilon} \int_{\Omega} G''(\phi)(\nabla \cdot J)(\nabla \cdot \tilde{J}_h) + \varepsilon \int_{\Omega} \nabla_h \nabla \cdot J \cdot \nabla_h \nabla \cdot \tilde{J}_h \\ &= b(\tilde{J}_h) + \varepsilon \int_{\Omega} (\Delta \nabla \cdot J)(\nabla \cdot \tilde{J}_h) + \varepsilon \int_{\Omega} \nabla_h \nabla \cdot J \cdot \nabla_h \nabla \cdot \tilde{J}_h. \end{aligned}$$

Since $\nabla \cdot \tilde{J}_h \in \mathcal{L}_0(\mathcal{T}_h)$, we get with (G.i)

$$a_\phi^h(J, \tilde{J}_h) - b(\tilde{J}_h) = \varepsilon \int_{\Omega} (R_h^E(\nabla \nabla \cdot J) - \nabla_h \nabla \cdot J) \cdot \nabla_h \nabla \cdot \tilde{J}_h.$$

To estimate this term, we use the triangle inequality and (G.iii) to obtain

$$\begin{aligned} \|R_h^E(\nabla \nabla \cdot J) - \nabla_h \nabla \cdot J\|_{L^2(\Omega)} &\leq \|R_h^E(\nabla \nabla \cdot J) - \nabla \nabla \cdot J\|_{L^2(\Omega)} + \|\nabla_h \nabla \cdot J - \nabla \nabla \cdot J\|_{L^2(\Omega)} \\ &= \|R_h^E(\nabla \nabla \cdot J) - \nabla \nabla \cdot J\|_{L^2(\Omega)} + \|P_h^E(\nabla \nabla \cdot J) - \nabla \nabla \cdot J\|_{L^2(\Omega)} \\ &\leq 2\|R_h^E(\nabla \nabla \cdot J) - \nabla \nabla \cdot J\|_{L^2(\Omega)}, \end{aligned}$$

by the best-approximation property of the orthogonal projection. So finally with the help of Proposition 4.11

$$\begin{aligned} |a_\phi^h(J, \tilde{J}_h) - b(\tilde{J}_h)| &\leq \varepsilon \|R_h^E(\nabla \nabla \cdot J) - \nabla_h \nabla \cdot J\|_{L^2(\Omega)} \|\nabla_h \nabla \cdot \tilde{J}_h\|_{L^2(\Omega)} \\ &\leq 2\varepsilon \|R_h^E(\nabla \nabla \cdot J) - \nabla \nabla \cdot J\|_{L^2(\Omega)} \|\nabla_h \nabla \cdot \tilde{J}_h\|_{L^2(\Omega)} \\ &\leq ch |\nabla \nabla \cdot J|_{H^1(\Omega)} \|\nabla_h \nabla \cdot \tilde{J}_h\|_{L^2(\Omega)}. \end{aligned}$$

□

Analogous to the $H(\nabla \cdot)$ case, we generalize the definition of $H(\nabla \nabla \cdot)$.

Definition 4.22 (H^m -grad-div). The spaces $H^m(\nabla \nabla \cdot, \Omega)$, $m \geq 0$, are the spaces of all vector fields $\xi \in H^m(\Omega)^2$ with $\nabla \cdot \xi \in H^{m+1}(\Omega)$. The norm is given by

$$\|\xi\|_{H^m(\nabla \nabla \cdot, \Omega)}^2 := \|\xi\|_{H^m(\Omega)}^2 + \|\nabla \cdot \xi\|_{H^m(\Omega)}^2 + \|\nabla \nabla \cdot \xi\|_{H^m(\Omega)}^2.$$

We also define the semi-norm

$$|\xi|_{H^m(\nabla \nabla \cdot, \Omega)}^2 := |\xi|_{H^m(\Omega)}^2 + |\nabla \cdot \xi|_{H^m(\Omega)}^2 + |\nabla \nabla \cdot \xi|_{H^m(\Omega)}^2.$$

◇

Using this definition, we summarize the results in

Proposition 4.23. Let $J \in H(\nabla \nabla \cdot, \Omega)$ and $J_h \in \mathcal{RT}_0(\mathcal{E}_h)$ be solutions of the problems (3.9) and (4.18), respectively, with $r \in L^2(\Omega)$. Furthermore, assume that $\phi_h = \phi$. Then there exists a constant c independent of h such that

$$\|J - J_h\|_h \leq ch |J|_{H^1(\nabla \nabla \cdot, \Omega)}.$$

4.7 Matrix representation

Using the basis (4.5) for $\mathcal{L}_0(\mathcal{T}_h)$ and (4.12) for $\mathcal{RT}_0(\mathcal{E}_h)$, we define a number of matrices. First the three mass matrices $\mathbf{V} : \mathbb{R}^{N_T} \rightarrow \mathbb{R}^{N_T}$ and $\mathbf{B}_0, \mathbf{B}_1^i : \mathbb{R}^{N_E^0} \rightarrow \mathbb{R}^{N_E^0}$

$$\mathbf{V}_{kl} := \int_{\Omega} Z_k(x) Z_l(x) = \delta_{kl} |T_k|$$

$$(\mathbf{B}_0)_{ij} := \int_{\Omega} \Psi_i(x) \cdot \Psi_j(x) \quad \text{and} \quad (\mathbf{B}_1^i)_{ij} := \int_{\Omega} \frac{1}{M(\phi^i(x))} \Psi_i(x) \cdot \Psi_j(x).$$

Next the two stiffness matrices $\mathbf{A}_0, \mathbf{A}_1 : \mathbb{R}^{N_E^0} \rightarrow \mathbb{R}^{N_E^0}$

$$(\mathbf{A}_0)_{ij} := \int_{\Omega} (\nabla \cdot \Psi_i(x)) (\nabla \cdot \Psi_j(x)),$$

$$(\mathbf{A}_1^i)_{ij} := \int_{\Omega} G''(\phi_h^i(x)) (\nabla \cdot \Psi_i(x)) (\nabla \cdot \Psi_j(x)).$$

Proposition 4.24. *The matrices $\mathbf{B}_0, \mathbf{B}_1^i, \mathbf{A}_0$ and \mathbf{A}_1^i are symmetric and have at most five entries per row. Furthermore, \mathbf{B}_0 and \mathbf{B}_1^i are positive definite and \mathbf{A}_0 is positive semi-definite.*

Proof. Symmetry is obvious. For the number of entries remember that Ψ_i has support only on T_i^{\pm} . Therefore, $(\mathbf{B}_0)_{ij}$ is only nonzero for those j for which E_j is adjacent to the same triangle as E_i . This is only true for the five edges shown in Figure 4.2. To see that the matrices are \mathbf{B}_0 and \mathbf{B}_1^i positive definite, consider

$$\begin{aligned} \mathbf{B}_1 \underline{\xi} \cdot \underline{\xi} &= \sum_{ij} (\mathbf{B}_1)_{ij} \underline{\xi}_i \underline{\xi}_j = \int_{\Omega} \frac{1}{M(\phi)} \left(\sum_i \underline{\xi}_i \Psi_i(x) \right) \cdot \left(\sum_j \underline{\xi}_j \Psi_j(x) \right) = \int_{\Omega} \frac{1}{M(\phi)} |\underline{\xi}_h|^2 \\ &\geq \int_{\Omega} |\underline{\xi}_h|^2. \end{aligned}$$

The right hand side is zero if and only if $\underline{\xi} = 0$. In the same way, we get

$$\mathbf{A}_0 \underline{\xi} \cdot \underline{\xi} = \int_{\Omega} (\nabla \cdot \underline{\xi}_h)^2 \geq 0.$$

Since $G''(x)$ may be negative, \mathbf{A}_1 is not positive definite in general, compare Section 3.2. \square

Finally we need the gradient and divergence matrices $\mathbf{G} : \mathbb{R}^{N_T} \rightarrow \mathbb{R}^{N_E^0}$ and $\mathbf{D} : \mathbb{R}^{N_E^0} \rightarrow \mathbb{R}^{N_T}$

$$\mathbf{G}_{ik} := - \int_{\Omega} Z_k (\nabla \cdot \Psi_i) \quad \text{and} \quad \mathbf{D}_{ki} := \int_{\Omega} Z_k (\nabla \cdot \Psi_i). \quad (4.23)$$

Although we call these matrices gradient and divergence, they clearly do not correspond to the (strong version of the) gradient and the divergence. Rather, they have to be multiplied with the inverse of their respective mass matrix:

$$\begin{aligned} g = \nabla_h \phi &\longleftrightarrow \underline{g} = \mathbf{B}_0^{-1} \mathbf{G} \phi & \nabla_h &\longleftrightarrow \mathbf{B}_0^{-1} \mathbf{G}, \\ \phi = \nabla \cdot J &\longleftrightarrow \underline{\phi} = \mathbf{V}^{-1} \mathbf{D} J & \nabla \cdot &\longleftrightarrow \mathbf{V}^{-1} \mathbf{D}. \end{aligned}$$

As is the case for their continuous counterparts, $-\mathbf{G}$ and \mathbf{D} are dual to each other in the sense that

$$\mathbf{B}_0 \mathbf{B}_0^{-1} \mathbf{G} \underline{\phi} \cdot \underline{g} =: \langle \mathbf{B}_0^{-1} \mathbf{G} \underline{\phi}, \underline{g} \rangle_{\varepsilon_h} = -\langle \underline{\phi}, \mathbf{V}^{-1} \mathbf{D} \underline{g} \rangle_{T_h} := -\mathbf{V} \underline{\phi} \cdot \mathbf{V}^{-1} \mathbf{D} \underline{g},$$

or simply

$$\mathbf{G}^t = -\mathbf{D}.$$

Note that we have the relation

$$\begin{aligned} (\mathbf{A}_1)_{ij} &= \int_{\Omega} G''(\phi_h)(\nabla \cdot \Psi_i)(\nabla \cdot \Psi_j) = \sum_{k=0}^{N_T-1} |T_k|^{-1} G''(\underline{\phi}_k) |T_k| (\nabla \cdot \Psi_i)|_{T_k} |T_k| (\nabla \cdot \Psi_j)|_{T_k} \\ &= \sum_{k=0}^{N_T-1} |T_k|^{-1} G''(\underline{\phi}_k) \left(\int_{\Omega} Z_k(\nabla \cdot \Psi_i) \right) \left(\int_{\Omega} Z_k(\nabla \cdot \Psi_j) \right) = - \sum_{k=0}^{N_T-1} \mathbf{G}_{ik} G''(\underline{\phi}_k) |T_k|^{-1} \mathbf{D}_{kj} \\ &= -(\mathbf{GNV}^{-1} \mathbf{D})_{ij} \end{aligned}$$

with $\mathbf{N}_{kk} = G''(\underline{\phi}_k)$. In other words

$$\mathbf{A}_0 = -\mathbf{GV}^{-1} \mathbf{D} \quad \text{and} \quad \mathbf{A}_1^i = -\mathbf{GN}^i \mathbf{V}^{-1} \mathbf{D}.$$

Therefore, the relation between the operator $\nabla_h \nabla \cdot$ and the stiffness matrix \mathbf{A}_0 is

$$\nabla_h \nabla \cdot \longleftrightarrow -\mathbf{B}_0^{-1} \mathbf{A}_0$$

With these definitions and the notation

$$\underline{G}'(\underline{\phi}^i) := \left(G'(\underline{\phi}_k^i) \right)_{k=0, \dots, N_T-1},$$

the matrix representation of (4.18) is

$$\frac{1}{a\tau} \mathbf{B}_1^i \underline{J}^* + \varepsilon \mathbf{A}_0 \mathbf{B}_0^{-1} \mathbf{A}_0 \underline{J}^* + \varepsilon^{-1} \mathbf{A}_1^i \underline{J}^* = -\mathbf{G} \underline{r} \quad (4.24)$$

with the right hand side

$$\underline{r} = \frac{1}{a\tau} \underline{w}^* - \varepsilon \mathbf{V}^{-1} \mathbf{D} \underline{g}_u + \varepsilon^{-1} \mathbf{N}^i \underline{u}^i, \quad (4.25a)$$

where \underline{g}_u and \underline{u}^i are defined by

$$\mathbf{B}_0 \underline{g}_u = \mathbf{G} \underline{u}^i, \quad \underline{u}^i = (1 + \gamma) \underline{f} - \frac{1}{a\tau} (\underline{\phi}^i - \underline{\phi}^k) \quad (4.25b)$$

and $\underline{w}^* = \underline{w}^i + \gamma \underline{w}^k$ (again, $\underline{w}^* = \underline{w}^i + \gamma/2 \cdot (\underline{w}^k + \underline{w}^{k-1})$ for TR-BDF2) with

$$\mathbf{B}_0 \underline{g}_\phi = \mathbf{G} \underline{\phi}^i, \quad \underline{w}^i = \varepsilon \mathbf{V}^{-1} \mathbf{D} \underline{g}_\phi + \varepsilon^{-1} \underline{G}'(\underline{\phi}^i). \quad (4.25c)$$

Updating $\underline{\phi}$ is done by

$$\underline{\phi}^{i+1} = \underline{\phi}^k + a\tau \left((1 + \gamma) \underline{f} - \mathbf{V}^{-1} \mathbf{D} \underline{J}^* \right). \quad (4.26)$$

Table 4.1: Algorithm to solve the Equations (4.24)–(4.26). Time- and space-adaptivity is not included in the algorithm.

```

1: Set initial condition  $\phi_0$ 
2: Assemble matrices  $\mathbf{A}_0$  and  $\mathbf{B}_0$ 
3: for  $n = 0, \dots, \#\text{timesteps} - 1$  do
4:   for all substeps  $k$  do
5:     for  $i = 0, \dots, I - 1$  do
6:       Assemble matrices  $\mathbf{A}_1^i$  and  $\mathbf{B}_1^k$ 
7:       Setup  $\underline{r}$  using (4.25)
8:       Solve (4.24) for  $\underline{J}^{i+1}$ 
9:       Update  $\underline{\phi}$  by (4.26)
10:    end for
11:  end for
12: end for

```

4.8 Solving the fully discrete equation

A generic procedure to solve Equations (4.24)–(4.26) is shown in Table 4.1. Of course, the main work is to solve (4.24) in Line 8. How to do this is covered in the following.

The challenge in solving this equation is that with Raviart–Thomas finite elements, one has not the possibility to lump masses, see Remark 4.25 below, so the inverse of \mathbf{B}_0 is a dense matrix. We propose several approaches to circumvent this difficulty:

1. Use a matrix-free method and compute the weak gradient by solving an equation in every step of an iterative solver.
2. Calculate a sparse approximate symmetric inverse of \mathbf{B}_0 .
3. Use a larger space for K_h and confine it to $\mathcal{RT}_0(\mathcal{E}_h)$ via a side condition, the “Arnold–Brezzi Method”.

These will be presented in the sections 4.8.1–4.8.3. Another possibility would be to revert to the split equation (4.4) and solve

$$\begin{pmatrix} -\varepsilon \mathbf{B}_0 & \varepsilon \mathbf{A}_0 \\ \varepsilon \mathbf{A}_0 & \frac{1}{a\tau} \mathbf{B}_1^i + \varepsilon^{-1} \mathbf{A}_1^i \end{pmatrix} \begin{pmatrix} \underline{K} \\ \underline{J} \end{pmatrix} = \begin{pmatrix} \underline{0} \\ -\mathbf{G}\underline{r} \end{pmatrix},$$

for $(\underline{K}, \underline{J})$. The disadvantage of this approach is that the system matrix is indefinite, see Section 4.8.4, and has twice as many rows. As such, it is inferior to method 3.

A completely different approach was finally taken by not using the discrete gradient defined by duality, but to avoid $\mathcal{RT}_0(\mathcal{E}_h)$ altogether when discretizing the fourth-order term. We therefore define a gradient $\mathcal{L}_0(\mathcal{T}_h) \rightarrow \mathcal{L}_0(\mathcal{T}_h)^2$ and use this gradient to discretize the fourth-order term. To achieve this, we

4. Define the energy on $\mathcal{L}_0(\mathcal{T}_h)$,

see Section 4.8.5.

In Section 4.11, we compare these methods. For that purpose, we first check the convergence for Laplace's problem and then compare their performance on three application problems.

We will use the following abbreviations for the different methods

1. *SINE* for Solve IN Every step for the method presented in Section 4.8.1,
2. *SPASI* for the SParse Approximate Inverse of Section 4.8.2,
3. *ABM* for the Arnold–Brezzi method of Section 4.8.3 and
4. *PCE* for method of Section 4.8.5, where we define a Piecewise Constant Energy.

Remark 4.25. Lumping masses, i.e. replacing \mathbf{B}_0 by a diagonal matrix $\tilde{\mathbf{B}}_0$, means to change the discrete gradient. So using any mass-lumping procedure, we would get a different discrete gradient

$$\tilde{\nabla}_h u_h|_T = \sum_{i \in \mathcal{E}(T)} \left(\tilde{\mathbf{B}}_0^{-1} \mathbf{G} \underline{u} \right)_i.$$

As a minimal condition for a gradient to be acceptable, we want that

$$f(x) = a \cdot x + b \quad \implies \quad \tilde{\nabla}_h f = a.$$

A straightforward calculation shows that this is only possible if the mesh is hexagonal. Therefore, using any mass-lumping technique is not suitable for our needs. \diamond

4.8.1 SINE

If we use an iterative solver to solve the equation

$$\left[\frac{1}{a\tau} \mathbf{B}_1^i + \varepsilon \mathbf{A}_0 \mathbf{B}_0^{-1} \mathbf{A}_0 + \varepsilon^{-1} \mathbf{A}_1^i \right] \underline{J}^* = -\mathbf{G} \underline{r}, \quad (4.27)$$

we do not need to assemble the (dense) matrix $\mathbf{A}_0 \mathbf{B}_0^{-1} \mathbf{A}_0$, but it is enough to know the result of the multiplication $\mathbf{A}_0 \mathbf{B}_0^{-1} \mathbf{A}_0 \underline{d}$ in each step of the iterative solver. So we set

$$\mathbf{A}_0 \mathbf{B}_0^{-1} \mathbf{A}_0 \underline{d} := \mathbf{A}_0 \underline{z}, \quad \text{where } \underline{z} \text{ is the solution of } \mathbf{B}_0 \underline{z} = \mathbf{A}_0 \underline{d}.$$

By Lemma 4.16, the left-hand side is positive definite, so we can use the conjugate gradient method as iterative solver.

The main disadvantage of this method is that the solving process is quite slow, since first of all we have to solve another $N_E^0 \times N_E^0$ system in every cg-step. In addition, as we can only use a matrix-free solver, standard preconditioner such as incomplete Cholesky cannot be used. We have not investigated a tailored preconditioner in the finite-element case.

4.8.2 Sparse approximate inverse

To avoid having to solve a LSE in each time step, we construct an approximate inverse

$$\mathbf{I} \approx \mathbf{B}_0^{-1},$$

so that we can actually assemble the operator and use the conjugate gradient method with an incomplete Cholesky preconditioner or a direct method. The approximate inverse should be *sparse* and, just like the full inverse, *symmetric* and *positive definite*.

Since the entries outside the sparsity pattern of the mass matrix are exponentially small, we look for a matrix \mathbf{I} having the same nonzero pattern as \mathbf{B}_0 .

Given such an approximate inverse \mathbf{I} , we can replace \mathbf{B}_0^{-1} in Equation (4.24) and solve

$$\left[\frac{1}{a\tau} \mathbf{B}_1^k + \varepsilon \mathbf{A}_0 \mathbf{I} \mathbf{A}_0 + \varepsilon^{-1} \mathbf{A}_1^i \right] \underline{J}^* = -\underline{\mathbf{G}} \underline{r}.$$

Note that the fourth-order term $\mathbf{A}_0 \mathbf{I} \mathbf{A}_0$ does not depend on ϕ , so the sparse approximate inverse needs only to be reconstructed when the mesh changes.

Our first approach to construct \mathbf{I} was to consider the entries \mathbf{I}_{ij} with $i \leq j$ as unknowns, set $\mathbf{I}_{ji} := \mathbf{I}_{ij}$ and then minimize

$$\|\mathbf{B}_0 \mathbf{I} - \text{id}\|_F,$$

where $\|\cdot\|_F$ is the Frobenius norm defined below. By construction, the result is symmetric and sparse but, unfortunately, we found that it is not positive definite.

To get a positive definite approximate inverse, we use a construction based on Kolotilina and Yeregin [1993]: \mathbf{B}_0 is symmetric positive definite, so we can factorize the mass matrix into $\mathbf{B}_0 = \mathbf{L}^t \mathbf{L}$. We now search for a lower triangular matrix \mathbf{S} with given sparsity pattern and $\mathbf{S} \approx \mathbf{L}^{-1}$. Then, we set

$$\mathbf{I} := \mathbf{S} \mathbf{S}^t \approx \mathbf{L}^{-1} \mathbf{L}^{-t} = \mathbf{B}_0^{-1}.$$

Let us first fix some notation.

Definition 4.26. The Frobenius matrix norm $\|\cdot\|_F$ is given by

$$\|\mathbf{M}\|_F^2 := \text{tr}(\mathbf{M}^t \mathbf{M}) = \sum_{ij} M_{ij}^2.$$

A *sparsity pattern* \mathcal{S}_M of a matrix \mathbf{M} is a set of indices with the property

$$(i, j) \notin \mathcal{S}_M \implies \mathbf{M}_{ij} = 0.$$

We denote by \mathcal{L} be the space of all lower triangular matrices with a fixed sparsity pattern \mathcal{S}^* to be defined later. Note that we have

$$\mathcal{S}^* \subset \{(i, j) \mid i \geq j\} \tag{4.28}$$

since we look for a lower triangular matrix. \diamond

With these definitions, we can render the statement $\mathbf{S} \approx \mathbf{L}^{-1}$ precise: We search

$$\mathbf{S} = \arg \min_{\hat{\mathbf{S}} \in \mathcal{L}} \|\mathbf{id} - \mathbf{L}\hat{\mathbf{S}}\|_{\mathbb{F}}^2,$$

such that

$$\mathcal{S}_{\mathbf{S}\mathbf{S}^t} = \mathcal{S}_{\mathbf{B}_0}.$$

The necessary condition is

$$\left. \frac{d}{d\varepsilon} \right|_{\varepsilon=0} \|\mathbf{id} - \mathbf{L}(\mathbf{S} + \varepsilon\mathbf{T})\|_{\mathbb{F}}^2 = 0 \quad \forall \mathbf{T} \in \mathcal{L}.$$

Inserting the definition of the norm yields

$$\begin{aligned} 0 &= \left. \frac{d}{d\varepsilon} \right|_{\varepsilon=0} \text{tr} [(\mathbf{id} - \mathbf{S}^t \mathbf{L}^t - \varepsilon \mathbf{T}^t \mathbf{L}^t)(\mathbf{id} - \mathbf{L}\mathbf{S} - \varepsilon \mathbf{L}\mathbf{T})] \\ &= \text{tr} [- (\mathbf{L}\mathbf{T} + (\mathbf{L}\mathbf{T})^t) + (\mathbf{S}^t \mathbf{L}^t \mathbf{L}\mathbf{T} + (\mathbf{S}^t \mathbf{L}^t \mathbf{L}\mathbf{T}))] \end{aligned}$$

and therefore

$$\text{tr} \mathbf{L}\mathbf{T} = \text{tr} \mathbf{S}^t \mathbf{L}^t \mathbf{L}\mathbf{T} \quad \forall \mathbf{T} \in \mathcal{L}. \quad (4.29)$$

It is of course enough if the above condition is valid for a basis of \mathcal{L} , which we choose now. To construct one basis element $\tilde{\mathbf{T}}$, fix $(i_*, j_*) \in \mathcal{S}^*$ and set

$$\tilde{\mathbf{T}}_{ij} = \begin{cases} 1 & (i, j) = (i_*, j_*) \\ 0 & \text{else.} \end{cases}$$

i.e. $\tilde{\mathbf{T}}$ has only one non-zero entry. Then we have for any matrix \mathbf{A}

$$\text{tr} \mathbf{A}\tilde{\mathbf{T}} = \sum_k (\mathbf{A}\tilde{\mathbf{T}})_{kk} = \sum_{k,l} \mathbf{A}_{kl} \tilde{\mathbf{T}}_{lk} = \mathbf{A}_{j_*, i_*}$$

and (4.29) becomes

$$\begin{aligned} \mathbf{L}_{ji} &= (\mathbf{S}^t \mathbf{L}^t \mathbf{L})_{ji} & \forall (i, j) \in \mathcal{S}^* \\ \iff \mathbf{L}_{ij}^t &= (\mathbf{L}^t \mathbf{L}\mathbf{S})_{ij} = (\mathbf{B}_0 \mathbf{S})_{ij} & \forall (i, j) \in \mathcal{S}^*. \end{aligned}$$

Since \mathbf{L}^t is upper triangular and \mathcal{S}^* is the sparsity pattern of a lower triangular matrix, see (4.28), we get

$$(\mathbf{B}_0 \mathbf{S})_{ij} = \begin{cases} L_{ii} & i = j \\ 0 & \text{else.} \end{cases} \quad (4.30)$$

The values L_{ii} can for example be obtained by performing an incomplete Cholesky factorization of the mass matrix, since the diagonal entries are always present. If the L_{ii} are not known, Kolotilina and Yeregin propose to solve

$$(\mathbf{B}_0 \hat{\mathbf{S}})_{ij} = \delta_{ij} \quad (4.31)$$

and then scale $\hat{\mathbf{S}}$ with a diagonal matrix \mathbf{D} so that

$$(\mathbf{S}^t \mathbf{B}_0 \mathbf{S})_{ii} = (\mathbf{D} \hat{\mathbf{S}}^t \mathbf{B}_0 \hat{\mathbf{S}} \mathbf{D})_{ii} = 1.$$

We tried both approaches and found comparable results.

To solve (4.30) or (4.31), we need a bijection between \mathcal{S}^* and \mathbb{R}^s , where $s = \#\mathcal{S}^*$. Then we can translate the equations into a linear system of equations.

It remains to construct the sparsity pattern for \mathbf{S} .

Construction of \mathcal{S}^* .

We relax our initial requirement $\mathcal{S}_I = \mathcal{S}_{\mathbf{B}_0}$ to

$$\mathcal{S}_{\mathbf{B}_0} \subset \mathcal{S}_I = \mathcal{S}_{\mathbb{S}^s}.$$

Furthermore, instead of working with the set \mathcal{S}^* , we consider a matrix \mathbf{Z} with the property

$$\mathbf{Z}_{ij} = \begin{cases} 1 & (i, j) \in \mathcal{S}^* \\ 0 & (i, j) \notin \mathcal{S}^*. \end{cases}$$

So we want to find a lower triangular matrix \mathbf{Z} with

$$(i, j) \in \mathcal{S}_{\mathbf{B}_0} \implies (\mathbf{Z}\mathbf{Z}^t)_{ij} \neq 0. \quad (4.32)$$

Since $(\mathbf{Z}\mathbf{Z}^t)_{ji} = (\mathbf{Z}\mathbf{Z}^t)_{ij}$, we assume $i \leq j$. Then

$$(\mathbf{Z}\mathbf{Z}^t)_{ij} = \sum_{k=0}^{n-1} \mathbf{Z}_{ik} \mathbf{Z}_{jk} = \underbrace{\sum_{k=0}^i \mathbf{Z}_{ik} \mathbf{Z}_{jk}}_{k \leq i \leq j} + \underbrace{\sum_{k=i+1}^j \mathbf{Z}_{ik} \mathbf{Z}_{jk}}_{i < k \leq j} + \underbrace{\sum_{k=j+1}^{n-1} \mathbf{Z}_{ik} \mathbf{Z}_{jk}}_{i \leq j < k}.$$

Because \mathbf{Z} is a lower triangular matrix, we have $\mathbf{Z}_{ik} \mathbf{Z}_{jk} = 0$ if $i < k$ or $j < k$. Therefore

$$(\mathbf{Z}\mathbf{Z}^t)_{ij} = \sum_{k=0}^i \mathbf{Z}_{ik} \mathbf{Z}_{jk}.$$

To determine \mathbf{Z} , we first set $\mathbf{Z}_{ii} = 1$. Then we determine the other entries by induction in i :

$i = 0$:

$$\begin{aligned} (\mathbf{Z}\mathbf{Z}^t)_{0j} &= \mathbf{Z}_{00} \mathbf{Z}_{j0} = \mathbf{Z}_{j0} \\ \implies \mathbf{Z}_{j0} &:= \begin{cases} 1 & (0, j) \in \mathcal{S}_{\mathbf{B}_0} \\ 0 & \text{else} \end{cases} \quad j = 1, \dots, n-1. \end{aligned}$$

$i > 0$ with \mathbf{Z} already determined up to column $i - 1$:

$$(\mathbf{Z}\mathbf{Z}^t)_{ij} = \underbrace{\sum_{k=0}^{i-1} \mathbf{Z}_{ik}\mathbf{Z}_{jk}}_{=:\sigma_{ij} \text{ (known)}} + \mathbf{Z}_{ii}\mathbf{Z}_{ji} = \mathbf{Z}_{ji} + \sigma_{ij}$$

$$\stackrel{(4.32)}{\implies} \mathbf{Z}_{ji} := \begin{cases} 1 & (i, j) \in \mathcal{S}_{\mathbf{B}_0} \text{ and } \sigma_{ij} = 0 \\ 0 & \text{else} \end{cases} \quad j = i + 1, \dots, n - 1.$$

In this way, we can iteratively construct \mathbf{Z} and therefore get \mathcal{S}^* .

4.8.3 Arnold–Brezzi method

A third way to address the problem of inverting the mass matrix uses the equivalent formulation (4.20) and is based on ideas of Arnold and Brezzi [1985]. Remember that in the definition of $\mathcal{RT}_0(\mathcal{E}_h)$, (4.11), we had two conditions for a vector field K_h to be in $\mathcal{RT}_0(\mathcal{E}_h)$:

$$(C1) \quad K_h \in \mathcal{RT}_{-1}(\mathcal{T}_h),$$

$$(C2) \quad \sum_{T \in \mathcal{T}_h} \int_T \eta(K_h \cdot \nu_T) = 0 \quad \forall \eta \in C^\infty(\Omega).$$

Instead of searching K_h and \tilde{K}_h in $\mathcal{RT}_0(\mathcal{E}_h)$ in Equation (4.20), we now only request K_h and \tilde{K}_h to be in $\mathcal{RT}_{-1}(\mathcal{T}_h)$ and put (C2) as a constraint. Since in $\mathcal{RT}_{-1}(\mathcal{T}_h)$ the degrees of freedom on each triangle are completely independent of the other triangles, the mass matrix on $\mathcal{RT}_{-1}(\mathcal{T}_h)$ will be a block-diagonal matrix with 3×3 blocks and can be easily inverted.

Before we calculate the Lagrange multiplier for the constraint, note that, since $K_h \cdot \nu_T$ is constant on edges for $K_h \in \mathcal{RT}_{-1}(\mathcal{T}_h)$, it is enough to request (C2) for all η_h being constant on the edges:

$$(C2) \iff \sum_{T \in \mathcal{T}_h} \int_{\partial T} \eta_h(K_h \cdot \nu_T) = 0 \quad \forall \eta_h \in \mathcal{L}_0(\mathcal{E}_h),$$

where the space $\mathcal{L}_0(\mathcal{E}_h)$ is

$$\mathcal{L}_0(\mathcal{E}_h) := \left\{ \eta_h \in L^2(\mathcal{E}_h) \mid \eta_h|_E = \text{const} \quad \forall E \in \mathcal{E}_h \right\}.$$

Proposition 4.27. *Let $(K_h, J_h^*) \in \mathcal{RT}_0(\mathcal{E}_h) \times \mathcal{RT}_0(\mathcal{E}_h)$ be the solution of (4.20). Then there exists $\lambda_h \in \mathcal{L}_0(\mathcal{E}_h)$ such that $(K_h, \lambda_h) \in \mathcal{RT}_0(\mathcal{E}_h) \times \mathcal{L}_0(\mathcal{E}_h)$ is the unique solution of*

$$\int_{\Omega} K_h \cdot \tilde{K}_h - \sum_{T \in \mathcal{T}_h} \int_T (\nabla \cdot J_h^*)(\nabla \cdot \tilde{K}_h) = \sum_{T \in \mathcal{T}_h} \int_{\partial T} \lambda_h(\tilde{K}_h \cdot \nu_T) \quad \forall \tilde{K}_h \in \mathcal{RT}_{-1}(\mathcal{T}_h), \quad (4.33a)$$

$$\sum_{T \in \mathcal{T}_h} \int_{\partial T} \eta_h(K_h \cdot \nu_T) = 0 \quad \forall \eta_h \in \mathcal{L}_0(\mathcal{E}_h) \quad (4.33b)$$

Proof. Define the bilinear form $c : \mathcal{L}_0(\mathcal{E}_h) \times \mathcal{RT}_{-1}(\mathcal{T}_h) \rightarrow \mathbb{R}$ by

$$c(\eta_h, K_h) := \sum_{T \in \mathcal{T}_h} \int_{\partial T} \eta_h(K_h \cdot \nu_T).$$

① **Existence of λ_h**

Since c is a bilinear form, there exist linear operators

$$C : \mathcal{L}_0(\mathcal{E}_h) \rightarrow \mathcal{RT}_{-1}(\mathcal{T}_h)' \quad \text{and} \quad C^t : \mathcal{RT}_{-1}(\mathcal{E}_h) \rightarrow \mathcal{L}_0(\mathcal{E}_h)'$$

with

$$\langle C\eta_h, \tilde{K}_h \rangle = \langle \eta_h, C^t \tilde{K}_h \rangle = c(\eta_h, \tilde{K}_h) \quad \forall (\eta_h, \tilde{K}_h) \in \mathcal{L}_0^0(\mathcal{E}_h) \times \mathcal{RT}_{-1}(\mathcal{E}_h).$$

As we have already seen, we have

$$(c(\eta_h, \tilde{K}_h) = 0 \quad \forall \eta_h \in \mathcal{L}_0(\mathcal{E}_h)) \iff \tilde{K}_h \in \mathcal{RT}_0(\mathcal{E}_h),$$

which can be translated into

$$\tilde{K}_h \in \ker C^t \iff \tilde{K}_h \in \mathcal{RT}_0(\mathcal{E}_h).$$

Now define the linear form $s : \mathcal{RT}_{-1}(\mathcal{T}_h) \rightarrow \mathbb{R}$ by

$$\langle s, \tilde{K}_h \rangle := \int_{\Omega} K_h \cdot \tilde{K}_h - \sum_{T \in \mathcal{T}_h} \int_T (\nabla \cdot J_h^*)(\nabla \cdot \tilde{K}_h).$$

Since K_h is a solution of (4.20), we have

$$\begin{aligned} \langle s, \tilde{K}_h \rangle &= 0 \quad \forall \tilde{K}_h \in \mathcal{RT}_0(\mathcal{E}_h) \\ \implies \langle s, \tilde{K}_h \rangle &= 0 \quad \forall \tilde{K}_h \in \ker C^t \\ \implies s &\in (\ker C^t)^\perp \\ \implies s &\in \text{Im } C, \end{aligned}$$

by the closed range theorem. Therefore, there exists a $\lambda_h \in \mathcal{L}_0(\mathcal{E}_h)$ with

$$C\lambda_h = s \iff c(\lambda_h, \tilde{K}_h) = \langle s, \tilde{K}_h \rangle \quad \forall \tilde{K}_h \in \mathcal{RT}_{-1}(\mathcal{T}_h), \quad (4.34)$$

which is Equation (4.33a).

② **Uniqueness of λ_h**

Given two Lagrange multiplier $\lambda_1, \lambda_2 \in \mathcal{L}_0^0(\mathcal{T}_h)$, we have

$$c(\lambda_1 - \lambda_2, \tilde{K}_h) = c(\lambda_1, \tilde{K}_h) - c(\lambda_2, \tilde{K}_h) \stackrel{(4.34)}{=} \langle s, \tilde{K}_h \rangle - \langle s, \tilde{K}_h \rangle = 0$$

for all $\tilde{K}_h \in \mathcal{RT}_{-1}(\mathcal{T}_h)$.

It remains to show that $c(\eta_h, \tilde{K}_h) = 0$ for all $\tilde{K}_h \in \mathcal{RT}_{-1}(\mathcal{T}_h)$ implies $\eta_h \equiv 0$. Choose

$$\tilde{K}_i(x) := \Psi_i(x) \Big|_{T_i^+} \in \mathcal{RT}_{-1}(\mathcal{T}_h),$$

where Ψ_i is the vector field defined in (4.12). Then

$$0 = c(\eta_h, \tilde{K}_i) = \int_{\partial T_i^+} \eta_h(\Psi_i \cdot \nu_{T_i^+}) \stackrel{\text{Prop. 4.8}}{=} \int_{E_i} \eta_h = |E_i| \eta_h \Big|_{E_i} \implies \eta_h \Big|_{E_i} = 0.$$

Since i was arbitrary, we have $\eta_h \equiv 0$. □

Lemma 4.28. *Problem (4.18) is equivalent to the following problem:*

Find $(K_h, J_h^*, \lambda_h) \in \mathcal{RT}_{-1}(\mathcal{T}_h) \times \mathcal{RT}_0(\mathcal{E}_h) \times \mathcal{L}_0(\mathcal{E}_h)$ such that

$$\int_{\Omega} K_h \cdot \tilde{K}_h - \sum_{T \in \mathcal{T}_h} \int_T (\nabla \cdot J_h^*) (\nabla \cdot \tilde{K}_h) = \sum_{T \in \mathcal{T}_h} \int_{\partial T} \lambda_h (\tilde{K}_h \cdot \nu_T) \quad \forall \tilde{K}_h \in \mathcal{RT}_{-1}(\mathcal{T}_h), \quad (4.35a)$$

$$\sum_{T \in \mathcal{T}_h} \int_{\partial T} \eta_h (K_h \cdot \nu_T) = 0 \quad \forall \eta_h \in \mathcal{L}_0(\mathcal{E}_h), \quad (4.35b)$$

$$\begin{aligned} \int_{\Omega} \frac{1}{a\tau M(\phi_h^k)} J_h^* \cdot \tilde{J}_h + \varepsilon \int_{\Omega} (\nabla \cdot K_h) (\nabla \cdot \tilde{J}_h) + \varepsilon^{-1} \int_{\Omega} G''(\phi_h^i) (\nabla \cdot J_h^*) (\nabla \cdot \tilde{J}_h) \\ = \int_{\Omega} r_h (\nabla \cdot \tilde{J}_h) \quad \forall \tilde{J}_h \in \mathcal{RT}_0(\mathcal{E}_h). \end{aligned} \quad (4.35c)$$

Proof. It remains to show that any solution of (4.35) solves (4.20). First we note that due to (4.35b), K_h is in $\mathcal{RT}_0(\mathcal{E}_h)$. Next, for test functions $\tilde{K}_h \in \mathcal{RT}_0(\mathcal{E}_h) \subset \mathcal{RT}_{-1}(\mathcal{T}_h)$, the right hand side of (4.35a) is zero and so (4.35a) = (4.20a). The equation for J_h was not changed. Since the solution to (4.20) is unique and λ_h is unique, the solution to (4.35) is also unique. □

Matrix representation

The additional matrices we need are

- a mass matrix $\hat{\mathbf{B}}_0$ on $\mathcal{RT}_{-1}(\mathcal{T}_h)$,
- a modified stiffness matrix $\hat{\mathbf{A}}_0 : \mathcal{RT}_{-1}(\mathcal{T}_h) \rightarrow \mathcal{RT}_0(\mathcal{E}_h)$ and
- a matrix representation of C .

As basis for $\mathcal{RT}_{-1}(\mathcal{T}_h)$, we use

$$\hat{\Psi}_{3n+m}(x)|_{T_k} := \begin{cases} x & \text{for } n=k, m=0 \\ \begin{pmatrix} 1 \\ 0 \end{pmatrix} & \text{for } n=k, m=1 \\ \begin{pmatrix} 0 \\ 1 \end{pmatrix} & \text{for } n=k, m=2 \\ 0 & \text{else} \end{cases} \quad n=0, \dots, N_T-1, m=0, \dots, 2, \quad (4.36)$$

to define the mass matrix $\hat{\mathbf{B}}_0 : \mathbb{R}^{3N_T} \rightarrow \mathbb{R}^{3N_T}$

$$(\hat{\mathbf{B}}_0)_{3n_1+m_1, 3n_2+m_2} := \int_{\Omega} \hat{\Psi}_{3n_1+m_1} \cdot \hat{\Psi}_{3n_2+m_2},$$

and the new stiffness matrix $\hat{\mathbf{A}}_0 : \mathbb{R}^{3N_T} \rightarrow \mathbb{R}^{N_E^0}$.

$$(\hat{\mathbf{A}}_0)_{i, 3n+m} := \int_{\Omega} (\nabla \cdot \hat{\Psi}_{3n+m})(\nabla \cdot \Psi_i).$$

Finally, we need the matrix $\hat{\mathbf{C}} : \mathbb{R}^{N_E} \rightarrow \mathbb{R}^{3N_T}$

$$\hat{\mathbf{C}}_{3n+m, j} := \sum_{T \in \mathcal{T}_h} \int_{\partial T} \zeta_j(\hat{\Psi}_{3n+m} \cdot \nu_T) \quad \text{with} \quad \zeta_j(x) = \begin{cases} 1 & \text{on } E_j, \\ 0 & \text{else.} \end{cases}$$

With the definitions, we can write Equations (4.35) as

$$\frac{1}{a\tau} \mathbf{B}_1^k \underline{J}^* + \varepsilon \hat{\mathbf{A}}_0 \underline{K} + \varepsilon^{-1} \mathbf{A}_1^i \underline{J}^* = -\mathbf{G} \underline{r} \quad (4.37a)$$

$$\hat{\mathbf{B}}_0 \underline{K} - \hat{\mathbf{A}}_0^t \underline{J}^* - \hat{\mathbf{C}} \underline{\lambda} = 0 \quad (4.37b)$$

$$\hat{\mathbf{C}}^t \underline{K} = 0 \quad (4.37c)$$

Since the new mass matrix $\hat{\mathbf{B}}_0$ can be easily inverted, we rewrite the second equation to get

$$\underline{K} = \hat{\mathbf{B}}_0^{-1} (\hat{\mathbf{A}}_0^t \underline{J}^* - \hat{\mathbf{C}} \underline{\lambda})$$

and replace \underline{K} in the two other equations. This yields

$$\begin{pmatrix} \frac{1}{a\tau} \mathbf{B}_1^i + \varepsilon \hat{\mathbf{A}}_0 \hat{\mathbf{B}}_0^{-1} \hat{\mathbf{A}}_0^t + \varepsilon^{-1} \mathbf{A}_1^i & -\varepsilon \hat{\mathbf{A}}_0 \hat{\mathbf{B}}_0^{-1} \hat{\mathbf{C}} \\ -\varepsilon \hat{\mathbf{C}}^t \hat{\mathbf{B}}_0^{-1} \hat{\mathbf{A}}_0^t & \varepsilon \hat{\mathbf{C}}^t \hat{\mathbf{B}}_0^{-1} \hat{\mathbf{C}} \end{pmatrix} \begin{pmatrix} \underline{J}^* \\ \underline{\lambda} \end{pmatrix} = - \begin{pmatrix} \mathbf{G} \underline{r} \\ 0 \end{pmatrix} \quad (4.38)$$

where \underline{r} is the same as in (4.25). The update procedure for ϕ remains

$$\underline{\phi}^{i+1} = \underline{\phi}^k + a\tau \left((1+\gamma) \underline{f} - \mathbf{V}^{-1} \mathbf{D} \underline{J}^* \right).$$

Proposition 4.29. *For some $\tau > 0$, the matrix in (4.38) is symmetric and positive-definite.*

Proof. Denote the matrix by \mathbf{M} . The symmetry of \mathbf{M} is evident. For positive definiteness we show that

$$\mathbf{M} \begin{pmatrix} J \\ \underline{\lambda} \end{pmatrix} \cdot \begin{pmatrix} J \\ \underline{\lambda} \end{pmatrix} > 0 \quad \text{for all } \begin{pmatrix} J \\ \underline{\lambda} \end{pmatrix} =: \underline{x} \neq 0.$$

Inserting the matrix yields

$$\begin{aligned} \mathbf{M}\underline{x} \cdot \underline{x} &= \left(\frac{1}{a\tau} \mathbf{B}_1^i + \varepsilon^{-1} \mathbf{A}_1^i \right) \underline{J} \cdot \underline{J} + \varepsilon \hat{\mathbf{B}}_0^{-1} \hat{\mathbf{A}}_0^t \underline{J} \cdot \hat{\mathbf{A}}_0^t \underline{J} - 2\varepsilon \hat{\mathbf{B}}_0^{-1} \hat{\mathbf{C}} \underline{\lambda} \cdot \hat{\mathbf{A}}_0^t \underline{J} + \varepsilon \hat{\mathbf{B}}_0^{-1} \hat{\mathbf{A}}_0^t \underline{J} \cdot \hat{\mathbf{C}} \underline{\lambda} \\ &= \left(\frac{1}{a\tau} \mathbf{B}_1^i + \varepsilon^{-1} \mathbf{A}_1^i \right) \underline{J} \cdot \underline{J} + \varepsilon \hat{\mathbf{B}}_0^{-1} (\hat{\mathbf{A}}_0^t \underline{J} - \hat{\mathbf{C}} \underline{\lambda}) \cdot (\hat{\mathbf{A}}_0^t \underline{J} - \hat{\mathbf{C}} \underline{\lambda}) \\ &= \left(\frac{1}{a\tau} \mathbf{B}_1^i + \varepsilon^{-1} \mathbf{A}_1^i \right) \underline{J} \cdot \underline{J} + \varepsilon \hat{\mathbf{B}}_0^{-1} \underline{z} \cdot \underline{z} \end{aligned}$$

with $\underline{z} = \hat{\mathbf{B}}_0^{-1} (\hat{\mathbf{A}}_0^t \underline{J} - \hat{\mathbf{C}} \underline{\lambda})$. It is easier to see how the result follows if we leave the matrix notation. Therefore, setting

$$J_h = \sum_{i=0}^{N_E-1} J_i \Psi_i, \quad \lambda_h = \sum_{i=0}^{N_E^0-1} \lambda_i \zeta_i \quad \text{and} \quad z_h = \sum_{s=0}^{3N_T-1} z_s \hat{\Psi}_s,$$

we get

$$\mathbf{M}\underline{x} \cdot \underline{x} = \frac{1}{a\tau} \int_{\Omega} \frac{1}{M(\phi_h^i)} |J_h|^2 + \varepsilon^{-1} \int_{\Omega} G''(\phi_h^i) (\nabla \cdot J_h)^2 + \varepsilon \int_{\Omega} |z_h|^2. \quad (4.39)$$

In the same way we get by the definition of \underline{z}

$$\begin{aligned} & \hat{\mathbf{A}}_0^t \underline{J} - \hat{\mathbf{C}} \underline{\lambda} = \hat{\mathbf{B}}_0 \underline{z} \\ \Leftrightarrow & \int_{\Omega} (\nabla \cdot \hat{\Psi}_s) (\nabla \cdot J_h) - \sum_{T \in \mathcal{T}_h} \int_{\partial T} \lambda_h \hat{\Psi}_s \cdot \nu_T = \int_{\Omega} \hat{\Psi}_s \cdot z_h \quad s = 0, \dots, 3N_T - 1 \\ \Leftrightarrow & \int_{\Omega} (\nabla \cdot \hat{K}_h) (\nabla \cdot J_h) - \sum_{T \in \mathcal{T}_h} \int_{\partial T} \lambda_h \hat{K}_h \cdot \nu_T = \int_{\Omega} \hat{K} \cdot z_h \quad \forall \hat{K}_h \in \mathcal{RT}_{-1}(\mathcal{T}_h). \end{aligned}$$

Now choosing $\hat{K} := J_h \in \mathcal{RT}_0(\mathcal{E}_h) \subset \mathcal{RT}_{-1}(\mathcal{T}_h)$, the second term on the left-hand side vanishes and we find

$$\int_{\Omega} (\nabla \cdot J_h)^2 = \int_{\Omega} J_h \cdot z_h. \quad (4.40)$$

Using $0 \leq M(\phi) \leq 1$ and $G(\phi) \geq -\beta$ in (4.39), we get

$$\begin{aligned} \mathbf{M}\underline{x} \cdot \underline{x} &\geq \frac{1}{a\tau} \int_{\Omega} |J_h|^2 - \frac{\beta}{\varepsilon} \int_{\Omega} (\nabla \cdot J_h)^2 + \varepsilon \int_{\Omega} |z_h|^2 \\ &= \frac{1}{a\tau} \int_{\Omega} |J_h|^2 - \frac{\beta}{\varepsilon} \int_{\Omega} J_h \cdot z_h + \varepsilon \int_{\Omega} |z_h|^2. \end{aligned}$$

In the case $z_h \equiv 0$, we have $J_h \not\equiv 0$ and therefore \mathbf{M} is positive definite for any $\tau > 0$. Indeed, if also $J_h \equiv 0$, then by the definition of \underline{z} we have

$$\hat{\mathbf{C}} \underline{\lambda} = 0 \iff c(\lambda_h, \hat{K}_h) = 0 \quad \forall \hat{K}_h \in \mathcal{RT}_{-1}(\mathcal{T}_h).$$

In the proof of Proposition 4.27.②, we have seen that this implies $\lambda_h \equiv 0$. But J_h and λ_h cannot both be zero.

In the case $z_h \neq 0$, we get

$$\begin{aligned} \mathbf{M}\underline{x} \cdot \underline{x} &\geq \int_{\Omega} \left| \frac{1}{\sqrt{a\tau}} J_h - \frac{\beta\sqrt{a\tau}}{2\varepsilon} z_h \right|^2 - \frac{\beta^2 a\tau}{4\varepsilon^2} \int_{\Omega} |z_h|^2 + \varepsilon \int_{\Omega} |z_h|^2 \\ &\geq \left(\varepsilon - \frac{\beta^2 a\tau}{4\varepsilon^2} \right) \int_{\Omega} |z_h|^2 \end{aligned}$$

and finally this is positive if

$$a\tau < \frac{4\varepsilon^3}{\beta^2}.$$

□

Remark 4.30. This is the same constraint as in (3.12). The simulations suggest that in the case of developed interfaces the restriction is much less severe, maybe more as in Lemma 3.8. ◇

4.8.4 Solution of the split equation

Another way to solve the fully discrete equations is to use the split version

$$\begin{pmatrix} -\varepsilon \mathbf{B}_0 & \varepsilon \mathbf{A}_0 \\ \varepsilon \mathbf{A}_0 & \frac{1}{a\tau} \mathbf{B}_1^i + \varepsilon^{-1} \mathbf{A}_1^i \end{pmatrix} \begin{pmatrix} \underline{K} \\ \underline{J} \end{pmatrix} = \begin{pmatrix} \underline{0} \\ -\underline{\mathbf{G}}r \end{pmatrix}, \quad (4.41)$$

and solve this system for $(\underline{K}, \underline{J})$. We show below that this system is indefinite. As we have seen in the previous section, we can get a system with the same number of rows which is symmetric positive definite. Therefore, we will not consider the method of solving the split equation any further.

To see that the system is indefinite, we set $M(\phi) \equiv 1$ and assume $G''(\phi) \equiv -\beta$ for simplicity. Then, $\mathbf{B}_1^i = \mathbf{B}_0$ and $\mathbf{A}_1^i = -\beta \mathbf{A}_0$ and we get

$$\begin{pmatrix} -\varepsilon \mathbf{B}_0 & \varepsilon \mathbf{A}_0 \\ \varepsilon \mathbf{A}_0 & \frac{1}{a\tau} \mathbf{B}_1^i + \varepsilon^{-1} \mathbf{A}_1^i \end{pmatrix} \begin{pmatrix} \underline{K} \\ \underline{J} \end{pmatrix} \cdot \begin{pmatrix} \underline{K} \\ \underline{J} \end{pmatrix} = -\varepsilon \mathbf{B}_0 \underline{K} \cdot \underline{K} + 2\mathbf{A}_0 \underline{K} \cdot \underline{J} + \frac{1}{a\tau} \mathbf{B}_0 \underline{J} \cdot \underline{J} - \beta \mathbf{A}_0 \underline{J} \cdot \underline{J}.$$

On the one hand, setting $\underline{K} = 0$ yields

$$\frac{1}{a\tau} \mathbf{B}_0 \underline{J} \cdot \underline{J} - \mathbf{A}_0 \underline{J} \cdot \underline{J},$$

which is positive for $0 < a\tau < (\mathbf{B}_0 \underline{J} \cdot \underline{J}) / (\mathbf{A}_0 \underline{J} \cdot \underline{J})$. On the other hand, setting $\underline{J} = 0$ yields

$$-\varepsilon \mathbf{B}_0 \underline{K} \cdot \underline{K},$$

which is negative. Therefore, system (4.41) is indefinite.

4.8.5 Piecewise constant energy

All previous methods were based on the discrete gradient defined by duality. Remember that we introduced the discrete gradient to be able to use the lowest-order Raviart–Thomas finite elements. The term which made the discrete gradient necessary was the fourth-order term in

$$\begin{aligned} \text{Hess } E(\phi)(\nabla \cdot J, \nabla \cdot \tilde{J}) &= \int_{\Omega} \frac{\delta^2 E}{\delta \phi^2}(\phi)(\nabla \cdot J)(\nabla \cdot \tilde{J}) \\ &= -\varepsilon \int_{\Omega} \Delta(\nabla \cdot J)(\nabla \cdot \tilde{J}) + \varepsilon^{-1} \int_{\Omega} G''(\phi)(\nabla \cdot J)(\nabla \cdot \tilde{J}). \end{aligned}$$

If we keep using the space $\mathcal{RT}_0(\mathcal{E}_h)$ for J_h and \tilde{J}_h , we know that $z_h := \nabla \cdot J_h$ and $\tilde{z}_h := \nabla \cdot \tilde{J}_h$ are piecewise constant and we have to deal with the term

$$- \int_{\Omega} \Delta_h z_h \tilde{z}_h = \int_{\Omega} \nabla_h z_h \cdot \nabla_h \tilde{z}_h.$$

The discrete Laplacian was up to now defined through first applying the discrete gradient and then the (usual continuous) divergence.

$$\mathcal{L}_0(\mathcal{T}_h) \xrightarrow{\nabla_h} \mathcal{RT}_0(\mathcal{E}_h) \xrightarrow{\nabla \cdot} \mathcal{L}_0(\mathcal{T}_h).$$

The idea for the method in this section is to skip the detour over $\mathcal{RT}_0(\mathcal{E}_h)$ and to define a discrete Laplacian

$$\Delta_h : \mathcal{L}_0(\mathcal{T}_h) \rightarrow \mathcal{L}_0(\mathcal{T}_h)$$

or, more generally, a discrete operator

$$\left(\frac{\delta^2 E}{\delta \phi^2}(\phi) \right)_h : \mathcal{L}_0(\mathcal{T}_h) \rightarrow \mathcal{L}_0(\mathcal{T}_h).$$

We proceed as follows: first, we define a piecewise constant energy $E_h(\phi_h)$ and then compute the variational derivatives. Since the energy is

$$E(\phi) = \int_{\Omega} \frac{\varepsilon}{2} |\nabla \phi|^2 + \varepsilon^{-1} G(\phi),$$

we construct a local gradient operator

$$\nabla_h : \mathcal{L}_0(\mathcal{T}_h) \rightarrow \mathcal{L}_0(\mathcal{T}_h)^2.$$

Before we do this, we show how to set up the fourth-order term, the right-hand side and the chemical potential given such a gradient operator.

Assume therefore that we have two $N_T \times N_T$ matrices \mathbf{dx} and \mathbf{dy} such that

$$\nabla_h \phi_h|_{T_k} = \begin{pmatrix} \mathbf{dx}_k \phi \\ \mathbf{dy}_k \phi \end{pmatrix}$$

with the notation that \mathbf{dx}_k is the k -th row of \mathbf{dx} . We only consider the Dirichlet part of the energy

$$E_h(\phi_h) = \int_{\Omega} \frac{1}{2} |\nabla_h \phi_h|^2$$

for the derivation of the second variation of the energy since $G(\phi_h)$ is already piecewise constant. The matrix representation of the fourth-order operator is given by

$$\begin{aligned} \mathbf{A}_{ij}^{\text{FOT}} &= E_h''(\phi_h)(\nabla \cdot \Psi_i, \nabla \cdot \Psi_j) =: E_h''(\phi_h)(z_h^i, z_h^j) = \int_{\Omega} \nabla_h z_h^i \cdot \nabla_h z_h^j \\ &= \sum_{k=0}^{N_T-1} \int_{T_k} (\mathbf{dx}_k z^i \mathbf{dx}_k z^j + \mathbf{dy}_k z^i \mathbf{dy}_k z^j). \end{aligned}$$

Note that \mathbf{A}^{FOT} is independent of ϕ , so it has only to be reassembled when the mesh changes. The matrix representation (4.24) then becomes

$$\left(\frac{1}{a\tau} \mathbf{B}_1^k + \varepsilon \mathbf{A}^{\text{FOT}} + \varepsilon^{-1} \mathbf{A}_1^i \right) \underline{J}^* = -\underline{\mathbf{G}}_L.$$

The right hand side is given by

$$r_h = \frac{1}{a\tau} w_h^* + \frac{\delta E_h}{\delta \phi_h}(\phi_h^i) u_h^i$$

and the chemical potential is

$$w_h^* = \frac{\delta E_h}{\delta \phi_h}(\phi_h^i) + \gamma \frac{\delta E_h}{\delta \phi_h}(\phi_h^k).$$

Remember from Definition 2.2 that the variational derivative of E_h is defined through

$$\text{diff } E(\phi)\psi = \int_{\Omega} \frac{\delta E}{\delta \phi}(\phi)\psi \quad \forall \psi.$$

For the discrete case, we get

$$\begin{aligned} \text{diff } E_h(\phi_h)\psi_h &= \frac{d}{d\delta} \Big|_{\delta=0} E_h(\phi_h + \delta\psi_h) = \int_{\Omega} \nabla_h \phi_h \cdot \nabla_h \psi_h = \sum_{k=0}^{N_T-1} \int_{T_k} \nabla_h \phi_h \cdot \nabla_h \psi_h \\ &= \sum_{k=0}^{N_T-1} \int_{T_k} \mathbf{dx}_k \phi \mathbf{dx}_k \psi + \mathbf{dy}_k \phi \mathbf{dy}_k \psi \end{aligned}$$

$$\begin{aligned}
&= \sum_{k=0}^{N_T-1} |T_k| \left(\mathbf{dx}_k \underline{\phi} \sum_{l=0}^{N_T-1} (\mathbf{dx}_{kl} \underline{\psi}_l) + \mathbf{dy}_k \underline{\phi} \sum_{l=0}^{N_T-1} (\mathbf{dy}_{kl} \underline{\psi}_l) \right) \\
&= \sum_{l=0}^{N_T-1} \sum_{k=0}^{N_T-1} \left(\mathbf{dx}_{lk}^t |T_k| \mathbf{dx}_k \underline{\phi} + \mathbf{dy}_{lk}^t |T_k| \mathbf{dy}_k \underline{\phi} \right) \underline{\psi}_l \\
&= \sum_{l=0}^{N_T-1} \int_{T_l} \left(\frac{1}{|T_l|} (\mathbf{dx}^t \mathbf{Vdx} \underline{\phi})_l + \frac{1}{|T_l|} (\mathbf{dy}^t \mathbf{Vdy} \underline{\phi})_l \right) \underline{\psi}_l \\
&=: \int_{\Omega} \frac{\delta E_h}{\delta \phi_h}(\phi_h) \psi_h.
\end{aligned}$$

Since in the continuous case, the variational derivative of the Dirichlet energy is the Laplacian of ϕ , $\delta E(\phi)/\delta \phi = -\Delta \phi$, it is natural to define the discrete Laplacian as

$$-\Delta_h \phi_h|_{T_l} := \frac{\delta E_h}{\delta \phi_h}(\phi_h)|_{T_l} = \left(\mathbb{T} \underline{\phi} \right)_l \quad \text{with} \quad \mathbb{T} = \mathbf{V}^{-1} \mathbf{dx}^t \mathbf{Vdx} + \mathbf{V}^{-1} \mathbf{dy}^t \mathbf{Vdy}. \quad (4.42)$$

Remark 4.31. Although \mathbb{T} is not symmetric as a matrix, it is symmetric with respect to the scalar product introduced in Section 4.7:

$$\langle \mathbb{T} \underline{u}, \underline{v} \rangle_{\mathcal{T}_h} = (\mathbf{dx}^t \mathbf{Vdx} + \mathbf{dy}^t \mathbf{Vdy}) \underline{u} \cdot \underline{v} = \underline{u} \cdot (\mathbf{dx}^t \mathbf{Vdx} + \mathbf{dy}^t \mathbf{Vdy}) \underline{v} = \langle \underline{u}, \mathbb{T} \underline{v} \rangle_{\mathcal{T}_h}.$$

◇

We now come to the construction of an appropriate gradient operator. The minimal requirements for such an operator are that the gradient is

- *linear.*
- *local* in the sense that only the neighbours of a triangle contribute to the gradient.
- *exact for linear functions.* That is, if $\phi_h \in \mathcal{L}_0(\mathcal{T}_h)$ is the restriction of an affine function, then the gradient should be exact. This is a little less than for the discrete gradient defined via duality, see Corollary 4.15.

If we speak of neighbours of a triangle T in the following, we always mean those triangles sharing an edge with T . So in particular, T is a neighbour of itself and so every triangle has four neighbours.

Gradient via least squares approximation

A first and seemingly natural choice was to define the discrete gradient via a least squares construction: on each triangle T_k , we search for a linear function

$$L^{(k)}(x) = \mathbf{g}^{(k)} \cdot x + c^{(k)}$$

which approximates ϕ_h on the neighbouring triangles in a way such that the mass difference is minimal in the sense of least squares, see Figure 4.3 for an example.

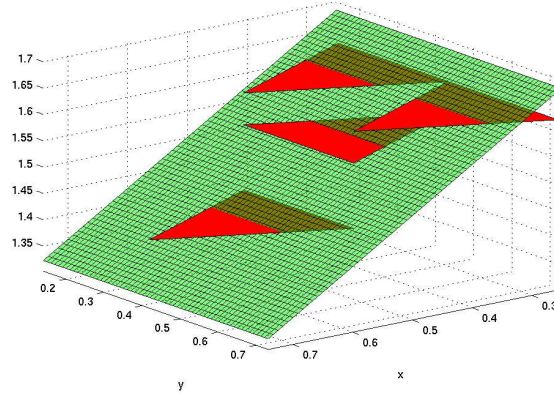


Figure 4.3: Approximation of the piecewise constant function ϕ_h (red) with an affine function (green). The gradient on the central triangle is set to the gradient of the approximation.

Switching to local triangle numbering, we denote T_k with T_0 and the neighbours with T_1 to T_3 . Then we have to solve

$$\sum_{l=0}^3 \left(\int_{T_l} L - \int_{T_l} \phi_h \right)^2 = \min!$$

Denoting with $(\underline{M}x_l, \underline{M}y_l)$ the centre of triangle T_l and with (g_x, g_y) the components of g , we get

$$\begin{aligned} & \sum_{l=0}^3 \left(|T_l| \begin{pmatrix} \underline{M}x_l & \underline{M}y_l & 1 \end{pmatrix} \begin{pmatrix} g_x \\ g_y \\ c \end{pmatrix} - |T_l| \underline{\phi}_l \right)^2 \\ = & \left\| \begin{pmatrix} |T_0| & & 0 \\ & \ddots & \\ 0 & & |T_3| \end{pmatrix} \begin{pmatrix} \underline{M}x_0 & \underline{M}y_0 & 1 \\ & \vdots & \\ \underline{M}x_3 & \underline{M}y_3 & 1 \end{pmatrix} \begin{pmatrix} g_x \\ g_y \\ c \end{pmatrix} - \begin{pmatrix} |T_0| & & 0 \\ & \ddots & \\ 0 & & |T_3| \end{pmatrix} \begin{pmatrix} \underline{\phi}_0 \\ \vdots \\ \underline{\phi}_3 \end{pmatrix} \right\|^2 \end{aligned}$$

which we write as

$$\| \mathbf{V}^{(k)} \mathbf{S}^{(k)} \underline{g}^{(k)} - \mathbf{V}^{(k)} \underline{\phi}^{(k)} \|^2$$

with obvious definitions. Solving the normal equations yields

$$\underline{g}^{(k)} = \left((\mathbf{V}^{(k)} \mathbf{S}^{(k)})^t \mathbf{V}^{(k)} \mathbf{S}^{(k)} \right)^{-1} (\mathbf{V}^{(k)} \mathbf{S}^{(k)})^t \mathbf{V}^{(k)} \underline{\phi}^{(k)}.$$

Then, we set

$$\nabla_h \phi_h|_{T_k} := \begin{pmatrix} g_x^{(k)} \\ g_y^{(k)} \end{pmatrix}.$$

Clearly, if ϕ_h is the restriction of an affine function, then $L^{(k)}$ coincides with this function and the gradient is exact. Additionally, by construction, $L^{(k)}$ is local and linear, so all our requirements are met.

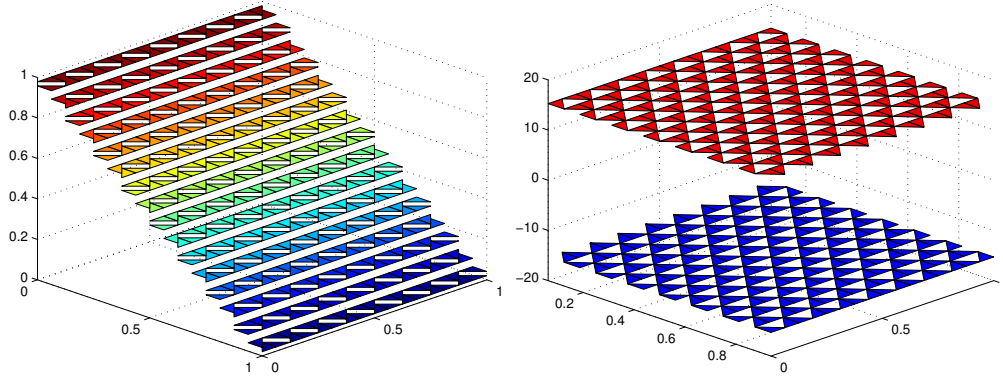


Figure 4.4: Left: Example of a piecewise constant function coming from a linear function. Right: The Laplacian of this function in the case where the piecewise constant gradients are defined via least squares. The values of $\Delta_h f_h$ are ± 15.27 , except for the boundary where f_h jumps; these values are not shown.

Unfortunately, the results are unsatisfactory. If a discrete function is the restriction of an affine function, then the Laplacian is *not* zero for all meshes, although the gradient is exact by construction. See Figure 4.4 for an example on a rather simple mesh.

Gradient via conditions on the Laplacian

Since the least-squares method failed, we look for another gradient. Of course, we want to keep the requirements of *locality* and *exactness for linear functions*. Locality means that we have $4N_T$ degrees of freedom for each of \mathbf{dx} and \mathbf{dy} . Being exact for affine functions means that if

$$u(x) = a \cdot x + b \implies \underline{u} = a_1 \underline{Mx} + a_2 \underline{My} + b \underline{1}, \quad (4.43)$$

where $a_1, a_2, b \in \mathbb{R}$ and $(\underline{Mx}_k, \underline{My}_k)$ is the centre of T_k , then we want

$$\mathbf{dx} \underline{u} = a_1 \underline{1} \quad \text{and} \quad \mathbf{dy} \underline{u} = a_2 \underline{1}.$$

This yields for each of \mathbf{dx} and \mathbf{dy} the $3N_T$ conditions

$$\mathbf{dx} \underline{Mx} = \underline{1} \quad \mathbf{dy} \underline{Mx} = 0, \quad (4.44a)$$

$$\mathbf{dx} \underline{My} = 0 \quad \mathbf{dy} \underline{My} = \underline{1}, \quad (4.44b)$$

$$\mathbf{dx} \underline{1} = 0 \quad \mathbf{dy} \underline{1} = 0. \quad (4.44c)$$

From the experience with the least-squares gradient, we add another condition, namely the Laplacian should be *zero for linear functions*. So for the \underline{u} from (4.43), we require

$$\nabla \underline{u} = 0 \iff \nabla \underline{Mx} = 0, \nabla \underline{My} = 0, \nabla \underline{1} = 0.$$

Since we already know that $\mathbf{dx} \underline{u} = a_1 \underline{1}$ and $\mathbf{dy} \underline{u} = a_2 \underline{1}$, we get from (4.42)

$$\mathbf{dx}^t \mathbf{V} \underline{1} = 0 \quad \mathbf{dy}^t \mathbf{V} \underline{1} = 0, \quad (4.44d)$$

which adds N_T conditions for each of \mathbf{dx} and \mathbf{dy} , yielding a total of $4N_T$ conditions.

However, there is one degree of freedom left per matrix. Therefore, we get a family of solutions:

$$\mathbf{dx} + \lambda_x \mathbf{zx} \quad \text{and} \quad \mathbf{dy} + \lambda_y \mathbf{zy}, \quad (4.45)$$

see Section A.9 on how to compute \mathbf{zx} and \mathbf{zy} .

To select one of these gradients, take a look at Figure 4.5, where we plotted the x_2 -derivative and the Laplacian of $f(x) = x_1(1 - x_1)$. The Laplacian is only exact for one specific λ_y . We cannot expect an exact Laplacian for all meshes and all quadratic functions. Our first idea was therefore to minimize the error of the Laplacian for quadratic functions, see Remark 4.32. Indeed, the results were good. However, there is another method yielding the same results while being much faster: In the left column of Figure 4.5, one can see that the x_2 -derivative,

$$(\mathbf{dy} + \lambda_y \mathbf{zy}) \underline{f},$$

of a function only depending on x_1 “drifts away” from zero for the “wrong” λ_s .

This serves us as a condition to determine λ_y . As all members of the family of gradients are exact for linear functions, we get as an additional requirement

$$0 = (\mathbf{dy} + \lambda_y \mathbf{zy}) \underline{f} = \frac{1}{2} a_2 (\mathbf{dy} + \lambda_y \mathbf{zy}) \underline{s}^{11} \quad \text{with} \quad \underline{s}_k^{ij} = \frac{1}{|T_k|} \int_{T_k} x_i x_j \, dx.$$

Since we have not enough degrees of freedom left to require the above to be zero on every triangle, we minimize the L^2 -norm of the discrete version of $\partial_y f$ to determine λ_x and λ_y :

$$\begin{aligned} \frac{2}{a_2} \int_{\Omega} \left| \partial_y^h f_h(x) \right|^2 &= \sum_{k=0}^{N_T-1} \int_{T_k} \left((\mathbf{dy} + \lambda_y \mathbf{zy}) \underline{s}^{11} \right)_k^2 = \sum_{k=0}^{N_T-1} |T_k| \left((\mathbf{dy} \underline{s}^{11})_k + \lambda_y (\mathbf{zy} \underline{s}^{11})_k \right)^2 \\ &= \lambda_y^2 \underbrace{\left(\sum_{k=0}^{N_T-1} |T_k| (\mathbf{dy} \underline{s}^{11})_k^2 \right)}_{=:c_2} + \lambda_y \underbrace{\left(2 \sum_{k=0}^{N_T-1} |T_k| (\mathbf{dy} \underline{s}^{11})_k (\mathbf{zy} \underline{s}^{11})_k \right)}_{=:c_1} + \underbrace{\left(\sum_{k=0}^{N_T-1} |T_k| (\mathbf{zy} \underline{s}^{11})_k^2 \right)}_{=:c_0}. \end{aligned}$$

The optimality conditions for minimizing in λ_y yield

$$\lambda_y = -\frac{c_1}{2c_2}. \quad (4.46)$$

The calculations for λ_x are analogous.

Remark 4.32. Another method to determine $\lambda_{x/y}$ is to consider a quadratic function

$$f(x) = \begin{pmatrix} q_{11} & \frac{1}{2}q_{12} \\ \frac{1}{2}q_{12} & q_{22} \end{pmatrix} x \cdot x + a \cdot x + b$$

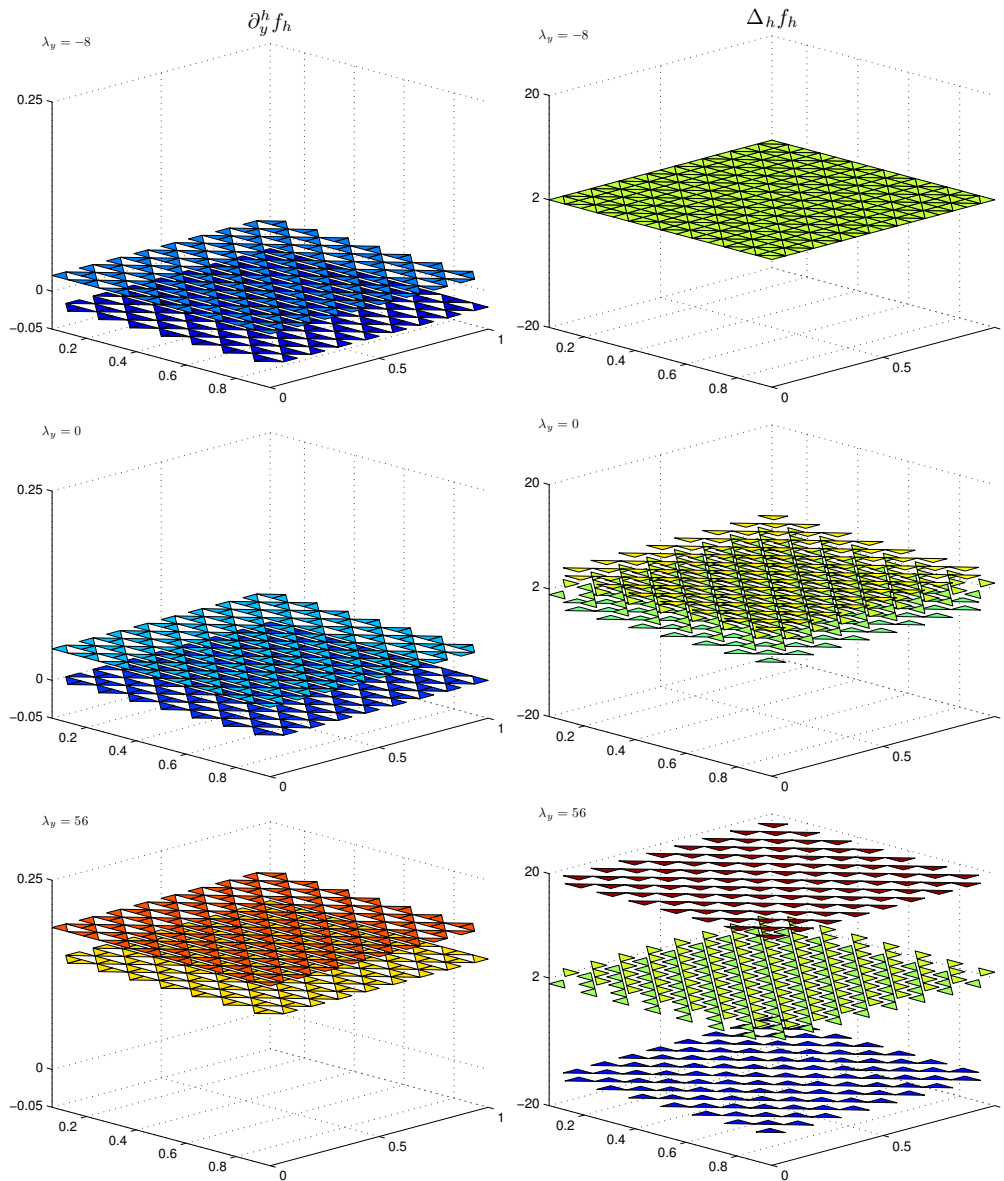


Figure 4.5: Discrete gradient in x_2 -direction and minus the discrete Laplacian of the function $f(x_1, x_2) = x_1(1 - x_1)$. The top row uses the optimal $\lambda_y = -8$ and the rows below use the arbitrarily chosen values $\lambda_y = 0$ and $\lambda_y = 56$.

and to estimate the L^2 -norm of the error that is made when taking the Laplacian. To keep the formulas shorter, we use here the Euclidian norm instead of the L^2 -norm and save some V 's. This yields

$$\begin{aligned} & \|q_{11} \mathbb{T}_s^{11} + q_{12} \mathbb{T}_s^{12} q_{22} \mathbb{T}_s^{22} - q_{11} \mathbb{1} - q_{22} \mathbb{1}\| \\ & \leq \| \mathbb{T} \| \left(|q_{11}| \|\underline{s}^{11}\| + |q_{12}| \|\underline{s}^{12}\| + |q_{22}| \|\underline{s}^{22}\| \right) + \sqrt{N} \left(|q_{11}| + |q_{22}| \right) \end{aligned}$$

with any matrix norm $\| \cdot \|$. To minimize the error, we have to calculate

$$\min_{\lambda_x, \lambda_y} \| \mathbb{T} \|_F,$$

where we chose the Frobenius norm as matrix norm. Tedious calculations lead to a minimization problem of the form

$$\sum_{i=0}^4 \sum_{j=0}^4 \gamma_{ij} \lambda_x^i \lambda_y^j, \quad (4.47)$$

where the coefficients $\gamma_{ij} = 0$ if $i + j > 4$. To get an impression of the coefficients, we show here one of them:

$$\gamma_{10} = b_{21}(c_{11} + c_{12}) + b_{11}(c_{21} + c_{22}),$$

where

$$\begin{aligned} b_{i1} &= \sum_{l,m} \frac{1}{|T_l|^2} \sum_{k_1, k_2} (\mathbf{d}\mathbf{x}_{k_i l} \mathbf{z}\mathbf{x}_{k_i m} + \mathbf{z}\mathbf{x}_{k_i l} \mathbf{d}\mathbf{x}_{k_i m}), \\ c_{i1} &= \sum_{l,m} \frac{1}{|T_l|^2} \sum_{k_1, k_2} \mathbf{d}\mathbf{x}_{k_i l} \mathbf{d}\mathbf{x}_{k_i m}, \\ c_{i2} &= \sum_{l,m} \frac{1}{|T_l|^2} \sum_{k_1, k_2} \mathbf{d}\mathbf{y}_{k_i l} \mathbf{d}\mathbf{y}_{k_i m}. \end{aligned}$$

Equation (4.47) was then solved using a globalized Newton scheme similar to the one in Section A.2. However, for all meshes tested, the resulting values for $\lambda_{x/y}$ were the same as when using (4.46). Since assembling the coefficients and solving the minimization is slow, we use (4.46). \diamond

4.9 Adaptivity

We already mentioned several times the special form of solutions to Cahn–Hilliard-type equations in the interfacial regime: The domain Ω can be divided into bulk and interface regions, where the latter usually takes up only a small fraction of the domain.

Having this in mind, we want the mesh to have the following structure:

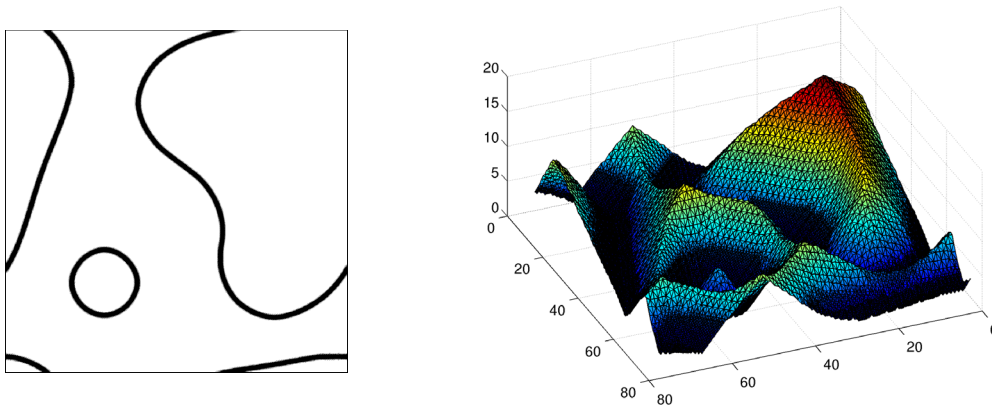


Figure 4.6: For the interfaces shown on the left, the solution of the eikonal equation yields the distance of the vertices to the interface, see the right plot.

- At the interfaces, the mesh should be fine enough to resolve the interface profile, see Figure 2.6 on page 23 for the width of the interface. Therefore, we set the mesh spacing to a constant (ε dependent) value near the interfaces. The necessary spatial resolution at the interface is investigated in Section 4.9.1.
- Away from the interfaces, we make use of the fact that the considered Cahn–Hilliard-type equations have sharp-interface limits. In these free boundary problems, the equation to be solved in the bulk is simply $-\Delta w = f$, see Section 2.6.2. Therefore, we coarsen geometrically in the distance from the interfaces.

To obtain the distance from the interfaces, we first mark the interface edges, i.e. in the case of epitaxial growth (EG) those edges with $\phi \geq 1/2$ on one neighbouring triangle and $\phi \leq 1/2$ on the other. Then we solve the eikonal equation using an algorithm from Bornemann and Rasch [2006], which gives us the distance at the vertices, see Figure 4.6. The distance of a triangle is set to the minimum of the distances of its vertices. Then we mark triangles for refinement and coarsening according to the above principles. See Figure 4.7 for an example.

If the mesh has been changed, all matrices have to be reassembled. Therefore, we try to keep the number of mesh adaptations small by taking the following measures:

- If the mesh has to be refined (i.e. triangles have been marked), make the region around the interfaces bigger and mark again. This increases the number of triangles to be refined (and thus the number of degrees of freedom), but reduces the number of mesh refinements. Especially, the situation where a small number of triangles is refined in each time step is avoided.
- Only coarsen the mesh if the number of triangles to be coarsened is large enough, say two percent of the number of degrees of freedom.

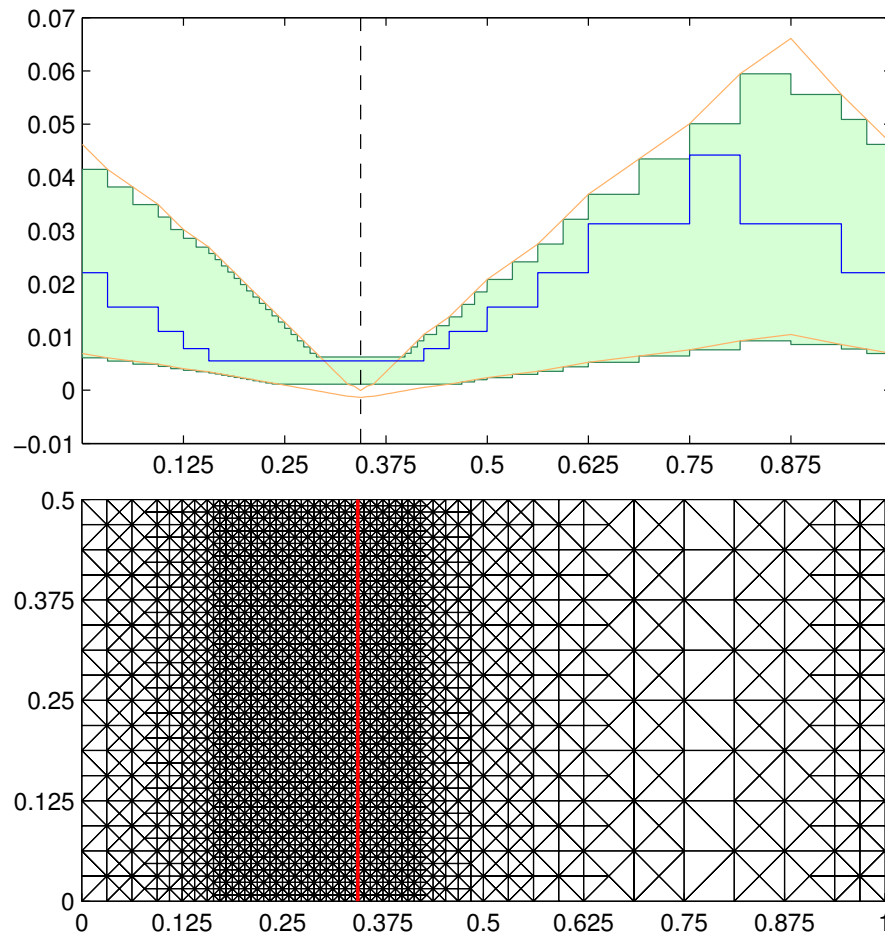


Figure 4.7: Space adaptivity constraints for a straight step moving from left to right. We shift and scale the computed distance function in two different ways to prescribe an upper and lower bound for the mesh size h at the vertices, see the orange curves. Transferred to the triangles, this yields the green corridor. The value of h is plotted in blue. The position of the step is marked with the dashed line in the upper picture and with the red line in the mesh.

- Of course, we prevent that a coarsened triangle is refined again in the next step, see Section 7.2.3.

For refining a triangle, we use longest-edge bisection. Coarsening would be more properly named “unrefinement”, since only triangles once refined can be coarsened. This implies that we set an upper bound on the size of the triangles with the choice of the initial mesh.

Remark 4.33. More generally speaking, the type of refinement procedure described above is valid in all situations where the evolution of ϕ emerges from a specific region of the domain. In our examples, this region consists of the interfaces.

In the case of the thin-film equation, a similar method is appropriate. Typical configurations consist of well-separated droplets connected by a thin flat precursor layer.

The important region for the dynamics is the transition region between precursor layer and droplet, whose width is small compared to the typical droplet sizes. In order to recover the dynamics, it is necessary that the mesh in this region is fine enough. See Rump [2008, Chapter 6] for more information on the numerics of the thin-film equation. \diamond

4.9.1 Resolution at the interface

As a measure for the size of a triangle, we use half the diameter of the triangle:

$$h_T = \frac{1}{2} \text{diam}(T).$$

As described above, the mesh spacing is set to a constant near the interface. This is done by defining a quantity h_{\min} and refining all triangles in the interfacial region for which

$$h_T > h_{\min}. \quad (4.48)$$

We investigate in this section how to choose h_{\min} , in particular how large we can set h_{\min} and still get correct results. All computations are made using $\varepsilon = 1$. The results can be transferred to the case $\varepsilon \neq 1$ by rescaling space and time accordingly.

The first test is as follows: we start with initial data in the shape of an octagon in a square periodic region of size $L = 73.4$, such that, using the “cross” grid from Figure 4.12, the faces of the octagon lie on the grid lines. This seems to be a good test to disclose problems. Letting evolve the initial step, it should eventually converge to the steady state of a circle, see Section 2.6.2. Therefore we stopped the simulation when a steady state was reached; more specifically we used

$$\max(w) - \min(w) < 10^{-9}$$

as an indicator that a steady state was reached and then computed the maximal flux to confirm the steady state. To check if the steady-state solution is a circle, we compute the step position as lined out in Section A.3.3. This provides us with points s_i lying approximately on

the interface. Then, we measure the distance from the centre c to the points to get the radii $r_i := |s_i - c|$ and compute the average deviation

$$\delta r := \frac{1}{N} \sum_{i=1}^N |r_i - \bar{r}| \quad \text{with} \quad \bar{r} = \frac{1}{N} \sum_{i=1}^N r_i.$$

Due to the difficulties with determining the step position, see Section A.3, we allow for a deviation of up to half the diameter of a triangle from a perfect circle. Therefore, if the value of $\delta r/h$ is smaller than one, we consider the state to be a circle. For a perfect circle, this value is obviously zero. For the octagonal initial data we get $\delta r/h = 2.52$, where $h = 0.2353$. The results are shown in Table 4.2.

We see that for any method, a value of $h_{\min} = 0.25$ is too large and the final state is *not* a circle. In the case of ABM, we see a clear transition in the behaviour and find good results for values of $h_{\min} \leq 0.2$, see Figure 4.8. Apparently, the PCE method needs a finer mesh to produce comparable results, but the results improve as h decreases. The results for SPASI are less clear.

In the cases where we were content with the results, we performed another test. It turned out that the most delicate quantity is the curvature of the interface, so we concentrate on this quantity. As we have seen in Section 2.6.2, we should have

$$w = \sqrt{\frac{10}{3}} \kappa \tag{4.49}$$

at the interface in case of the shallow-quench equation.

To get κ , we make use of the fact that the mass is constant during the simulation and that the solution converges to a circle with a known profile. Therefore, the steady state is very close to

$$\phi(x) = u_{\text{SQ}}^*(|x - c| - R_{\text{mass}})$$

for some value of R_{mass} , where c is the centre of the unit cell, i.e. $c = (L/2, L/2)$, and u_{SQ}^* is the optimal profile from Section 2.6.3.

To determine R_{mass} and with it $\kappa = 1/R$, we measure the mass M of the initial value (or any other) and then solve

$$M = \int_{[0,L]^2} u_{\text{SQ}}^*(|x - c| - R_{\text{mass}}) \, dx$$

for R_{mass} . This yields a value of $R_{\text{mass}} \approx 27.4825$. Using this method is much more accurate than determining the step position and extracting a radius.

Now we can compare the value of w measured in the simulation (there is only one value since $\max(w) - \min(w) < 10^{-9}$) to the target value w_{goal} ,

$$w_{\text{goal}} := \frac{\sqrt{10/3}}{R_{\text{mass}}}.$$

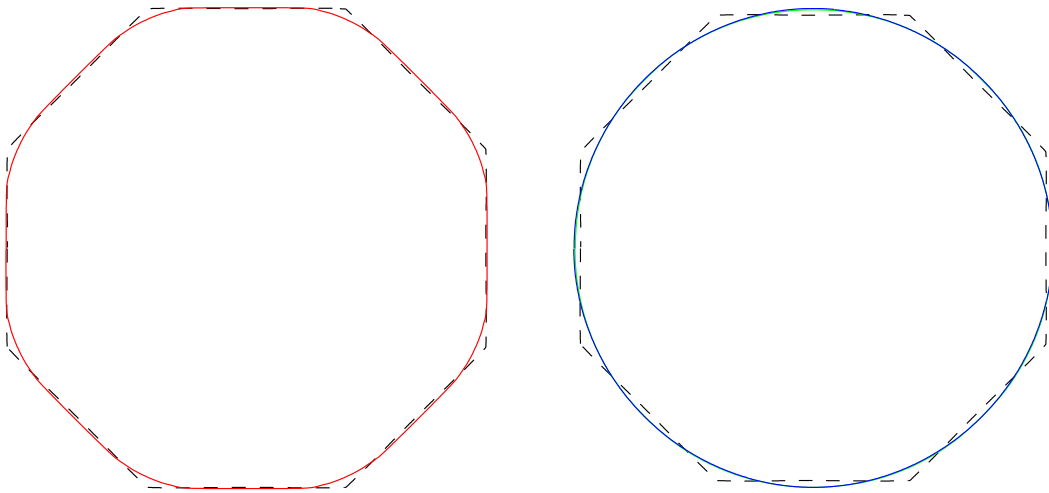
Table 4.2: Agreement of the simulations with the steady-state solutions of the sharp-interface models: Deviation of the radii from a circle.

h_{\min}	$\min_{T \in \mathcal{T}_h} h_T$	δr ($\delta r / \min h_T$)					
		ABM		PCE		SPASI	
0.25	0.2353	0.3153	(1.34)	0.3685	(1.57)	0.3293	(1.40)
0.2	0.1664	0.0524	(0.31)*	0.1896	(1.14)	0.0562	(0.34)
0.15	0.1176	0.0325	(0.28)	0.0952	(0.81)	0.1773	(1.51)*
0.075	0.0589	0.0075	(0.14)	0.0292	(0.50)	0.0552	(0.94)*

In the simulations marked with a star, the centre of the final circle was not the centre of the initial values, hence the deviations. Using the centre of the final data (which is of course not the desired outcome), the values for ABM are 0.0308 (0.19) and for SPASI 0.0530 (0.45) and 0.0388 (0.66).

Table 4.3: Agreement of the simulations with the steady-state solutions of the sharp-interface models: Deviation of the curvature.

h_{\min}	$\min_{T \in \mathcal{T}_h} h_T$	$ w - w_{\text{goal}} / w_{\text{goal}}$		
		ABM	PCE	SPASI
0.2	0.1664	$3.1 \cdot 10^{-2}$	—	$2.3 \cdot 10^{-2}$
0.15	0.1176	$4.6 \cdot 10^{-4}$	$3.0 \cdot 10^{-2}$	—
0.075	0.0589	$1.1 \cdot 10^{-3}$	$1.9 \cdot 10^{-3}$	$2.7 \cdot 10^{-2}$

**Figure 4.8:** On the left, the final state of a simulation with $h_{\min} = 0.25$ is shown. Although a steady state has been reached (flux $< 10^{-15}$), the step is not a circle but sticks to the grid lines. On the right, we see that for $h_{\min} = 0.2$ (green line) and $h_{\min} = 0.15$ (blue line), the final state is a circle.

The results are shown in Table 4.3. They are consistent with the results of first test: ABM produces very good results, PCE needs finer meshes to reach the same accuracy and SPASI is not satisfactory.

From the results for ABT, we conclude that a value of $\min h_T = 0.1664$ is sufficient. Since the interface has a width of approximately one, this means that about eight triangles across the interface are necessary to reproduce the curvature correctly.

We performed these tests also for different meshes and found that a value of 0.17 is enough in all situations.

4.9.2 Transferring piecewise constant functions to a new mesh

After each mesh adaption, we need to transfer ϕ_h to the new mesh. The essential property of the prolongation- and restriction-operator is mass conservation. Since we refine by longest-edge bisection, we need to distribute the mass of one triangle onto two triangles or vice versa. In the case that two triangles T_1 and T_2 are coarsened (more precisely unrefined, since only triangles once refined can be coarsened again) to a triangle T of double area, the value to be given to the new triangle is uniquely determined by

$$\int_T \phi = \int_{T_1} \phi_1 + \int_{T_2} \phi_2 \iff \phi = \frac{1}{2}\phi_1 + \frac{1}{2}\phi_2.$$

On the other hand, refining a triangle T to T_1 and T_2 , each having half the area of T , leaves us with one degree of freedom, denoted by a :

$$\phi_1 = a, \quad \phi_2 = 2\phi - a. \quad (4.50)$$

To decide how to choose a , we recall that the (continuous) energy is a Lyapunov-functional, see Lemma 2.1. Since we would like to keep this property in the discrete case, we minimize the discrete energy to determine a , see Section 4.9.3. As the energy can only be calculated when the new mesh is complete, e.g. has no more hanging nodes, we have to compute intermediate values on the refined triangles which will serve as starting values for the energy minimization.

The easiest way to set these intermediate values would be to choose $\phi_1 = \phi_2 = \phi$. But we can do better, since we have the gradient. Transferring a Raviart–Thomas vector field $\nabla_h \phi_h$ to a finer mesh is uniquely determined. We just have to take the value

$$\nabla_h \phi_h(x) \cdot \nu_E =: g,$$

where x is some point on the new edge E , as the new degree of freedom (remember this quantity is constant on lines). Given the gradient, we determine an affine function

$$L(x) = \alpha(\nu_E \cdot x) + \beta,$$

where α and β are such that

$$\nabla L(x) \cdot \nu_E = g \quad \text{and} \quad \int_T L = \int_T \phi_h.$$

Finally, the new ϕ_h on the refined triangles T_1 and T_2 is set to

$$\phi_1 := \int_{T_1} L = \phi + g((M_1 - M) \cdot \nu_E) \quad \text{and} \quad \phi_2 := \int_{T_2} L = \phi + g((M_2 - M) \cdot \nu_E).$$

In the case of the piecewise constant energy, the procedure is slightly different since the gradient is piecewise constant. We use $g_i = g$ as prolongation for the gradient. Then, we determine an affine function

$$L(x) = a \cdot x + \beta \quad \text{with} \quad \nabla L = \nabla_h \phi_h(x)|_T \quad \text{and} \quad \int_T L = \int_T \phi_h.$$

Finally, the new ϕ_h on the refined triangles T_1 and T_2 is set to

$$\phi_1 = \int_{T_1} L = \phi + \nabla_h \phi_h(x)|_T \cdot (M_1 - M) \quad \text{and} \quad \phi_2 = \int_{T_2} L = \phi + \nabla_h \phi_h(x)|_T \cdot (M_2 - M).$$

4.9.3 Energy minimization

When refining a triangle by bisection, the distribution of the mass onto the two new triangles is not unique. Therefore, we only generate temporary values for ϕ on the new mesh during mesh refinement as described in the previous section. When the new mesh is complete, we correct these temporary values to minimize the energy. In this section, we explain why the energy does not always decrease in this process. The algorithm we use for minimization can be found in Section A.2.

The discrete version of the energy (4.1) is given by

$$E_h(\phi_h) = \int_{\Omega} \frac{\varepsilon}{2} |\nabla_h \phi_h|^2 + \varepsilon^{-1} G(\phi_h). \quad (4.51)$$

By definition of the discrete gradient, this means

$$E_h(\phi_h) = \int_{\Omega} \frac{\varepsilon}{2} |g_h|^2 + \varepsilon^{-1} G(\phi_h),$$

where $g_h \in \mathcal{RT}_0(\mathcal{E}_h)$ is defined by

$$\int_{\Omega} g_h \cdot \tilde{g}_h = - \int_{\Omega} \phi_h (\nabla \cdot \tilde{g}_h) \quad \forall \tilde{g}_h \in \mathcal{RT}_0(\mathcal{E}_h).$$

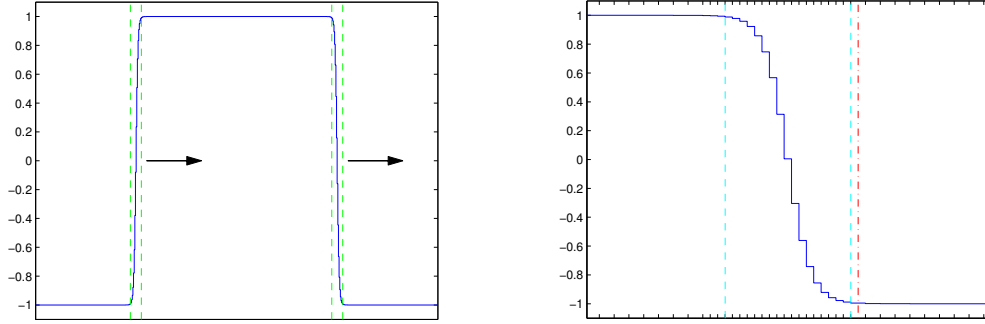


Figure 4.9: One-dimensional example for a refinement. The initial values are moved to the right until the first refinement occurs (left). One triangle will be bisected by the red dot-dashed line (right). The cyan dashed lines show the step width.

Let $\mathcal{L}_0^{\text{ref}}$ denote the space of piecewise constant functions on the refined mesh with the mass conservation restrictions being fulfilled. Minimizing the discrete energy does *not* mean

$$\inf_{\zeta_h \in \mathcal{L}_0^{\text{ref}}} \int_{\Omega} \frac{\varepsilon}{2} |\nabla \zeta_h|^2 + \varepsilon^{-1} G(\zeta_h),$$

since the gradient of a piecewise constant function is not defined. In that case, since the old configuration is still allowed, the energy on the new mesh would be lower or equal to the energy on the old mesh.

Rather, due to the definition of the discrete gradient, minimizing the energy means to find a critical point (g_h, ϕ_h) of

$$\int_{\Omega} \frac{\varepsilon}{2} |\tilde{g}_h|^2 + \int_{\Omega} (\nabla \cdot \tilde{g}_h) \zeta_h - \int_{\Omega} \varepsilon^{-1} G(\zeta_h) \quad (4.52)$$

with the lowest possible energy. Therefore, it is not clear if a mesh refinement leads to a lower or higher energy.

To get an idea of what will happen, let us look at a simple example in one space dimension: consider an interval of length L and take a “hump” composed of two optimal profiles of the shallow quench equation, i.e.

$$\phi(x, \xi) = \tanh(\sqrt{15/2}(x - \xi - L/4)) - \tanh(\sqrt{15/2}(x - \xi - 3L/4)) - 1.$$

At $t = 0$, we set $\xi = 0$, use the restriction operator (4.6) to get the initial values of ϕ_h and refine the mesh according to the principles in Section 4.9. Then we increase ξ a bit, use the restriction operator to set ϕ_h and check if the mesh has to be refined. This is repeated until the first refinement takes place, see Figure 4.9. Then, for every possible value of a in (4.50), we compute the discrete energy $E_h(a)$. As can be seen in Figure 4.10, for any value of a , the energy increases (although only slightly).

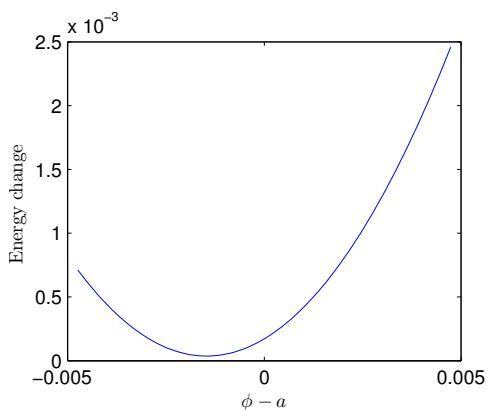


Figure 4.10: Energy change in the one dimensional example. The value ϕ of the triangle bisected by the red line in Figure 4.9 has to be distributed onto the two new triangles as a and $2\phi - a$. For every possible value of a , the energy increases.

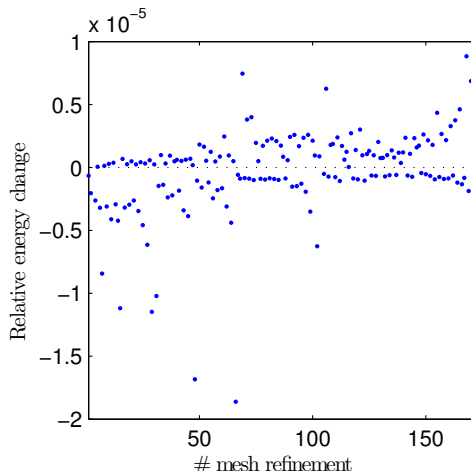


Figure 4.11: Energy change due to mesh adaption in a large simulation with 174 mesh adaptations. Two very negative values are not shown.

A second “real-world” example shows that this behaviour is typical. We recorded the energy change due to mesh adaption (coarsenings and refinements) during a large (two-dimensional) simulation from Section 5.2. The relative energy change, i.e.

$$\frac{E_h^{\text{ref}} - E_h}{E_h}$$

is shown in Figure 4.11. The overall absolute change due to mesh adaption was about -0.01 . In the same time the energy dropped from $2 \cdot 10^4$ to $7 \cdot 10^2$, so that the fraction of energy change that was due to mesh adaption is of the order 10^{-4} .

Remark. If we don’t use the energy minimization at all, but keep the temporary values from Section 4.9.2, then the energy often increases a lot. But what happens next is that the error estimator for the time adaptivity switches to very small time steps. Then, within usually five to ten time steps, the solution evolves to a state which has the same energy as the state in the case of energy minimization. This is of course due to the gradient-flow structure of the equation. \diamond

4.10 Linear solver

For the linear solver, we use the external package PETSc, see Section 7.1.

It was quite clear to us that for a symmetric positive definite LSE which is sparse and has a considerable number of unknowns, the best choice should be the preconditioned conjugate

gradient (PCG) method. As preconditioner, we tested the standard preconditioner shipped with PETSc and found as expected that incomplete Cholesky decomposition (ICC) yields the best results.

It was therefore a surprise when we discovered that a full Cholesky decomposition $A = L^t L$ is in most cases much faster than PCG with ICC, see the results of the next section. This was not the case for the Cholesky decomposition included in PETSc, but with another package called CHOLMOD [Davis and Hager, 2005, and references therein]. A crucial point is that the rows of the matrix are first permuted using an approximate minimal degree strategy [Amestoy et al., 1996], so that the fill-in is reduced. Still, the number of nonzeros of L is 10–15 times higher than the number of nonzeros of A .

Also remember that we have a non-constant system matrix, so that the (numerical) decomposition has to be repeated after every sub-timestep. The symbolical decomposition (i.e. finding the entries that are nonzero) only has to be redone if the mesh changes.

To see how far we can go with the Cholesky decomposition, we started a simulation with slightly more than 600 000 unknowns and got for the memory usage

	ABM	PCE	SPASI
Memory [GB]	2.3	3.0	3.4

Since 4GB of memory are quite standard nowadays, about a million unknowns are possible using Cholesky decomposition, see also the simulation in Section 6.5.3.

4.11 Comparison of the methods

In this section, we compare the different methods presented in Section 4.8 in two ways:

First, we solve Poisson's problem in the space of piecewise constant functions. The reason for the choice of this model problem is, as discussed at the beginning of Section 4.8.5, that the only nonconforming term of the discretization is

$$\int_{\Omega} \nabla_h \nabla \cdot J_h^{i+1} \cdot \nabla_h \nabla \cdot \tilde{J}.$$

Defining $z_h := \nabla \cdot J_h$ and $\tilde{z}_h := \nabla \cdot \tilde{J}$, we get

$$\int_{\Omega} \nabla_h z_h \cdot \nabla_h \tilde{z}_h.$$

Adding a right-hand side f_h , we get Laplace's problem

$$-\Delta_h z_h = f_h,$$

where the solution z_h is piecewise constant. For the methods based on the dual gradient, the Laplacian is

$$\Delta_h : \mathcal{L}_0(\mathcal{T}_h) \xrightarrow{\nabla_h} \mathcal{RT}_0(\mathcal{E}_h) \xrightarrow{\nabla \cdot} \mathcal{L}_0(\mathcal{T}_h),$$

and for the method using the piecewise constant energy, the Laplacian is

$$\Delta_h : \mathcal{L}_0(\mathcal{T}_h) \xrightarrow{\nabla_h} \mathcal{L}_0(\mathcal{T}_h)^2 \xrightarrow{\nabla \cdot} \mathcal{L}_0(\mathcal{T}_h).$$

In each case, the error in the L^2 -norm is measured for varying h .

Then, those methods that showed convergence in the first test are tested on problems characteristic for the applications in Chapters 5 and 6.

4.11.1 Poisson's problem

We solve here the problem

$$-\Delta_h u_h = f_h \tag{4.53}$$

with periodic boundary conditions for a number of different meshes, see Figure 4.12 for a list of the meshes. For each mesh, (4.53) is solved for different mesh sizes and the error is measured, see below. Here, the mesh size h is defined as half the length of the longest edge. For the cross mesh, the mesh is refined by bisection to get a finer mesh whereas for the cartesian, hexagonal and delaunay mesh, a new and finer mesh is loaded to retain the mesh type. For the latter, we used Matlab to generate the mesh and then converted it to a periodic mesh.

The domain on which the problem is solved is $\Omega = [0, 1] \times [0, L_y]$, where $L_y = 1$ for all meshes but the hexagonal mesh, where $L_y = 2/\sqrt{3}$ (to have the same number of triangles). The number of triangles ranges from 28 to 453 532 in the case of the delaunay mesh and from $32 = 2^5$ to $524 288 = 2^{19}$ in all other cases.

As right-hand side, we chose

$$f(x) = -8\pi^2 \left(1 + \frac{1}{L_y^2}\right) \sin(2\phi x_1) \sin(2\pi x_2/L_y)$$

so that the exact solution of $-\Delta u = f$ is

$$u(x) = \sin(2\pi x_1) \sin(2\pi x_2/L_y).$$

Then we calculate the error

$$\|u - u_h\|_{L^2(\Omega)}$$

using a numerical integration on the triangles which is exact up to fourth-order polynomials, see Section A.10.

The results are plotted in Figure 4.13. Since the solution for SINE is the same as for ABM, only the latter is shown. The procedures to solve Poisson's equation are described in Section A.5.

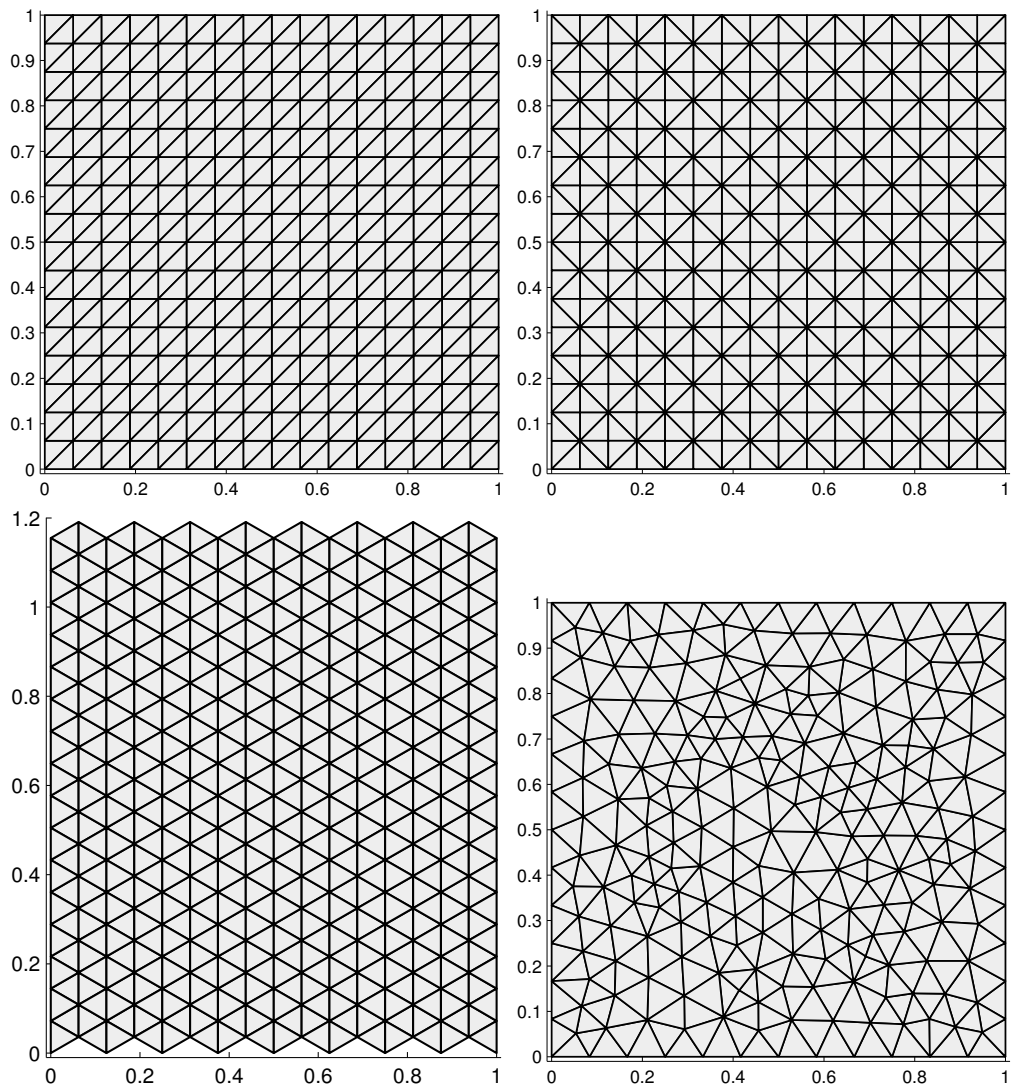


Figure 4.12: Meshes used for comparison: cartesian (top left), cross (top right), hexagonal (bottom left) and delaunay (bottom right). All meshes are shown after two refinement levels.

Table 4.4: Key figures of the simulations chosen for the comparison.

	Mesh changes	Time-step size	Unknowns
Coarsening (early stage)	—	10^{-9} to 10^{-2}	50 000
Coarsening (late stage)	2%	10^{-2} to 30	30 000 to 50 000
Epitaxial growth	10%	around $2 \cdot 10^{-4}$	55 000

As expected, the error converges linearly in h for all meshes when using ABM. We can see that using the PCE, the error also converges linearly in h . On the other hand, SPASI shows slower convergence for very small h on the hexagonal and delaunay meshes, but performs good on cartesian and cross meshes. Quite expected are the non-converging results for the least-squares gradient.

4.11.2 Test problems from the applications

We chose two problems that are prototypical for the simulations that arise in the applications presented in chapters 5 and 6. The number of unknowns are chosen such that the fastest method takes two to three hours. As we will see, the slowest methods then take well over a hundred hours.

- *Coarsening*. Here, we simulate the variable-quench equation, see Chapter 5. We separated the simulation into two parts, since the characteristics of the early stage are quite different from the late stages of the evolution. In the early stage, the mesh does not change (everything is interface) and the time-steps are very short, compare also to (3.12). In the later stages, some mesh-changes occur and the step sizes increase. The number of unknowns is about 50 000 at the beginning and about 30 000 at the end.
- *Epitaxial growth*. We chose here a straight moving step, see Chapter 6, which has as a consequence permanent mesh changes (about every 10th step). The time-step sizes are more or less constant. The number of unknowns is around 55 000.

Each of the three simulations was run with the four gradient methods and with both the conjugate gradient method and Cholesky decomposition. All simulations were run using the “cross” mesh on standard PCs. For details on the hard- and software we used, see Section A.11.

The results are shown in Table 4.5.

4.11.3 Conclusions

We first note that concerning the linear solver, Cholesky decomposition (using the Cholmod package) is superior to the conjugate gradient method in nearly all simulations. The CG method is only faster if the time steps are very small, which means that the eigenvalues of the system matrix are large and only a small number of steps is necessary.

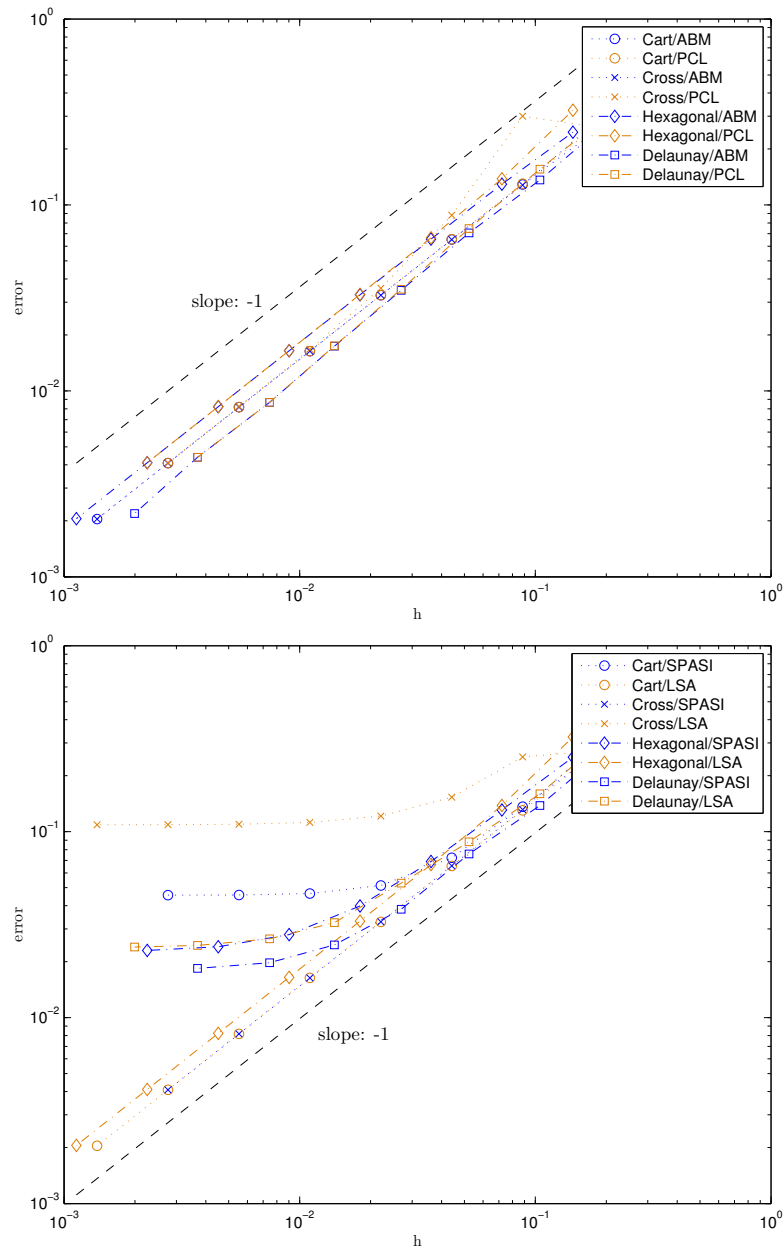


Figure 4.13: Error $\|u - u_h\|_{L^2(\Omega)}$ versus h . The Arnold–Brezzi method and the piecewise constant energy show convergence (upper plot). The sparse approximate inverse is only convergent on the cartesian and cross meshes and the gradient via least squares only converges on hexagonal meshes.

Table 4.5: Comparison of the different methods on three application problems for both the conjugate gradient and Cholesky decomposition as linear solver. The setup time contains everything that has to be done after a mesh change. If a value varies over time, we show the largest value.

	conjugate gradient				Cholesky			
	Time [hours]	Steps	Setup time [minutes]	Memory [MB]	Time [hours]	Steps	Setup time [minutes]	Memory [MB]
Coarsening (early stage)	ABM	0:13	41	0:02	75	40	0:02	120
	PCE	0:06	65	0:16	95	65	0:16	240
	SINE	0:08	40	0:01	50	—	—	—
	SPASI	0:06	43	0:05	115	43	0:05	230
Coarsening (late stage)	ABM	143:04	1248	0:02	75	1246	0:02	120
	PCE	132:44	2815	0:16	95	2793	0:16	240
	SINE	134:06	1251	0:00	50	—	—	—
	SPASI	65:30	1288	0:05	115	1274	0:05	230
Epitaxial growth	ABM	21:23	1659	0:03	85	1659	0:03	120
	PCE	3:54	1712	0:16	110	1715	0:16	240
	SINE	28:32	1659	0:00	60	—	—	—
	SPASI	4:41	1657	0:05	120	1660	0:05	250

Note: In the coarsening simulation, the mesh was not sufficiently fine for PCE, compare also to Section 4.9.1. This is the reason for the large number of steps. We therefore have to use a finer mesh for PCE which yields a time of 37:12 hours in 1358 steps for the Cholesky solver.

Table 4.6: Comparison of the running times for a large system and small time-steps using the CG solver.

	Time [hours]	Steps	Setup time [minutes]	Memory [GB]
ABM	1:32	76	0:24	1.0
PCE	0:58	113	4:30	1.2
SINE	1:40	76	0:06	0.6
SPASI	0:47	74	0:29	1.3

The best performing discretization method is ABM with Cholesky. This also matches our experience from other simulations. The next best method (performance-wise) is SPASI, although there we introduce another source of approximation error by using an approximate inverse. Also, the tests in Section 4.9.1 were not satisfactory. The PCE method is not very fast and needs a finer mesh than the other methods due to the different definition of the discrete gradient. Moreover, the Cholesky decomposition of SPASI and PCE has much more nonzeros than the one of ABM and therefore needs almost twice as much memory. The biggest disadvantage of SINE is that only iterative solvers without preconditioner can be used, which makes it very slow.

To see how the methods perform in the cases where CG is the better choice, we started another simulation of the early stage of coarsening with slightly more than 600 000 unknowns and found the running times shown in Table 4.6. The differences in the running times are not immense and do not justify to prefer one method over another considering that the early stage accounts for only a small fraction of the total simulation. Note that SINE is even in this case the slowest method. The fastest method using Cholesky was ABM taking about eleven hours.

The bottom line is that we will always use ABM, in most situations with Cholesky and in some special cases with the conjugate gradient method.

Coarsening

As the first application for our numerical method we chose coarsening processes. A large number of physical systems, including metal alloys [Domb and Lebowitz, 1983], polymer alloys and polymer–liquid-crystal mixtures [Papon et al., 2002], can be described by Cahn–Hilliard-type equations.

We consider here the shallow- and deep-quench equation introduced in Section 2.4 with random values of small amplitude as initial data. In the interfacial regime, numerical simulations and physical experiments suggest that the solutions show statistically self-similar behaviour with a characteristic length $\ell(t)$. This length scales as

- $\ell(t) \sim t^{1/3}$ for the shallow-quench equation and as
- $\ell(t) \sim t^{1/4}$ for the deep-quench equation.

The scale invariance of the corresponding sharp-interface equations shows that these are the only possible values. Indeed, remember from Section 2.6.2 that there are sharp-interface models whose behaviour is approximated by the shallow- and deep-quench equations in the interfacial regime. This was the Mullins–Sekerka (MS) free boundary problem for shallow quench and motion by surface diffusion (SD) for deep quench. These are both scale-invariant, i.e. solutions of MS and SD are still solutions if time and space are scaled by

$$x \mapsto \lambda x, t \mapsto \lambda^3 t \quad \text{and} \quad x \mapsto \lambda x, t \mapsto \lambda^4 t,$$

respectively. So if there is a scaling law for $\ell(t)$, it has to be the one stated above.

The statement $\ell \sim t^\alpha$ can obviously be decomposed into an upper bound $\ell \lesssim t^\alpha$ and a lower bound $\ell \gtrsim t^\alpha$. For the lower bound, no general predictions can be made without knowing the geometric configuration. For example, parallel stripes are stationary and therefore do not coarsen at all. A time-averaged version of the upper bound was shown in Kohn and Otto [2002].

This application is a good trial for the performance of our method since the coarsening is slow, especially in the deep-quench case. Therefore, simulations up to a large final time are necessary. Moreover, as the coarsening stops when $\ell(t)$ becomes comparable to the system size, large system sizes are also essential.

The scaling behaviour is connected to another property of the sharp-interface models: MS is nonlocal, so information is mediated through the bulk, whereas in SD only the curvature of the boundary, a local property, affects the motion of the boundary. Translated to the shallow- and deep-quench equations, the flux is usually nonzero in the whole domain (“flux through the bulk”) for shallow quench but is restricted to the transition regions for deep quench.

5.1 A model for phase segregation

For the derivation of the equations we follow the lines of Kohn and Otto [2002] and start from a temperature-dependent “variable quench” equation and identify the shallow- and deep-quench equations as certain limits of this equation. This variable quench equation can be derived from a three-dimensional Ising model with Kawasaki dynamics, see Domb and Lebowitz [1983]. It describes a mixture of two components, where we denote the relative concentration of one of the components by c . Setting $m := 2c - 1$, we get the order parameter $m \in [-1, 1]$.

We define the free energy of the mixture by

$$E(m) = \int_{\Omega} \frac{\beta}{2} (|\nabla m|^2 + (1 - m^2)) + \frac{1}{2} \mathfrak{E}(m) \quad (5.1)$$

where β denotes the inverse temperature and $\mathfrak{E}(m)$ is the entropy of the mixture:

$$\mathfrak{E}(m) = (1 + m) \log(1 + m) + (1 - m) \log(1 - m).$$

The relative concentration m then evolves according to

$$\partial_t m + \nabla \cdot J = 0, \quad (5.2a)$$

$$J = -(1 - m^2) \nabla \frac{\delta E}{\delta m}(m), \quad (5.2b)$$

where the variational derivative of E is

$$\frac{\delta E}{\delta m}(m) = -\beta(\Delta m + m) + \frac{1}{2} \frac{\delta \mathfrak{E}}{\delta m}(m), \quad \frac{\delta \mathfrak{E}}{\delta m}(m) = \log \frac{1 + m}{1 - m}$$

and thus

$$\nabla \frac{\delta E}{\delta m}(m) = -\beta \nabla(\Delta m + m) + \frac{1}{2} \nabla \frac{\delta \mathfrak{E}}{\delta m}(m), \quad \nabla \frac{\delta \mathfrak{E}}{\delta m}(m) = \frac{2}{1 - m^2} \nabla m.$$

Inserting this in equation (5.2) yields

$$\partial_t m - \Delta m + \beta \nabla \cdot ((1 - m^2) \nabla(\Delta m + m)) = 0. \quad (5.3)$$

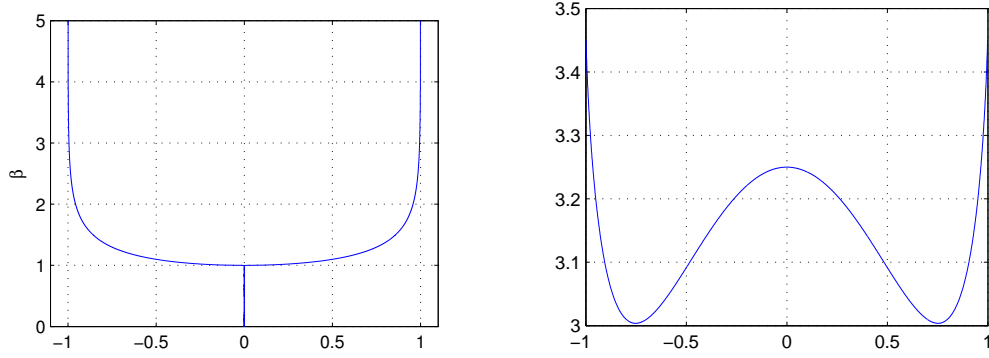


Figure 5.1: Left: Position of the minima of G_{VQ} for different values of the inverse temperature β and $\alpha = 5$. Right: Potential G_{VQ} for $\alpha = 5$ and $\beta = 1.3$. Note that G_{VQ} is bounded with $G_{VQ}(\pm 1) = \alpha \log 2$.

We now look at the bulk equilibrium values, which are the minima of

$$\tilde{G}(m) := \beta(1 - m^2) + (1 + m)\log(1 + m) + (1 - m)\log(1 - m). \quad (5.4)$$

For $\beta \leq 1$, the function \tilde{G} is convex and has only one minimum at zero. For $\beta > 1$, there are two minima at $g^+ > 0$ and $g^- = -g^+$. If $\beta \rightarrow \infty$, the entropy is small compared to the first term, so $g^+ \rightarrow 1$. See Figure 5.1 for a plot of the location of the bulk equilibrium values and a plot of G_{VQ} , which is a rescaled version of Equation (5.4), see (5.12c).

Therefore, two regimes are considered:

- the *shallow quench* limit, where $\beta \rightarrow 1$, $\beta > 1$ and
- the *deep quench* limit, where the temperature is very small ($\beta \rightarrow \infty$).

5.1.1 Shallow quench

First we determine the behaviour of the bulk equilibrium values in the regime $\beta \rightarrow \infty$, $\beta > 1$. That is, we look for a (constant) value m fulfilling

$$\text{diff } E(m) = 0 \iff \beta m = \frac{1}{2} \log \frac{1+m}{1-m}. \quad (5.5)$$

Since $g^\pm \rightarrow 0$ in this regime, we set $\epsilon = \beta - 1$ with $\epsilon > 0$, $\epsilon \ll 1$ and make the Ansatz

$$g^+ = f(\epsilon), \quad f(\epsilon) \ll 1.$$

Then equation (5.5) is

$$\begin{aligned} 2(\epsilon + 1)f(\epsilon) &= \log(1 + f(\epsilon)) - \log(1 - f(\epsilon)) \\ &\approx f(\epsilon) - \frac{1}{2}f(\epsilon)^2 + \frac{1}{3}f(\epsilon)^3 - \left(-f(\epsilon) - \frac{1}{2}f(\epsilon)^2 - \frac{1}{3}f(\epsilon)^3\right) \\ &= 2f(\epsilon) + \frac{2}{3}f(\epsilon)^3. \end{aligned}$$

Dividing by $2f(\epsilon)$ yields

$$\epsilon + 1 \approx 1 + \frac{1}{3}f(\epsilon)^2 \iff f(\epsilon) \approx \sqrt{3\epsilon},$$

which means

$$g^+ \approx \sqrt{3(\beta - 1)} \quad \text{for } 0 < \beta - 1 \ll 1. \quad (5.6)$$

Therefore, we rescale m in Equation (5.3) by

$$m = \sqrt{3\epsilon} \hat{m}.$$

To keep the nonlinear term, we have to rescale time and space according to

$$t = \frac{4\alpha_1^4}{\epsilon^2} \hat{t} \quad \text{and} \quad x = \alpha_1 \sqrt{\frac{2}{\epsilon}} \hat{x},$$

where α_1 is a constant to scale the potential that will be fixed later. This yields

$$\epsilon^{5/2} (\partial_{\hat{t}} \hat{m} + \hat{\Delta}^2 \hat{m} - 2\alpha_1^2 \hat{\Delta}(\hat{m}^3) + 2\alpha_1^2 \hat{\Delta} \hat{m}) + \mathcal{O}(\epsilon^{7/2}) = 0,$$

so to highest order, we get the Cahn–Hilliard equation

$$\partial_{\hat{t}} \hat{m} + \hat{\nabla} \cdot \left(-M_{\text{SQ}}(\hat{m}) \hat{\nabla} \frac{\delta E_{\text{SQ}}}{\delta \hat{m}}(\hat{m}) \right) = 0 \quad (5.7a)$$

with constant mobility

$$M_{\text{SQ}} \equiv 1 \quad (5.7b)$$

and the energy and potential

$$E_{\text{SQ}}(\hat{m}) = \int_{\hat{\Omega}} \frac{1}{2} |\hat{\nabla} \hat{m}|^2 + G_{\text{SQ}}(\hat{m}) \quad \text{and} \quad G_{\text{SQ}}(\hat{m}) = \frac{\alpha_1^2}{2} (1 - \hat{m}^2)^2. \quad (5.7c)$$

5.1.2 Deep quench

As before, we first determine the location of the bulk equilibrium values in the case $\beta \gg 1$. We set $\epsilon = 1/\beta$ and assume

$$g^+ = 1 - f(\epsilon), \quad f(\epsilon) \ll 1.$$

Then equation (5.5) reads as

$$\begin{aligned} 1 - f(\epsilon) &= \frac{\epsilon}{2} (\log(2 - f(\epsilon)) - \log(f(\epsilon))) \\ \iff \log(f(\epsilon)) &= -\frac{2}{\epsilon} (1 - f(\epsilon)) + \log(2 - f(\epsilon)) \\ \iff f(\epsilon) &= \exp\left(\underbrace{-\frac{2}{\epsilon} (1 - f(\epsilon))}_{\approx 1}\right) \underbrace{(2 - f(\epsilon))}_{\approx 2} \approx 2e^{-2/\epsilon} \end{aligned}$$

and therefore

$$g^+ \approx 1 - 2e^{-2\beta} \quad \text{for } \beta \gg 1. \quad (5.8)$$

For $\beta \rightarrow \infty$, we rescale Equation (5.3) by

$$t = \frac{\alpha_2^4}{\beta} \hat{t} \quad \text{and} \quad x = \alpha_2 \hat{x},$$

where, again, α_2 is a constant to be fixed later. This yields

$$\beta \left(\partial_{\hat{t}} m - \hat{\nabla} \cdot \left((1 - m^2) \hat{\nabla} (\hat{\Delta} m + \alpha_2^2 m) \right) \right) - \alpha_2^2 \Delta m = 0,$$

which is to highest order a Cahn–Hilliard-type equation

$$\partial_{\hat{t}} m + \hat{\nabla} \cdot \left(-M_{\text{DQ}}(m) \hat{\nabla} \frac{\delta E_{\text{DQ}}}{\delta m}(m) \right) = 0 \quad (5.9a)$$

with degenerate mobility

$$M_{\text{DQ}}(m) = 1 - m^2 \quad (5.9b)$$

and the energy and potential

$$E_{\text{DQ}}(m) = \int_{\hat{\Omega}} \frac{1}{2} |\hat{\nabla} m|^2 + G_{\text{DQ}}(m) \quad \text{and} \quad G_{\text{DQ}}(m) = \frac{\alpha_2^2}{2} (1 - m^2). \quad (5.9c)$$

5.2 Simulating the equations

In this section, we prepare the equations such that we can apply our discretization and we choose various parameters, e.g. the α_i in the potentials.

5.2.1 Shallow quench

The energy of the shallow-quench equation is already in the form (4.1) of Chapter 4 and the mobility is constant, so we can apply our discretization without further work.

5.2.2 Deep quench

In contrast, the deep-quench equation (5.9) imposes a difficulty: the degenerate mobility. In our discretization scheme the mobility appears in the denominator, see Equation (4.18), and becomes zero for $m = \pm 1$. Another difficulty is that the potential is concave. As we have seen in Section 3.2, the time-step restriction to guarantee the positive-definiteness of the operator is more strict in the case of a concave potential.

We investigate two possible ways to overcome these problems:

- One possibility is based on the decomposition of the domain into bulk regions and interfacial regions, compare to Section 2.6.2. Due to the fact that there is no flux in the bulk, it is enough to solve the equation for the flux in the interfacial region, where the mobility is nonzero.
- The other idea is to use the variable-quench equation with a $\beta \gg 1$, which is the natural regularization of the deep-quench equation. Then the bulk equilibrium values are away from one, see (5.8), and therefore the mobility is nonzero.

We describe these two ideas in the following.

Restriction to interfacial layer

The asymptotic analysis carried out in Cahn et al. [1996] suggests that in the interfacial regime, Ω can be decomposed into regions $\Omega = \Omega_\epsilon^+ \cup \Omega_\epsilon^I \cup \Omega_\epsilon^-$, where $\phi \approx g^\pm$ in Ω_ϵ^\pm and Ω_ϵ^I includes the transition regions with centre Γ .

Since there is effectively no flux in Ω_ϵ^\pm , the error made when solving the equation for the flux only in Ω_ϵ^I is small. This idea is realized as follows.

Remember that the degrees of freedom are located on the edges of the mesh. We say that an edge is in the bulk if on both adjacent triangles, m has reached at least 99.9% of the bulk equilibrium values. All other edges are considered interfacial edges. Then, the system (4.24) is reduced to those rows and columns that correspond to an interfacial edge. This reduced system is then solved to compute the flux J . Consequently, $|m| \leq 0.999g^+ < 1$ and thus the mobility is positive.

Of course this new system is no longer symmetric and, additionally, the time-step restriction due to the concave potential still applies. Thus two of the features of our discretization are no longer valid and consequently we don't follow this Ansatz any further.

Using variable-quench with a low temperature

Considering that the deep-quench equation was the limit of the variable-quench equation for $\beta \rightarrow \infty$, the morphology of the solution of deep quench and of variable quench for $\beta \gg 1$ should be similar. This is again supported by the asymptotic analysis of Cahn et al. [1996, §4], who show that solutions to the variable-quench equation with large but finite β behave to highest order as motion by surface diffusion.

We have seen in (5.8) that the bulk equilibrium values approach ± 1 exponentially fast for increasing β (see also Figure 5.1 on page 91), so we cannot expect to be able to use a really large β . Numerical tests showed that $\beta = 4$ is a good compromise between a large β and a positive mobility. For this value, the bulk equilibrium values are

$$g^\pm \approx \pm 0.999326 \tag{5.10}$$

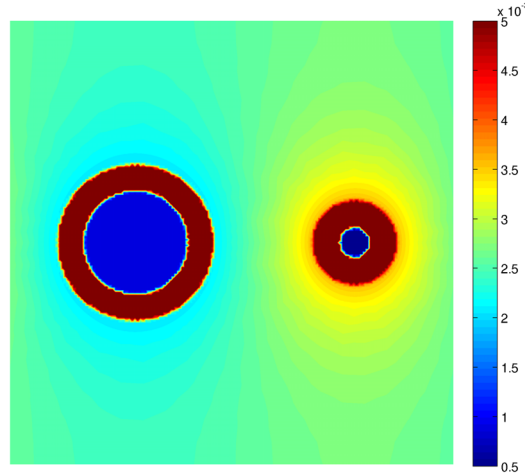


Figure 5.2: Mobility of VQ in the case of two islands with different radii. The mobility is near one at the interfaces (note that all values $\geq 5 \cdot 10^{-3}$ are coloured red) and $\approx 3 \cdot 10^{-3}$ in the bulk.

which yields a value of $M(g^\pm) \approx 10^{-3}$. To check if this is good enough, consider a setup of two circular islands with different radii. For the deep-quench equation, this is a stationary situation since the interfaces are not connected and there is no flux through the bulk. With variable quench and $\beta = 4$, we can see in Figure 5.2 that the mobility in the bulk is not zero but of order 10^{-3} as expected. So there will be some Ostwald ripening, i.e. the larger island grows slowly at the expense of the smaller island.

In the actual coarsening simulations, the change of the solution due to the flux through the bulk is small compared to the change induced by other effects like coalescence and interface movement, see also Figure 5.13 on page 108.

In summary, the variable-quench equation with $\beta = 4$ seems to be a good approximation of the deep-quench equation.

Before we can use the variable-quench equation, we want the energy to have the form (4.1) used in the discretization. To get rid of the β in front of the Dirichlet term in the energy (5.1), we rescale time and space according to

$$t = \beta \alpha_3^2 \hat{t}, \quad x = \sqrt{\alpha_3 \beta} \hat{x} \quad \text{and} \quad E = \beta \hat{E}, \quad (5.11)$$

where $\alpha_3 > 0$ is again an additional parameter to be chosen later. The transformed operators are

$$\partial_t = \frac{1}{\beta \alpha_3^2} \partial_{\hat{t}}, \quad \nabla = \frac{1}{\sqrt{\alpha_3 \beta}} \hat{\nabla},$$

so the rescaled version of equation (5.2) is

$$\partial_{\hat{t}} m + \hat{\nabla} \cdot \left(-(1 - m^2) \alpha_3 \hat{\nabla} \frac{\delta E}{\delta m}(m) \right) = 0.$$

Now writing the energy (5.1) in the new spatial variable,

$$E(m) = \int_{\Omega} \frac{\beta}{2} |\hat{\nabla} m|^2 + \frac{\alpha_3 \beta^2}{2} (1 - m^2) + \frac{\alpha_3 \beta}{2} \mathfrak{E}(m),$$

we find

$$\hat{E}(m) = \int_{\Omega} \frac{1}{2} |\hat{\nabla} m|^2 + \frac{\alpha_3 \beta}{2} (1 - m^2) + \frac{\alpha_3}{2} \mathfrak{E}(m).$$

and therefore

$$\frac{\delta \hat{E}}{\delta m}(m) = -\alpha_3 \beta \Delta m - \alpha_3 \beta m + \frac{\alpha_3}{2} \mathfrak{E}(m) = \alpha_3 \frac{\delta E}{\delta m}(m).$$

Thus, we have again a Cahn–Hilliard-type equation

$$\partial_t m + \hat{\nabla} \cdot \left(-M_{\text{VQ}}(m) \hat{\nabla} \frac{\delta E_{\text{VQ}}}{\delta m}(m) \right) = 0 \quad (5.12a)$$

with degenerate mobility

$$M_{\text{VQ}}(m) = 1 - m^2 \quad (5.12b)$$

and the energy and potential

$$\hat{E}(\hat{m}) = \int_{\Omega} \frac{1}{2} |\hat{\nabla} \hat{m}|^2 + G_{\text{VQ}}(\hat{m}) \quad \text{and} \quad G_{\text{VQ}} = \frac{\alpha_3}{2} (\beta(1 - \hat{m}^2) + \mathfrak{E}(\hat{m})). \quad (5.12c)$$

5.2.3 Choice of parameters

It is convenient for the simulations if all equations lead to solutions where the transition regions have a similar width. Then, the refinement rules presented in Section 4.6 apply to all cases equally.

We took the following approach: first, we chose α_3 in the potential of the variable-quench equation such that the interface was similar to the one in epitaxial growth. This yielded

$$\alpha_3 = 5.$$

Then, the α 's in the shallow- and deep-quench equation were chosen such that $G''(0)$ is the same in all cases. Remember that the most unstable wavelength λ^* during spinodal decomposition is determined by $G''(0)$. Having the same λ^* for all equations, we can use the same system size (and mesh) for a given number of coarsening levels. The resulting values are

$$\alpha_1 = \sqrt{\frac{15}{2}} \quad \text{and} \quad \alpha_2 = \sqrt{15}.$$

In all three cases the most unstable wavenumber and -length is then

$$\hat{k}^* = \sqrt{\frac{15}{2}} \approx 2.74 \quad \text{and} \quad \hat{\lambda}^* = 2\pi \sqrt{\frac{2}{15}} \approx 2.29.$$

As computational domain, we use a square $\hat{\Omega} = [0, \hat{L}]^2$ with periodic boundary conditions. We want to allow five levels of coarsening, which yields

$$\hat{L} = 2^5 \lambda^* \approx 73.4.$$

The number of degrees of freedom (=edges) in these simulations ranges from approx. 585 000 at the beginning to approx. 115 000 in the end. The simulations were stopped when a stationary state was detected. This was the case at around $\hat{t} = 10\,000$ for shallow quench and at $\hat{t} = 350\,000$ for variable quench.

We always start with random initial values of amplitude 0.01.

To be able to compare the results to other simulations, we have converted some of the parameters from their scaled values used in the simulations to the “usual” scaling, see the table below. By usual, we mean e.g. the one used in Kohn and Otto [2002].

	shallow quench	variable quench
k^*	1	0.61
λ^*	6.28	10.26
L	201.01	328.25
\hat{t}_{end}	10 000	350 000
t_{end}	575 000	35 000 000

For the shallow- and deep-quench equation, we can compute the one-dimensional minimizers, see Section 2.6.3:

$$m_{\text{SQ}}^{\min}(x) = \tanh(\sqrt{15/2}x) \quad \text{and} \quad m_{\text{DQ}}^{\min}(x) = \sin(\sqrt{15}x).$$

These two profiles, together with numerical data for the variable-quench profile are plotted in Figure 2.5 on page 23.

5.3 Results

We present in the following the results of our numerical simulations. They confirm a number of assertions:

- (A1) The morphology of the solutions has two phases: first a phase of spinodal decomposition then a phase of coarsening.

- (A2) During spinodal decomposition, the initial values grow exponentially with dispersion relation ω .
- (A3) The first phase ends with a predominant wavelength.
- (A4) In the coarsening regime, the solution is self-similar with a characteristic length $\ell(t)$.
- (A5) The characteristic length grows as $\ell(t) \sim t^{1/3}$ for shallow quench and as $\ell(t) \sim t^{1/4}$ for deep quench.
- (A6) The morphology of the shallow-quench equation is dominated by the flux through the bulk, whereas in the deep-quench equation, flux is only present along the interfaces.

For (A1), take a look at Figure 5.6 on page 103, where a measure for ℓ is plotted. The two phases can be easily distinguished.

In the next two sections, we compare the predictions (A2)–(A6) with our numerical simulations. All simulations use the parameters from Section 5.2.3.

5.3.1 Spinodal decomposition

In Section 2.6.1, we saw that as long as $|m| \ll 1$, the Fourier transform of m behaves like

$$\mathcal{F}(m) = c \exp(\omega(|k|)t),$$

where the growth factor, or dispersion relation, is given by

$$\omega(|k|) = -|k|^4 + 15|k|^2$$

for both shallow- and variable quench. To check this behaviour numerically, we proceeded as follows. At fixed time intervals, we interpolated the values of m to a equidistant cartesian mesh and applied Fourier transformation. For each wave vector k , we therefore get the function $\mathcal{F}(m)(k, \cdot)$. Then we take the logarithm of this function and use a linear fit to determine the slope:

$$\omega_{\text{num}}(|k|) := \text{slope of } \log \mathcal{F}(m)(k, \cdot).$$

The results are shown in Figure 5.3 on page 101, where each blue dot corresponds to the value of $\omega_{\text{num}}(|k|)$ for one wave vector k . Only values where the amplitude grew enough to identify a slope are shown. We find very good agreement.

In Figure 5.4, the Fourier transform of m is depicted at the end of the phase of spinodal decomposition. We can see that wave numbers around the predicted value of $\hat{k}^* \approx 2.74$ dominate.

5.3.2 Coarsening

We first address the self-similarity of the solutions (A4). Although one can describe the self-similarity more rigorously in various ways, we think that the most easily accessible one is still the visual impression. Therefore, we measured ℓ as explained below and took snapshots at times t_1 and t_2 chosen such that $2\ell(t_1) \approx \ell(t_2)$. We then shrank the second snapshot to half its size and glued four copies of it together. The result is shown in Figure 5.5.

Next, we come to the scaling behaviour (A5). The analysis in Kohn and Otto [2002] uses two different measures of the length-scale. The first one is the so-called interfacial energy density,

$$e(t) := \frac{1}{|\Omega|} \int_{\Gamma} 1,$$

which scales as $1/\text{length}$. They expect $e \sim \ell^{-1}$ and therefore (A5) translates into

$$e(t) \sim t^{-1/3} \text{ for SQ} \quad \text{and} \quad e(t) \sim t^{-1/3} \text{ for DQ.}$$

To get $e(t)$, we use that in the interfacial regime, the energy E can be approximated by the energy of the optimal profile multiplied by the length of the interface. Thus, we find

$$e(t) \approx \frac{1}{\sqrt{40}|\Omega|} E_{\text{SQ}}(m(t)) \quad \text{and} \quad e(t) \approx \frac{2}{\sqrt{15}\pi|\Omega|} E_{\text{DQ}}(m(t)).$$

In Figure 5.7, we show the normalized energies $E(t)/E(0)$ for shallow quench and deep quench. We see that the energies have indeed the anticipated scaling.

However, there are two nuisances. First, the graphs are rather bumpy at the end. This is due to the fact that for large times, and therefore large $\ell(t)$, individual events like coalescence of two ‘‘islands’’ have more impact on the total energy. Second, remember that we use the variable-quench equation with $\beta \gg 1$ to approximate the deep-quench equation. The energy measured is nevertheless E_{DQ} , which seems a bit odd. To eliminate the first flaw, we have run five simulations for each of shallow quench and deep quench and consider the average of the measured quantities. To get around the second one, we resort to the other measure of the length-scale mentioned in Kohn and Otto [2002], a negative norm:

$$L(t) := \frac{1}{\sqrt{|\Omega|}} \|m(x, t)\|_{H^{-1}(\Omega)}. \quad (5.13)$$

By definition of the H^{-1} -norm, see Section 2.2, L indeed scales as length. Computing L involves solving Poisson’s problem, so this measure is computationally more expensive than computing the energy. See Section A.6 for the implementation.

The results are shown in Figures 5.8 and 5.9. The coarsening rate is in good agreement with the theoretical prediction.

To get an idea of the evolution of m , snapshots of the simulations at the times marked in Figures 5.8 and 5.9 are shown in Figures 5.10 and 5.11, respectively.

Finally, we investigate the flux J in the two equations. The assertion (A6) concerning the flux, namely flux through the bulk in shallow quench and flux along the interface in deep quench, can be seen immediately from the equations. Denoting the variational derivative of E with w we get

$$J_{\text{SQ}} = -\nabla w_{\text{SQ}} \quad \text{and} \quad J_{\text{DQ}} = -M_{\text{DQ}}(m)\nabla w_{\text{DQ}}.$$

As the mobility for the deep-quench equation is zero in the pure phases, i.e. for $m = \pm 1$, flux only takes place along the interfaces.

To visualize the flux, we encoded the magnitude of the flux in the colour saturation (light = low flux, dark = high flux) and the direction is encoded in the hue. Figures 5.12 and 5.13 show that, as expected, there is a lot of flux in the bulk in shallow quench. For deep quench, most of the flux is along the interfaces. The detail on the bottom right shows that there is a little flux through the bulk, but that the magnitude of this flux is negligible compared to the flux along the interfaces.

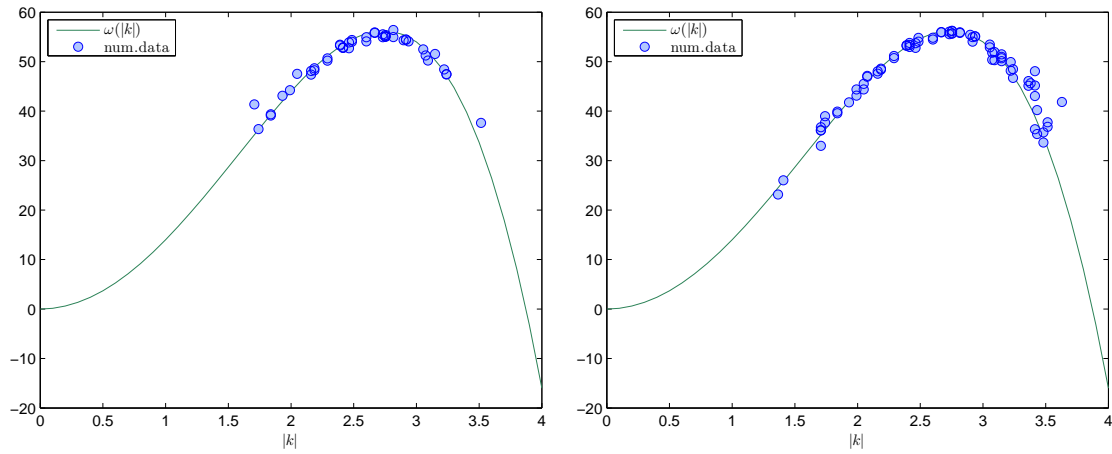


Figure 5.3: The predicted and measured values of the growth factor during spinodal decomposition for shallow (left) and variable (right) quench.

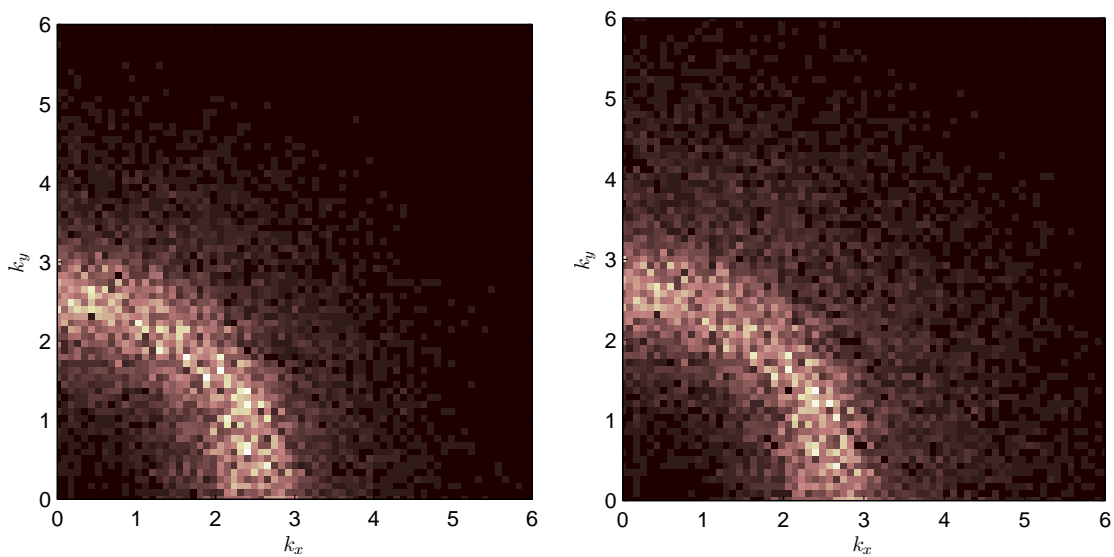


Figure 5.4: Plot of the Fourier transform $\mathcal{F}(m)$ at the end of the first phase for the shallow-quench (left) and the variable-quench (right) equations. The predicted value for the most unstable wave number $\hat{k}^* \approx 2.74$ is predominant.

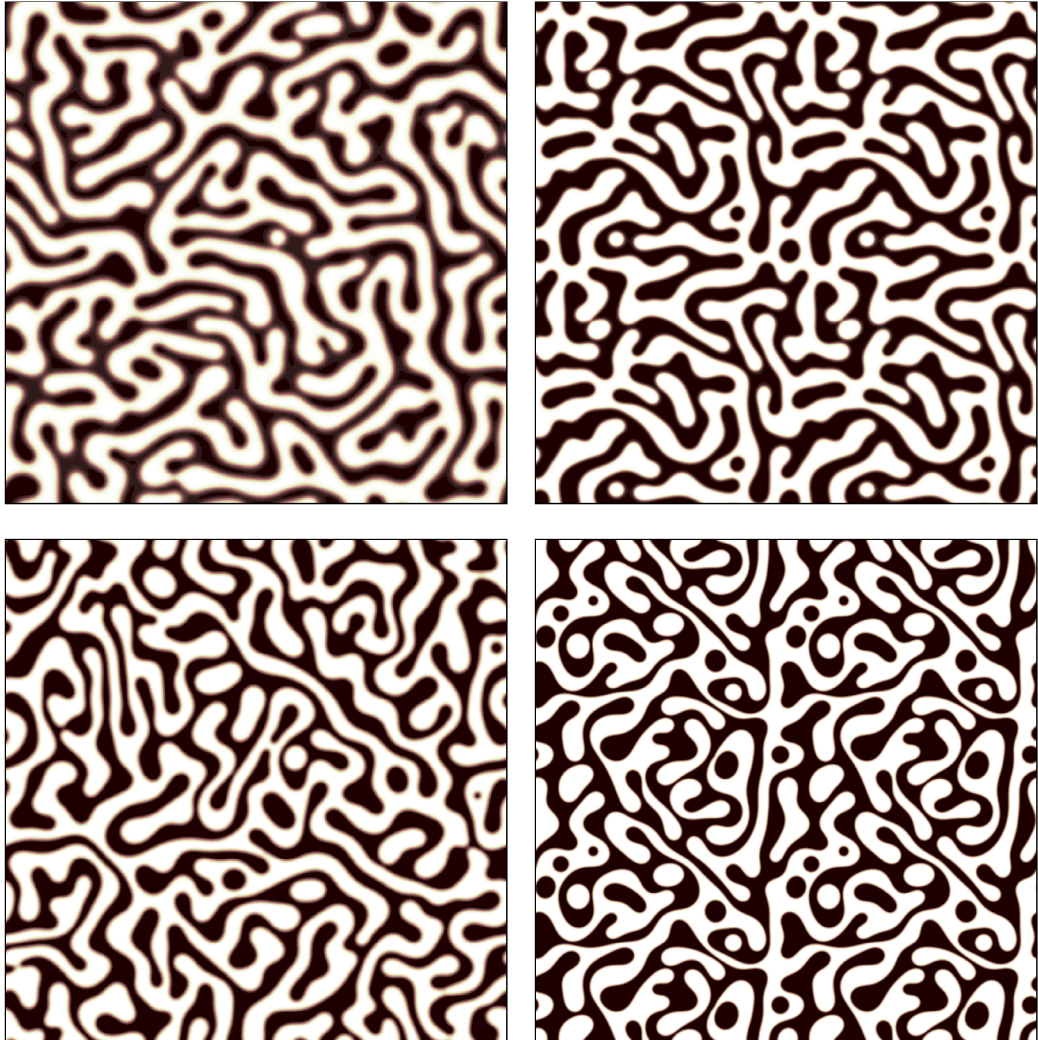


Figure 5.5: Self-similarity of the solution. On the left is shown a snapshot with $L(t) = 50$. On the right, four copies of a snapshot with $L(t) = 100$ were scaled to half their size and put together. The upper row is for shallow quench and the lower row for deep quench.

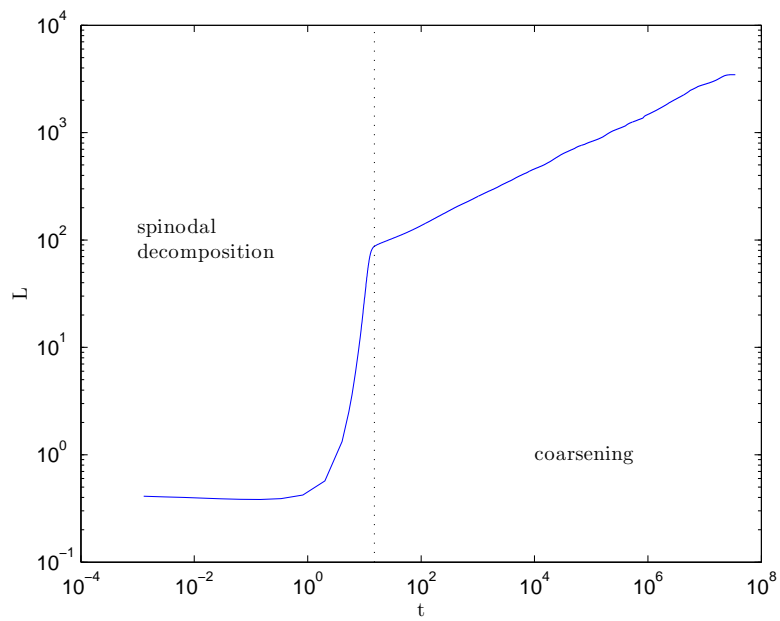


Figure 5.6: The length-scale L versus time in a log–log plot for variable quench. The two phases are clearly visible.

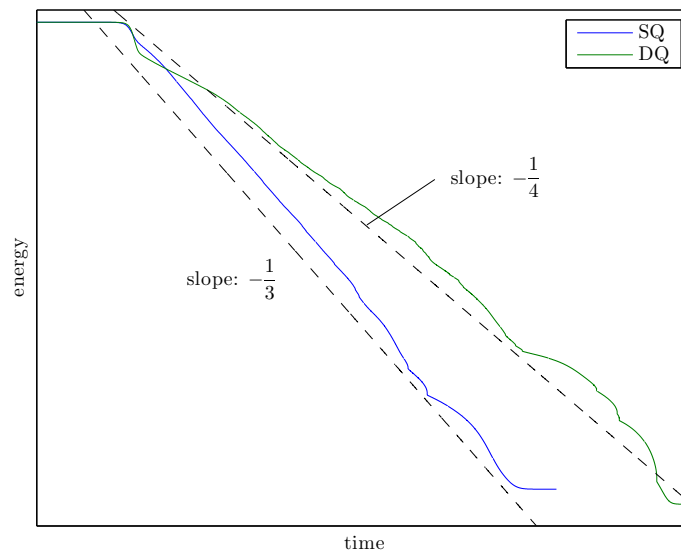


Figure 5.7: Energy versus time in a log–log plot. The different scaling of the energy for shallow quench (SQ) and deep quench (DQ) is clearly visible.

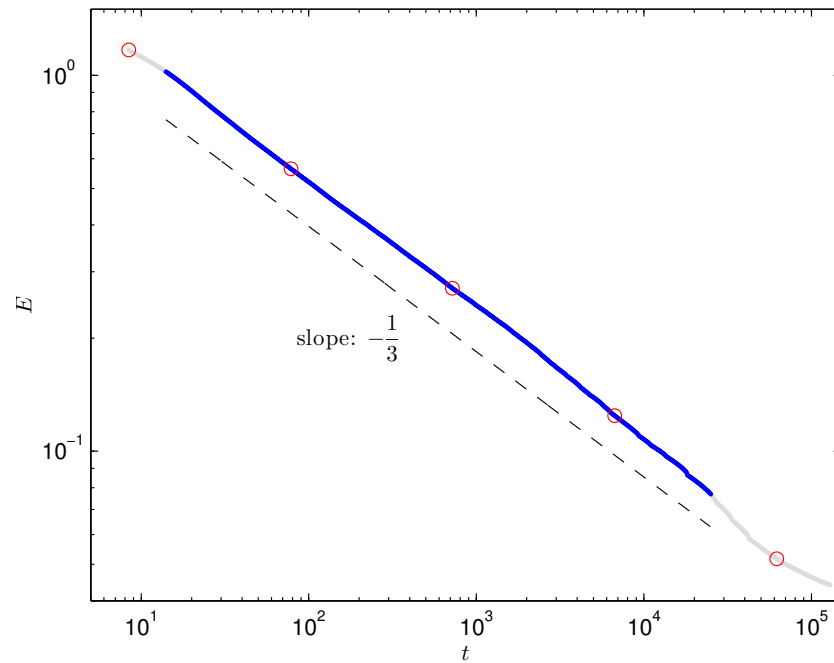


Figure 5.8: Mean of the energy E over five simulations for shallow quench. Snapshots of the simulation taken at the times marked with a red circle are shown in Figure 5.10.

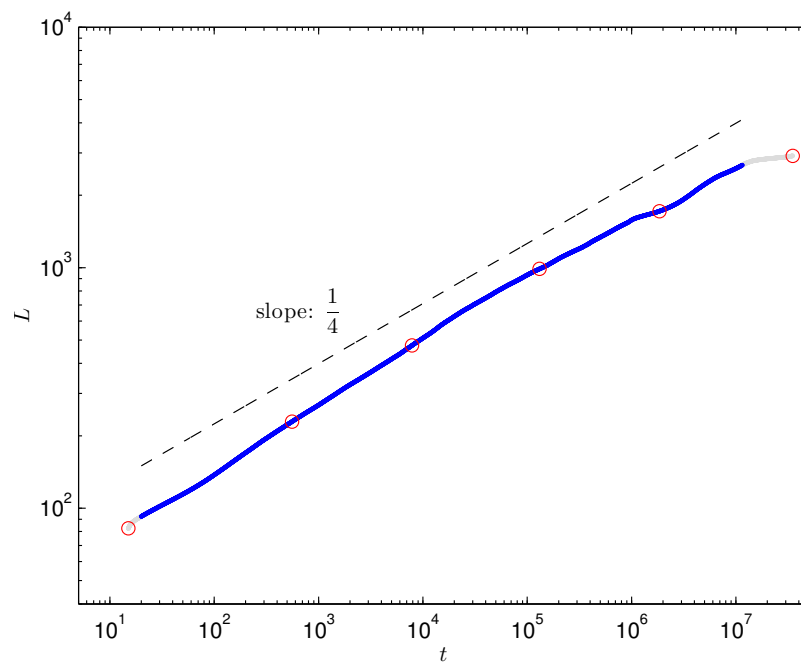


Figure 5.9: Mean of the negative norm L over five simulations for variable quench. Snapshots of the simulation taken at the times marked with a red circle are shown in Figure 5.11.



Figure 5.10: Snapshots of the shallow-quench simulation at the times marked in Figure 5.8 $t \approx 8$, $t \approx 78$, $t \approx 721$, $t \approx 6678$, $t \approx 61591$ and $t \approx 552594$.

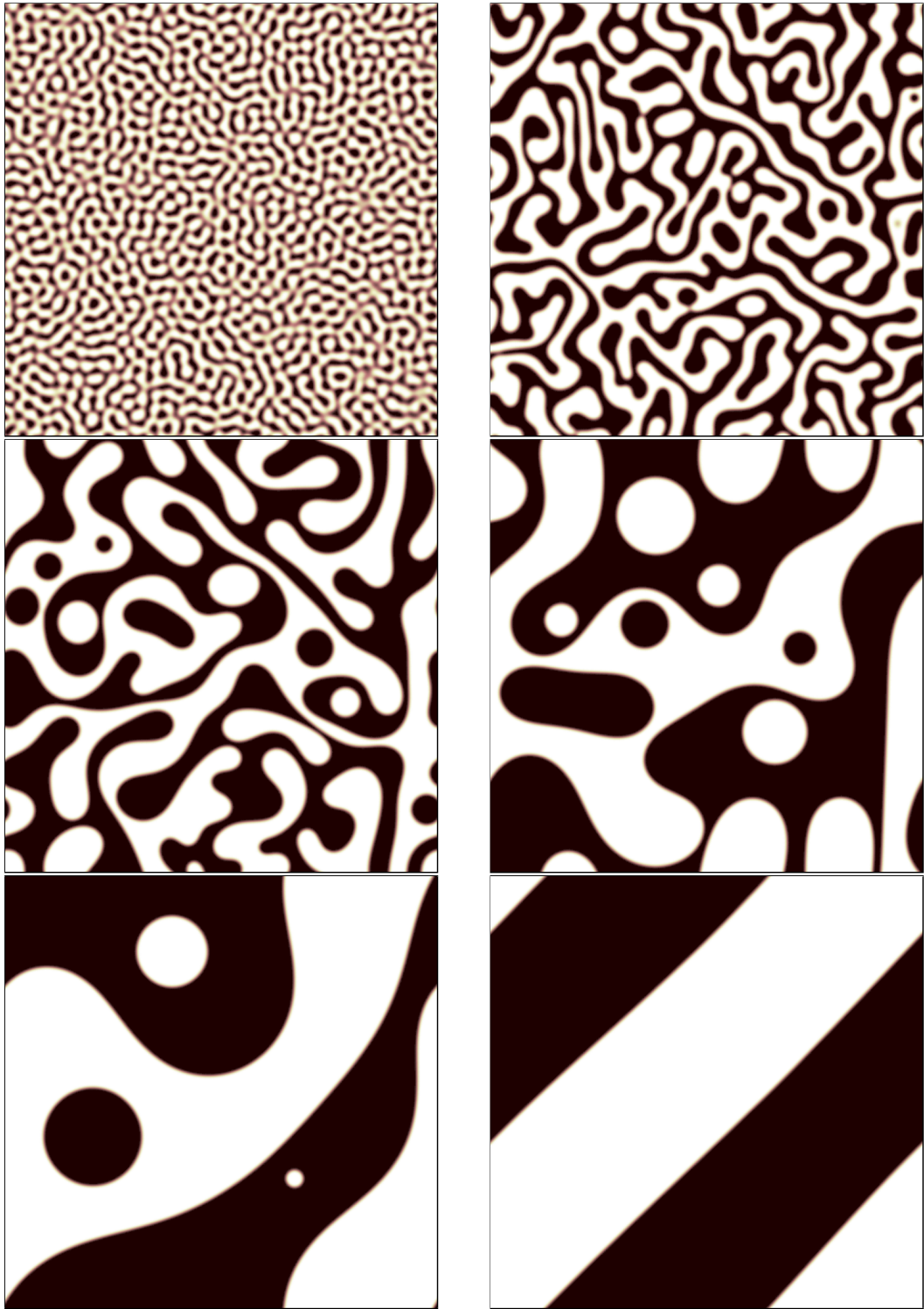


Figure 5.11: Snapshots of the variable-quench simulation at the times marked in Figure 5.9: $t \approx 15$, $t \approx 557$, $t \approx 7850$, $t \approx 131268$, $t \approx 1849668$ and $t \approx 35039562$.

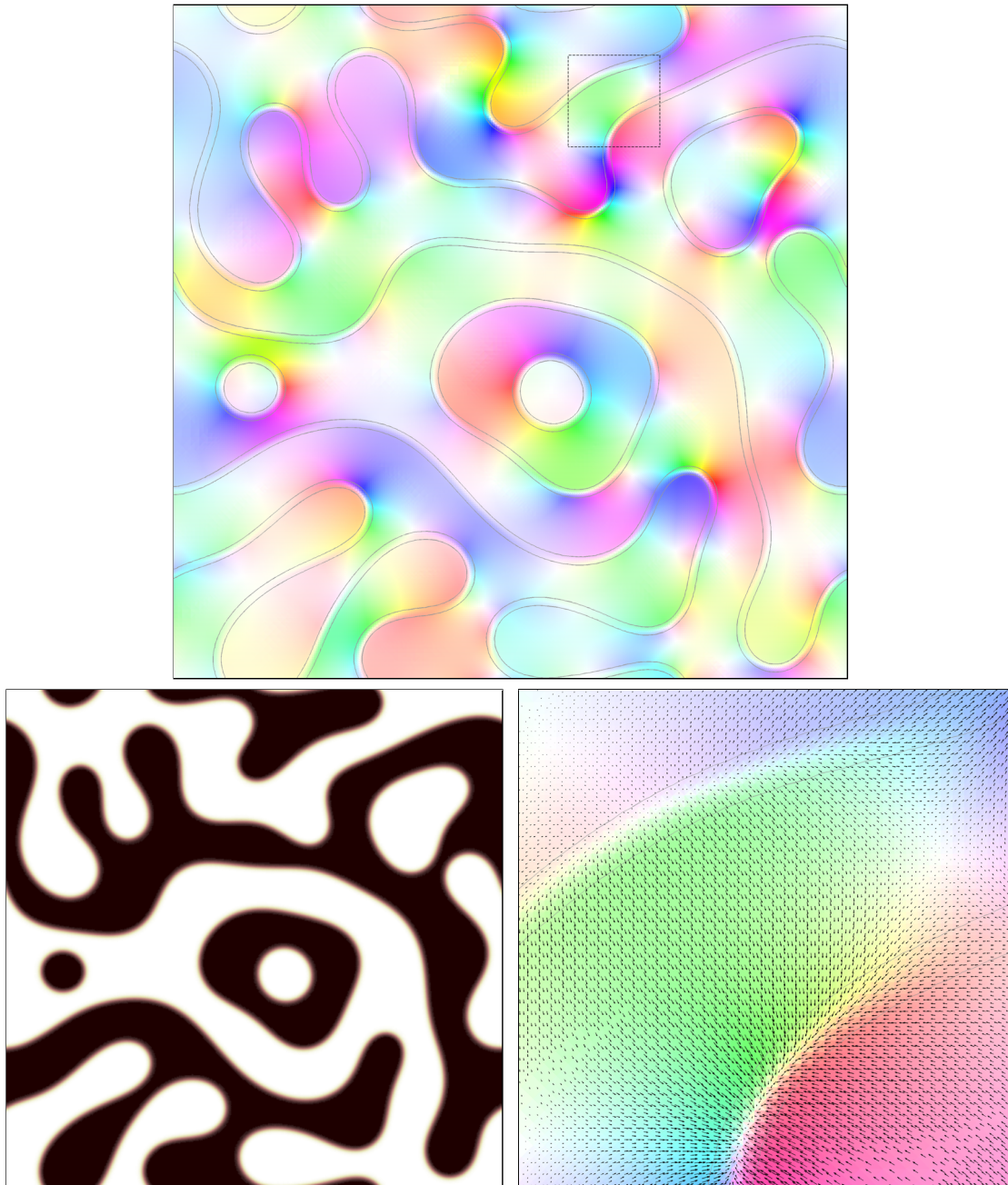


Figure 5.12: The upper picture shows the flux in a shallow quench simulation: the hue shows the direction and the saturation shows the magnitude of the flux (see circle on the right). A lot of flux takes place in the bulk (the grey lines indicate the interfaces). The lower pictures show ϕ (left) and a detail of the upper picture (right).

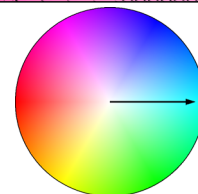
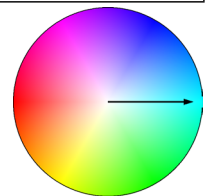




Figure 5.13: The upper picture shows the flux in a variable quench simulation: the hue shows the direction and the saturation shows the magnitude of the flux (see circle on the right). Clearly visible is that the flux goes along the interfaces (grey lines), as expected for an approximation of surface diffusion. The lower pictures show ϕ (left) and a detail of the upper picture (right).



Epitaxial Growth

As the second application for our numerical method, we chose epitaxial growth. Epitaxial growth is the layer-by-layer growth of a thin crystalline film on a substrate. This growth process is often realized using molecular beam epitaxy, see Section 6.1.

There are various models for epitaxial growth of thin films that are distinguished by different scales in time and space, see e.g. Voigt [2005] for an overview. These models range from full atomistic descriptions using molecular dynamics and kinetic Monte Carlo methods over discrete–continuous models in which only the growth direction is resolved on an atomistic scale and the lateral direction is coarse-grained, to fully continuous models which describe the film height as a smooth hypersurface.

We concentrate here on the discrete–continuous case. The standard models for this mesoscopic scale are step-flow models, which are motivated in Section 6.3. Since step-flow models are based on free-boundary problems, numerical simulations of these models cannot handle topological changes. To overcome this limitation, we consider in Section 6.4 a diffuse-interface approximation introduced by Otto, Penzler, Rätz, Rump, and Voigt [2004]. This point of view is opposite to the one of Section 2.6.2: there, we started from a Cahn–Hilliard-type equation (corresponds to the diffuse-interface approximation) and found as a limit a free-boundary problem (corresponds to the step-flow model).

To emphasize the possibility to handle topological changes, we focus on one of the instabilities explained in Section 6.2. For certain parameters, an initially straight step develops an instability that leads to a pinch-off. We compare our simulation to a step-flow simulation. Up to the pinch-off, the results coincide. Only the diffuse-interface approximation can go beyond the topological change, see Section 6.6 for the results.

Parts of the following have been published in Otto, Penzler, and Rump [2005] and Haußer, Otto, Penzler, and Voigt [2008].

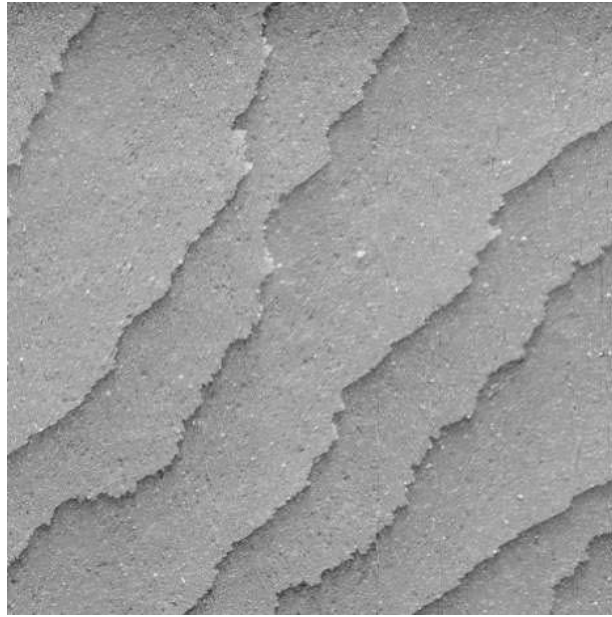


Figure 6.1: Epitaxial growth of Si(001). The figure shows atomistically flat terraces, which are separated by steps of atomic height. Courtesy of Polop et al.

6.1 Molecular beam epitaxy

Molecular beam epitaxy (MBE) is a technology to grow ordered crystalline films that inherit their atomic structure from the substrate. It produces almost defect-free, high-quality materials and is widely used in research and industry to make semiconductor devices and structures.

Apart from its technological relevance, MBE also played a central role in the growth and development of nanoscience and nanotechnology [McCray, 2007].

MBE consists mainly in depositing single atoms or molecules onto a substrate in a vacuum chamber. We consider here homoepitaxy, i.e. the substrate is of the same material as the deposited atoms. Furthermore, we think of a crystalline substrate which has been cut with a small angle to its crystallographic orientation. Therefore, the substrate consists of atomistically flat terraces, divided by atom-high steps, see Figure 6.2. Vapor atoms arriving at the surface become adatoms (ad-sorbed atoms) and diffuse on the flat terraces. Upon coming to a step, the adatoms attach to the step with a certain rate and therefore the crystal grows layer by layer.

Figure 6.1 shows a scanning tunneling microscopy (STM) image of a silicon surface. The flat terraces and atom-high steps are clearly visible.

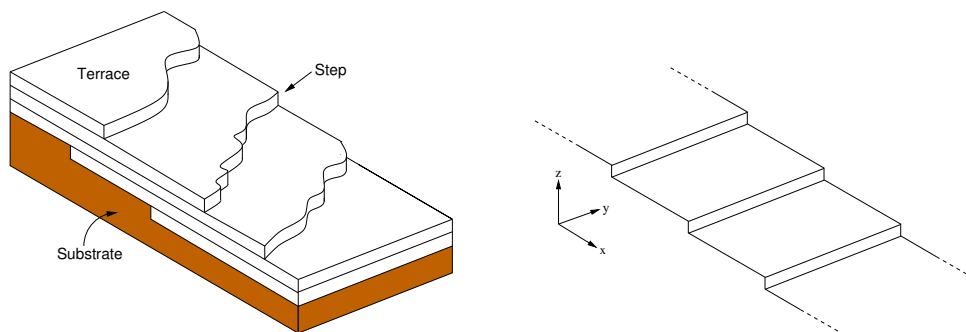


Figure 6.2: Terms used in this section. Left: The thin film grows layer by layer on a substrate through (horizontal) movement of the steps. The typical shape of the film are large terraces separated by atom-high steps. Right: Infinite descending step train.

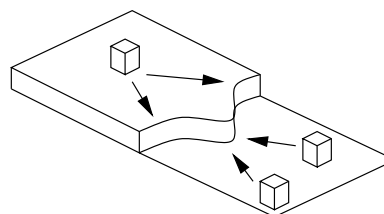
6.2 Instabilities in epitaxial growth

An initially straight step typically does not remain straight during growth, but is subject to various instabilities. There are essentially three types of instabilities which influence the film morphology during growth: step bunching, step meandering and mound formation, see e.g. Politi et al. [2000]; Krug [2005]. They all have their origin on the atomistic scale and result from asymmetries in the energy barriers for individual hops of atoms on the surface. However, a fully atomistic description of the film is limited to sample sizes of several nanometers and thus far off from any feature size in semiconductor devices. In order to predict the surface morphology on larger length scales, continuum models are required which incorporate the instabilities generated on the atomistic scale. Fully continuous models with these properties still have to be derived. On a mesoscopic scale, discrete–continuum models, the step-flow models, are promising candidates. These will be described in Section 6.3.

Step-meander instability

From the three mentioned instabilities, we focus on the step-meander instability. Already in the mid-sixties of the last century, Ehrlich and Hudda [1966] found that adatoms attach to a step down with a lower rate than to a step up. The (not entirely correct) picture one can have in mind is that the adatom has to “jump over the edge” before being able to attach to the step and thus loses all its bonds. In reality the situation is often more complicated and includes reordering of several atoms, but in any case it costs the adatom more energy when attaching to a step from an upper terrace. The energy barrier the atom has to overcome is called the Ehrlich–Schwoebel (ES) barrier.

This asymmetry in attachment rate is suspected to contribute to all three kinds of instabilities. Concerning the step-meander instability, the intuition is the following. Given a step with a bulge, adatoms coming from the lower terrace will, due to the diffusive nature of their movement,



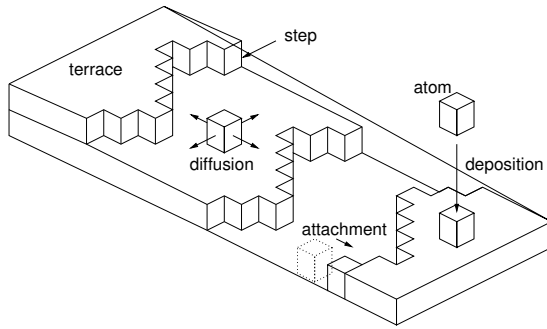


Figure 6.3: Microscopic processes: Vapour atoms are deposited on a surface, where they become ad(sorbed)atoms and diffuse on the flat terraces; eventually adatoms attach to steps.

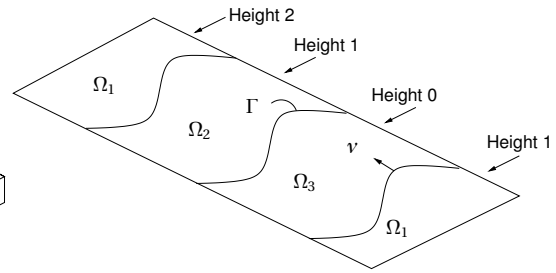


Figure 6.4: Step-flow model: steps are treated as smooth curves Γ and every terrace becomes a separate domain with a height assigned to it.

be more likely to attach to the bulge. In the same way, adatoms coming from the upper terrace will be more likely to attach to the rear part of the step. Therefore, if the adatoms from the lower terrace attach with a higher rate, the bulge will grow.

The results of simulations of this instability are presented in Section 6.5.2 for the onset of growth and in Section 6.6 for the grown instability.

6.3 Step-flow model

The layer-by-layer growth of the crystalline surface is usually described using the classical Burton–Cabrera–Frank (BCF) model [Burton et al., 1951], which is a semi-continuous model: discrete in the height, but continuous in the lateral directions.

We start from a simplified lattice-gas point of view with lattice spacing a , see Figure 6.3. Adatoms are deposited with rate f per site and hop from one site to another with rate D until they reach a step. They can then attach to the step and also detach again. Desorption of adatoms has been neglected, which is valid in typical MBE experiments [Maroutian et al., 1999].

In the step-flow model, atomistic hops on terraces are modelled by a continuum diffusion equation for the adatom density. The atomistic processes of attachment and detachment at the steps are incorporated by appropriate boundary conditions. Moreover, the atomistically rough steps are treated as smooth curves Γ , see Figure 6.4, and the local geometry enters via the curvature.

If nucleation of new islands or steps on the terraces can be neglected (which we will assume in the following), the growth dynamics are essentially described by the attachment kinetics at the steps, i.e. the boundary conditions of the adatom density at the terrace boundaries. This leads to a free boundary problem for the adatom densities on the terraces with free

boundaries given by the step position. On each terrace Ω_i , the adatom density ρ obeys the diffusion equation $\partial_t \rho - Da^2 \Delta \rho = f$.

We will focus on a regime where the adatom density has enough time to relax to its quasi-stationary equilibrium on the terraces. Thus, the diffusion equation can be replaced by

$$-Da^2 \Delta \rho = f. \quad (6.1a)$$

The fluxes of adatoms to a step are given by

$$j^\pm = \pm Da \nabla \rho^\pm \cdot \nu, \quad (6.1b)$$

where “+” and “−” denote quantities at a step up (i.e. on the lower terrace) and a step down, respectively, and ν denotes the normal pointing from an upper to a lower terrace. Since adatoms not only attach to steps, but also detach from steps due to thermodynamical effects, there is an equilibrium density ρ^* for an infinite straight step. For curved steps, the equilibrium density has to be modified by the curvature of the step, which leads to the linearized Gibbs–Thomson relation

$$\rho_{\text{eq}} = \rho^*(1 + \xi \kappa),$$

where ξ is the capillary length and κ is the curvature. We define the curvature of a convex island to be positive.

Assuming first order kinetics for the attachment/detachment of adatoms at the steps, the fluxes at the steps (terrace boundaries) are proportional to the deviation of the adatom density from equilibrium, i.e. the adatom density satisfies the following *kinetic boundary conditions* at a step:

$$j^\pm = k^\pm(\rho^\pm - \rho_{\text{eq}}). \quad (6.1c)$$

With this notation, asymmetric attachment rates $0 < k_- < k_+$ model the Ehrlich–Schwoebel barrier. Finally, the normal velocity of a step is given by

$$\frac{1}{a} V = j^+ + j^-. \quad (6.1d)$$

For a more detailed description of the step-flow model see e.g. Krug [2005].

6.3.1 Nondimensionalization

Before we proceed to the diffuse-interface approximation, we rewrite equations (6.1) to render them more concise. We first state the equations in terms of the *excess adatom density*

$$w := \rho - \rho^*.$$

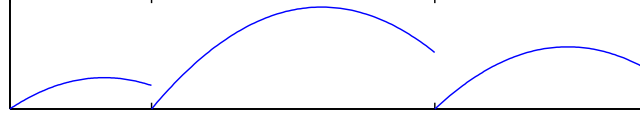


Figure 6.5: Solution w of the one-dimensional version of (6.2) in the case of descending steps. Due to the ES barrier, not all adatoms coming from an upper terrace (here in each case left of a step) attach to the step. Therefore, the excess density w is positive on the upper terrace and jumps to zero at the step.

Similar to Otto et al. [2004], we rescale units according to

$$\begin{aligned} t &= F^{-1}\hat{t}, & x &= (DF^{-1}\rho^*a^2\xi)^{1/3}\hat{x}, \\ f &= F\hat{f}, & w &= (D^{-1}F(\rho^*)^2a^{-2}\xi^2)^{1/3}\hat{w} \end{aligned}$$

and get in the new variables a *Mullins–Sekerka-type* free-boundary problem:

$$\begin{aligned} -\hat{\Delta}\hat{w} &= \hat{f} & \text{in } \Omega_i, \\ \hat{w}^\pm &= \hat{\kappa} \pm \zeta^\pm \hat{\nabla}\hat{w}^\pm \cdot \nu & \text{on } \Gamma, \\ \hat{V} &= \hat{\nabla}(w^+ - w^-) \cdot \nu & \text{on } \Gamma. \end{aligned}$$

The dimensionless parameters ζ^\pm are anti-proportional to the attachment rates k_\pm :

$$\zeta^\pm = \frac{1}{k^\pm} (FD^2(\rho^*)^{-1}a\xi^{-1})^{1/3}.$$

In the following, we will only consider the case of unlimited attachment to a step up, i.e. $\zeta^+ = 0$, and we drop the hats, so we finally get

$$-\Delta w = f \quad \text{in } \Omega_i, \quad (6.2a)$$

$$w^+ = \kappa \quad \text{on } \Gamma, \quad (6.2b)$$

$$w^- + \zeta^- \frac{\partial w^-}{\partial \nu} = \kappa \quad \text{on } \Gamma, \quad (6.2c)$$

$$V = \frac{\partial w^+}{\partial \nu} - \frac{\partial w^-}{\partial \nu} \quad \text{on } \Gamma. \quad (6.2d)$$

6.3.2 One-dimensional solution

To get a flavour of the solutions of the equation, consider the one-dimensional version with steps at position $x = 0$ and $x = L$. The solution of (6.2) is then the quadratic function

$$w(x) = -\frac{f}{2}x^2 + \frac{f}{2} \frac{L^2 + 2L\zeta^-}{L + \zeta^-},$$

see Figure 6.5 for an example. Note that, due to the Ehrlich–Schwoebel barrier, the solution w jumps at the steps.

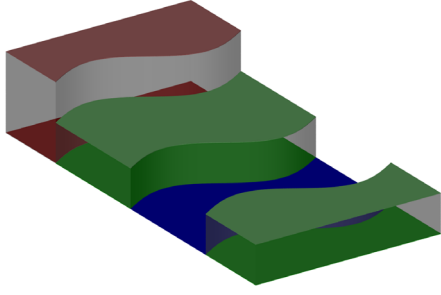


Figure 6.6: BCF model: each domain is associated with a discrete height, thus forming a three-dimensional landscape with sharp interfaces.

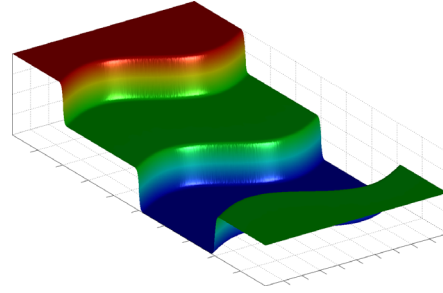


Figure 6.7: Diffuse-interface approximation: the sharp interfaces are "smeared out", resulting in a smooth function.

6.4 Diffuse-interface approximation

Let us first show how to imagine a diffuse-interface approximation: the BCF model is two-dimensional, but every domain Ω_i is labelled with a discrete height, so one can imagine it as a three-dimensional landscape with sharp jumps, see Figure 6.6. The diffuse-interface approximation can now be thought-of as a smeared-out version of this landscape, where the sharp jumps are replaced by smooth transition regions of width ε , see Figure 6.7.

The connection between diffuse-interface models in form of a Cahn–Hilliard-type equation and their sharp-interface limits were discussed in Section 2.6.2.

For step-flow growth, diffuse-interface approximations have already been introduced in Liu and Metiu [1997] and Karma and Plapp [1998], but none of these included the Ehrlich–Schwoebel-barrier.

We come now to a model which does include the ES-barrier. Consider the Cahn–Hilliard-type equation

$$\partial_t \phi + \nabla \cdot J = f, \quad (6.3a)$$

$$\frac{1}{M(\phi)} J = -\nabla \frac{\delta E_\varepsilon}{\delta \phi}(\phi), \quad (6.3b)$$

where $E_\varepsilon(\phi)$ is the Ginzburg–Landau free energy with a double-well potential G

$$E_\varepsilon(\phi) = \int_\Omega \frac{\varepsilon}{2} |\nabla \phi|^2 + \varepsilon^{-1} G(\phi), \quad G(\phi) = 18\phi^2(1-\phi)^2 \quad (6.3c)$$

and $M(\phi)$ is a mobility function modelling the Ehrlich–Schwoebel barrier:

$$M(\phi) = (1 + \varepsilon^{-1} \zeta^- \sigma(\phi))^{-1},$$

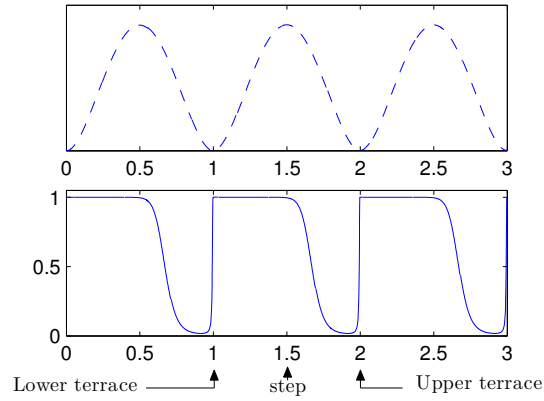


Figure 6.8: The potential G (dashed line) and the mobility function M (solid line): coming from an upper terrace (“ $\phi = 2$ ”), the atoms experience reduced mobility while attaching to the step (“ $\phi = 1.5$ ”). On the other hand, coming from a lower terrace to the step, there is no reduction in mobility. This models the Ehrlich–Schwoebel barrier.

where $\sigma(\phi)$ is an asymmetric function in ϕ . See Figure 6.8 for a plot of the potential and the mobility. The potential G is restricted to the interval $[0, 1]$ and then periodically continued, so that we get a multi-well potential with equally deep wells at the integers. As we have seen in Section 2.6, this yields the anticipated structure, namely flat terraces at the integers, corresponding to the atom-high monolayers, with thin transition regions in-between.

In Otto, Penzler, Rätz, Rump, and Voigt [2004] it was shown by formal asymptotic expansion that the above equation yields for $\varepsilon \rightarrow 0$ a slightly more general form of the BCF-model (6.2):

$$-\Delta w = f \quad \text{in } \Omega_i, \quad (6.4a)$$

$$\begin{pmatrix} z^+ & z^m \\ z^m & z^- \end{pmatrix} \begin{pmatrix} \nabla w^+ \cdot \nu \\ \nabla w^- \cdot \nu \end{pmatrix} = \begin{pmatrix} w^+ - \kappa \\ -w^- + \kappa \end{pmatrix} \quad \text{on } \Gamma, \quad (6.4b)$$

$$V = \frac{\partial w^+}{\partial \nu} - \frac{\partial w^-}{\partial \nu} \quad \text{on } \Gamma, \quad (6.4c)$$

where the coefficients are given by

$$\begin{aligned} z^+ &= \zeta^- \int_0^1 (1-\phi)^2 \frac{\sigma(\phi)}{\sqrt{2G(\phi)}} d\phi, \\ z^m &= \zeta^- \int_0^1 \phi(1-\phi) \frac{\sigma(\phi)}{\sqrt{2G(\phi)}} d\phi, \\ z^- &= \zeta^- \int_0^1 \phi^2 \frac{\sigma(\phi)}{\sqrt{2G(\phi)}} d\phi. \end{aligned}$$

Note that, as long as the coefficient matrix is positive semi-definite, this still yields a thermodynamically consistent evolution, i.e. the length of Γ (and therefore the energy) decreases.

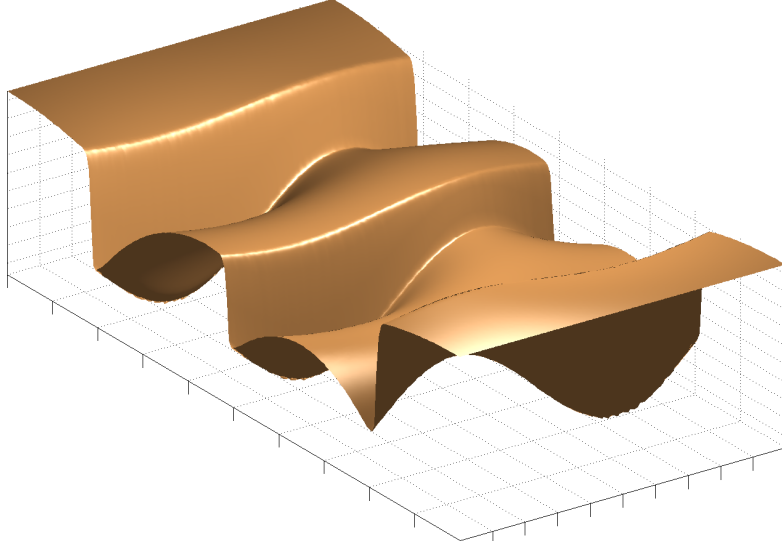


Figure 6.9: Excess density w for data as in Figure 6.7. Clearly visible are the boundary values $w = \kappa$ at a step up, the jump due to the ES-barrier and the smooth solution of $-\Delta w = 1$ on the terraces.

For our numerics, we choose

$$\sigma(\phi) = 6(p+4)(p+5)\phi^p\phi^2(1-\phi)^2, \quad p \gg 1.$$

Together with the definition of G , this yields the coefficient matrix

$$\begin{pmatrix} z^+ & z^m \\ z^m & z^- \end{pmatrix} = \begin{pmatrix} \frac{6}{p^2+5p+6} & \frac{2}{p+3} \\ \frac{2}{p+3} & 1 \end{pmatrix} \zeta^-.$$

Henceforth, we will call the system (6.4) with these coefficients the p -BCF model. The BCF model (6.2) has the coefficients $z^+ = z^m = 0$, $z^- = \zeta^-$.

Thus, the connection between the BCF model, the p -BCF model and the diffuse-interface approximation (DIA) is given by the following limits:



In the diffuse-interface approximation, the position of the boundary Γ (the steps), is given by the level sets

$$\left\{ \phi = \mathbb{Z} + \frac{1}{2} \right\}.$$

Note that the L^2 -differential of the energy

$$\frac{\delta E_\varepsilon}{\delta \phi}(\phi) =: w_\varepsilon$$

is the approximation of the excess density w from equations (6.2), see Figure 6.9.

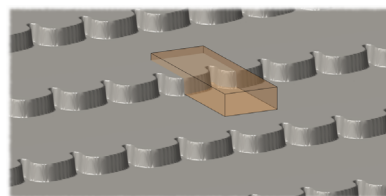
6.5 Numerical tests

In this section, we use the discretization presented in Chapters 3 and 4 to simulate the Cahn–Hilliard-type equation (6.3), which is our diffuse-interface approximation.

Before we come to the simulation of the step meander instability in Section 6.6, we perform some tests to be confident that the diffuse-interface approximation is indeed a good approximation to the BCF model for feasible values of ε and p .

The simulations aim to reflect the growth of a small section of a larger working piece, so a periodic setup is necessary to exclude boundary effects. On the other hand, the steps should all move in one direction or else the valleys would be filled up after a short time. Therefore, a very useful setup for simulations in epitaxial growth is an infinite descending step train, see Figure 6.2 on page 111.

This leads us to an important feature of our software. Often several monolayers have to be deposited until an interesting feature becomes apparent. Assume as an example that we have to wait for the deposition of five monolayers. Then, to simulate the step train, one has to use a computational domain having at least five times the length of the terrace. To exclude boundary effects, it should be even larger. As a way out of this, we use “skew periodic” meshes. That is, having a rectangular computational domain, we can connect the right side of the domain having height, say, one to the left side with height two and the program will take care of the jump in the solution, see Section 7.2.2 for the implementation. Using this feature, it is enough to have a computational domain of the length of one terrace.



6.5.1 Straight step train

As a first test, we consider a straight step train without ES barrier, i.e. with $\zeta^- = 0$, and compare the results to the solution of the BCF-model. The amount of deposited material on a rectangular unit cell $\Omega = [0, L_x] \times [0, L_y]$ after a time t is fL_xL_yt . Due to mass conservation, the step has to move with speed fL_x .

To obtain the speed of the step in the diffuse-interface approximation, we have to determine the step position, that is the level set $\{\phi = 1/2\}$. Since the discrete function ϕ_h is piecewise constant, determining the step position is not trivial. Therefore, this test is more a test of the quality of the step-finding algorithm than a test of the approximation to the BCF model. The details on the algorithm are lined out in Section A.3

The result can be seen in Figure A.2: the step moves indeed with speed fL_x . Deviations come from determining the exact position of the step. Moreover, it takes some time until the excess density w_ε has built up, so the step moves slower at the beginning.

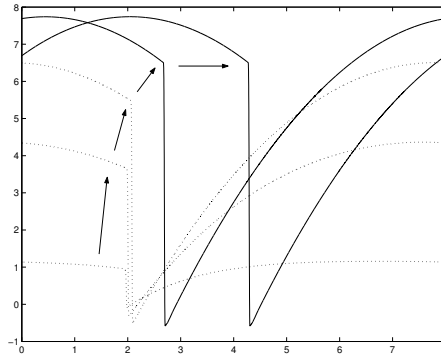


Figure 6.10: At the beginning of a simulation, it takes some time for w_ε to build up (dotted lines). Only then, the step moves with the correct speed.

6.5.2 Linear instability of a step train

Now we come to the simulation of the instability explained in Section 6.2. At the onset of the growth of the instability, the behaviour can be predicted using linear stability analysis.

The setup for the simulation is as follows. We consider a planar equidistant descending step train, i.e. an infinite sequence of straight steps down with distance L_x between one step and the next. Each step is modified with the same small lateral perturbation. Since perturbations get in-phase very quickly, it is enough to consider only perturbations that are already in-phase, i.e. identical for each step. Hence it is natural to work in a periodic setup with periodic cell $[0, L_x] \times [0, L_y]$, where L_x is the step spacing and L_y is a period of the perturbation, see Figure 6.11. We will speak of *step meandering* if the initial perturbation increases under growth, so that the steps do not stay straight.

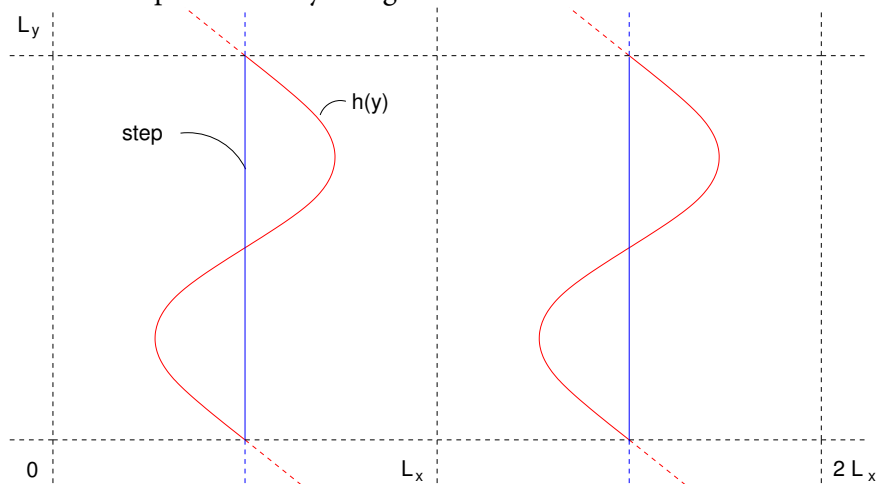


Figure 6.11: Top view of the unit cell $[0, L_x] \times [0, L_y]$: each step is perturbed by $\delta h(y)$, where $h(y)$ is a function of order one and $\delta \ll 1$.

Linear stability analysis

The following linear stability analysis is based on the original BCF model (6.2) and follows that of Bales and Zangwill [1990].

First, we insert an δ -expansion of w , i.e. $w = w_0 + \delta w_1 + \mathcal{O}(\delta^2)$, into the equations to derive the δ -order evolution of the boundary, which is the evolution of the perturbation $h(t, y)$. The resulting $\mathcal{O}(\delta)$ -equations are

$$\begin{aligned} -\Delta w_1 &= 0 && \text{on terrace,} \\ \partial_t h &= \partial_x w_1^+ - \partial_x w_1^- && \text{at step,} \\ w_1^+ &= -ch - \partial_{yy} h && \text{at step up,} \\ \zeta^- \partial_x w_1^- + w_1^- &= -(c - fL_x - f\zeta^-)h - \partial_{yy} h && \text{at step down,} \end{aligned}$$

where $c = f(\zeta^- L_x + \frac{1}{2}L_x^2)/(\zeta^- + L_x)$. Fourier transformation in y yields equations for w_1 and h , which are now functions of (x, k) and k , respectively:

$$\begin{aligned} -\partial_{xx} w_1 + |k|^2 w_1 &= 0, \\ \partial_t h &= \partial_x w_1^+ - \partial_x w_1^-, \\ w_1^+ &= (|k|^2 - c)h, \\ \zeta^- \partial_x w_1^- + w_1^- &= (f\zeta^- - c + fL_x + |k|^2)h. \end{aligned}$$

This motivates the Ansatz

$$w_1(x, k) := c_1 \exp(|k|x) + c_2 \exp(-|k|x).$$

Straightforward calculations for c_1 and c_2 yield for each wave vector k

$$\partial_t h_k(t) = c_1 |k| (1 - \exp(|k|L_x)) - c_2 |k| (1 - \exp(-|k|L_x)) = \omega(k) h_k(t)$$

with the dispersion relation

$$\omega(k) = \frac{-|k| \left[(|k|^2 - c) |k| \zeta^- \sinh(|k|L_x) + (2|k|^2 - 2c + f(\zeta^- + L_x)) (\cosh(|k|L_x) - 1) \right]}{\sinh(|k|L_x) + |k| \zeta^- \cosh(|k|L_x)}. \quad (6.5)$$

The calculations for the p -BCF model (6.4) are analogous. The resulting dispersion relation is given by

$$\begin{aligned} \omega_p(k) &= \frac{-|k|^2 \sinh(|k|L_x) \left[(|k|^2 - c_p)(z^+ + 2z^m + z^-) + fL_x(z^m + z^+) \right]}{\sinh(|k|L_x)(z^+ z^- |k|^2 - (z^m |k|)^2 + 1) + |k| \cosh(|k|L_x)(z^- + z^+) + 2z^m |k|} \\ &\quad - \frac{|k| (\cosh(|k|L_x) - 1) \left[2(|k|^2 - c_p) + f(z^- - z^+ + L_x) \right]}{\sinh(|k|L_x)(z^+ z^- |k|^2 - (z^m |k|)^2 + 1) + |k| \cosh(|k|L_x)(z^- + z^+) + 2z^m |k|} \end{aligned} \quad (6.6)$$

with $c_p := fL_x \frac{z^- + z^m + L_x/2}{z^- + 2z^m + z^+ + L_x}$.

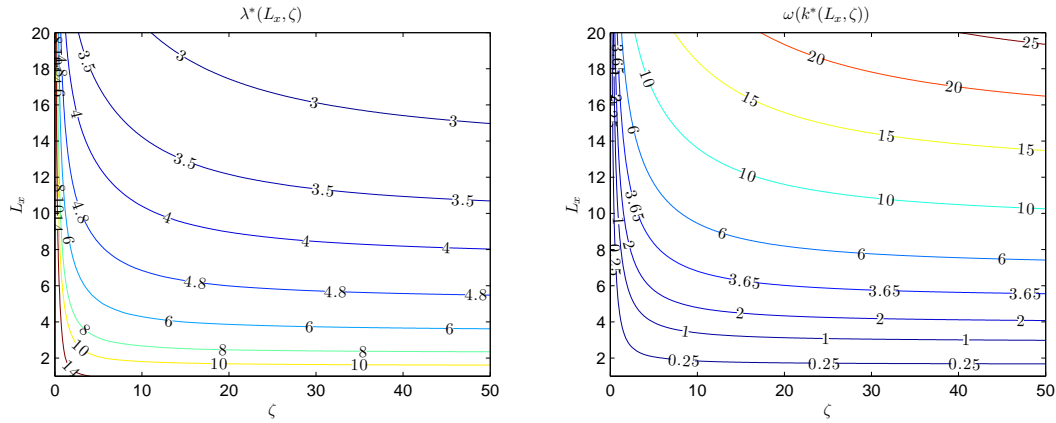


Figure 6.12: Contour lines of the most unstable wave length λ^* and the corresponding growth factor ω .

Note that due to the linearization, the dispersion relation for a given wave number $|k|$ is independent of all other wave numbers. This permits us to decompose the perturbation in perturbations of just a single wave number and analyze only one of those.

As can be seen from Equation 6.5, for $\zeta^- = 0$ or $f = 0$ we have $\omega(k) < 0$ for all k , so that any initial perturbation will flatten out during growth. Only if $\zeta^- > 0$ and $f > 0$, there is a region of wave numbers for which $\omega(k) > 0$. In this case an instability will develop.

Choice of parameters

Using the formula for the BCF-model (6.5) and setting $f = 1$, we have only two parameters left: L_x and ζ . For each pair of these, we can find the most unstable wavelength λ^* and the most unstable wavenumber k^* , see Figure 6.12. When choosing L_x and ζ , we have a few things in mind:

- A large unstable wavelength means a long step (L_y large) and therefore many degrees of freedom.
- If the corresponding growth factor ω is too small, the instability grows very slowly, so the simulation has to run for a very long time. On the other hand, if the growth factor is too big, then we leave the linear regime too quickly.
- If L_x is too large, the excess density w is large on the terraces. In the asymptotic analysis in Otto et al. [2004], we have seen that $\phi \approx \mathbb{Z} + \varepsilon w$ on the terraces and ε should be chosen so that $\varepsilon w \ll 1$.
- If L_x is large, the wave number $k = nL_x/2\pi$ for the n -th mode is also large. In most cases, $\omega(k)$ is only positive for $n = 1$ and so only the first mode will grow.

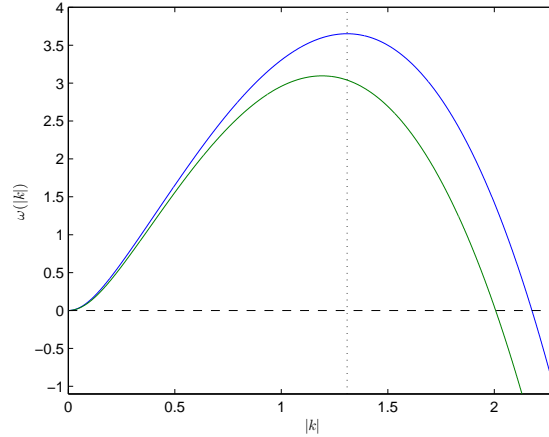


Figure 6.13: Plot of the growth factors for the BCF-model (blue) and the p -BCF-model (green). The most unstable wave number for BCF is at the dotted line.

Considering these conditions, we choose $\zeta^- = 13.5308$ and $L_x = 6.4$. For these values, we get the growth factors depicted in Figure 6.13 and find

$$\lambda^* = 4.8, \quad k^* \approx 1.31 \quad \text{and} \quad \omega(k^*) \approx 3.65.$$

Since the diffuse-interface approximation approximates the p -BCF model, we will compare the numerically measured growth factors to the value

$$\omega_p(k^*) \approx 3.04.$$

Numerical results

For our numerical simulations, we use the setup shown in Figure 6.11. As outlined above, we choose $L_x = 6.4$. For the vertical size, we take the most unstable wave number, i.e. $L_y = 4.8$. As initial values, we use the one-dimensional minimizing profile in x direction, where the step position ξ (that is $\phi_0(\xi, \cdot) = 1/2$) was modified by a function $h(y)$:

$$\phi_0(x, y) = \frac{1}{2} \left(1 - \tanh \left(\frac{3}{\varepsilon} (x - \xi + \delta h(y)) \right) \right)$$

The amplitude of the initial perturbation is $\delta = 0.005$. The function $h(y)$ was either a sinus function, so a fixed wave number, or random values in the range $[-1, 1]$.

To find the dispersion relation, or growth factor, we determine the step-position after each time step, see Section A.3 for the algorithm. Then the amplitudes of all modes are calculated by Fourier transformation. Finally, the numerical growth factor ω_{num} was determined by linear fit to the logarithm of the amplitudes.

The first series of simulations shows the convergence of ω_{num} to ω_p when ε decreases, see Figure 6.15. The initial values use a step position modified by a mode-one perturbation. We see good agreement for $\varepsilon \leq 1/16$.

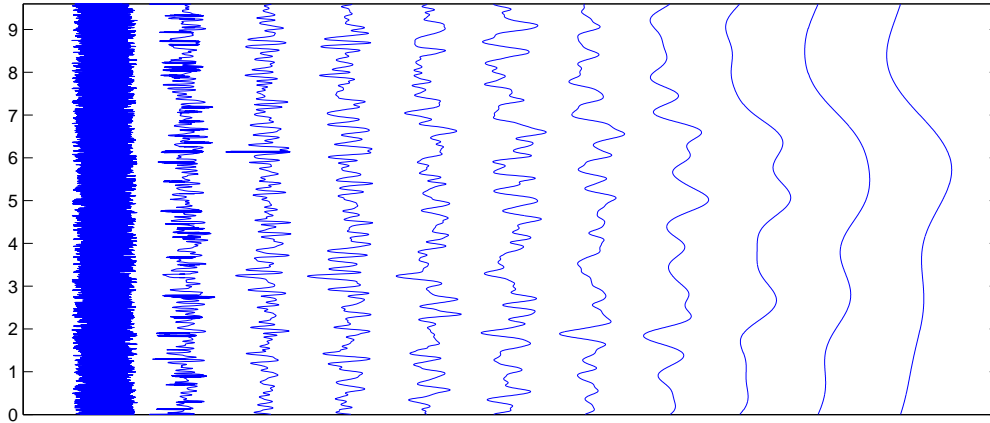


Figure 6.14: The shape of a randomly perturbed step at different (logarithmically equidistant) times. Note that the x -axis does not display the position of the step and that the amplitudes of the steps were scaled to be equal.

With this value for ε , we started four simulations with step position perturbed by mode one to four, respectively, and a fifth one with randomly perturbed step position. For each, we measure the growth factor and find good agreement with the predicted values, see Figure 6.16. In Figure 6.14, we depicted the development of the randomly perturbed initial values.

To avoid that just the biggest possible wave number grows, we performed additional simulations with $L_y = 9.6$ for comparison, which confirmed the results.

6.5.3 Linear instability of a circular island

A further test is the growth of a circular island [Li et al., 2004]. The domain here is a ball of finite radius and the initial values are a circular island, where the radius is perturbed with a periodic function of the angle, i.e.

$$r(\theta) = r_0 + \delta \sin(n\theta), \quad \theta \in [0, 2\pi), n \in \mathbb{N}. \quad (6.7)$$

In contrast to the step train, the growth factor here is not constant, but changes over time.

As already lined out in Rätz [2007], this test is unfavorable for a diffuse-interface approximation, as the density w_ε needs some time to build up. Therefore, the growth factor “lags behind” and the agreement between theory and numerical solution is suboptimal. Additionally, the step gets longer during growth, so the number of unknowns increases constantly.

Another issue in our simulation was that the perturbation grew so large that nonlinear effects appeared. Nevertheless, we show the results to see that our implementation works fine with no-flux boundary conditions, a delaunay macro-grid and up to a million degrees of freedom.

The simulation was run with a circular domain of radius 13.6 with no-flux boundary conditions and initial values as in (6.7) with $r_0 = 2.7$, $\delta = 0.1$ and $n = 6$. The mesh was generated in Matlab and, as usual, refined by bisection. At $t \approx 0.96$, the domain will be completely filled

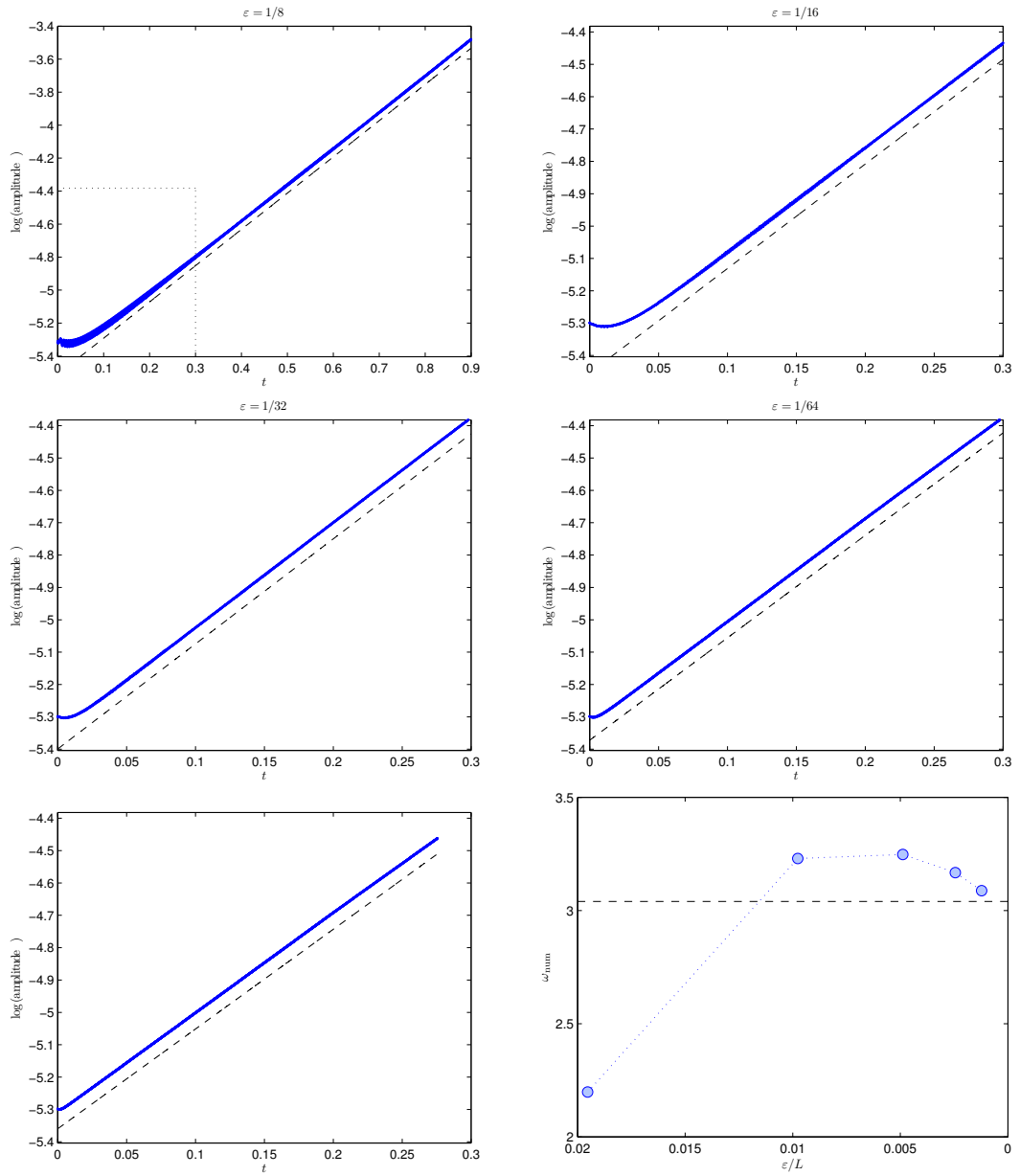


Figure 6.15: Plot of the amplitude of mode one for different values of ε . For $\varepsilon = 1/8$ (top left), the simulation had to run longer to get a usable slope. The lower right plot shows the growth factors with compared to the predicted factor from the p -BCF model. For $\varepsilon \leq 1/16$, we see good agreement. An explanation for the slight oscillation of the amplitude can be found in Section A.3.

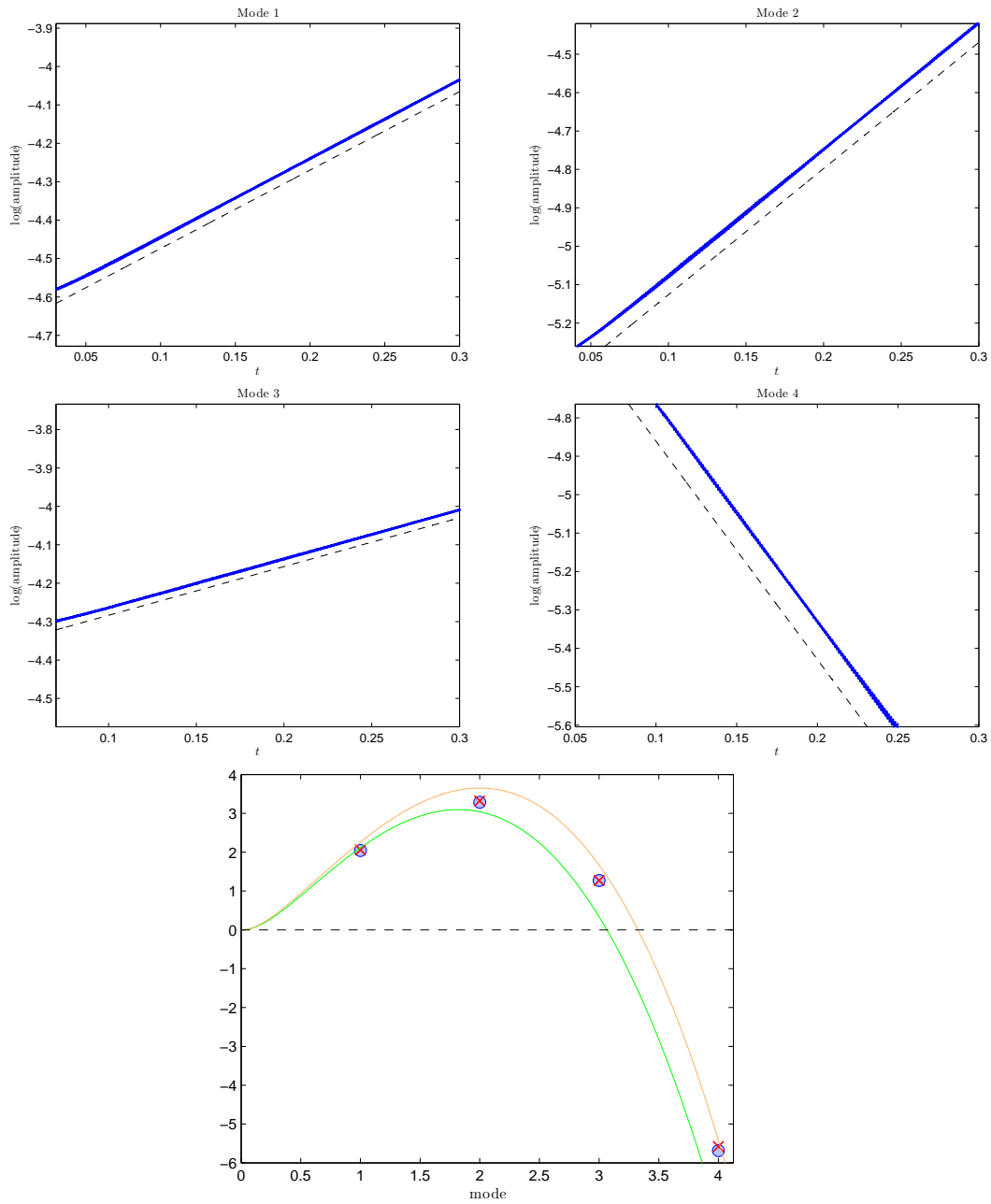


Figure 6.16: Plot of the amplitude of modes one to four for $\varepsilon \approx 0.01$. In the lowermost plot, ω (orange), ω_p (green) and the numerical values for fixed-mode initial values (blue dots) and white-noise initial values (red crosses) are shown. Again, we find good agreement.

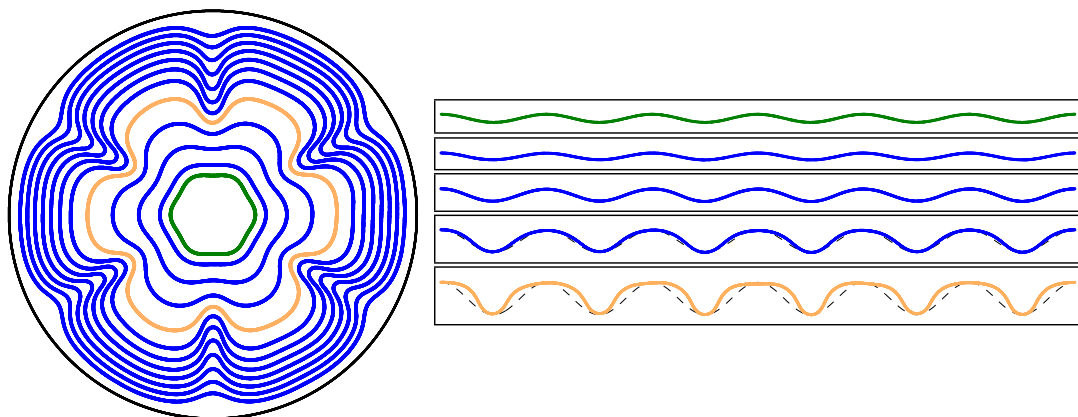


Figure 6.17: On the left, the step position for different times is shown. On the right, we show the deviation from a circle, i.e. the perturbation. A sine of the same amplitude was added for comparison (dashed line). We see that the initial perturbation (green) grows first and later shrinks again. The amplitude of the perturbation gets quite large, so that nonlinear effects appear (orange) and the perturbation no longer has a sinusoidal shape.

up and the simulation stops. The initial values were resolved with around 85 000 degrees of freedom. Since the step becomes longer during the simulation, the number of degrees of freedom increased up to around 950 000 just before the end.

In Figure 6.17, we plotted the position of the step at different times and the development of the perturbation. We see that the perturbation grows first and then shrinks again. However, due to nonlinear effects, the perturbation loses its sinusoidal shape. Therefore, linear stability analysis does not apply here and thus we do not compare the numerical growth factor to the theory.

A piece of the diffuse-interface approximation including the mesh is shown in Figure 6.18 on the facing page.

6.6 Nonlinear instability of a step train

If we continue the simulation of Section 6.5.2 and the instability grows larger, we leave the linear regime. The following nonlinear regime is of much more importance for practical purposes, because meandering patterns observed during growth show large amplitudes. Depending on the parameters used, one observes one of the following [Pierre-Louis et al., 1998; Danker et al., 2003; Pierre-Louis et al., 2005]:

- Endless growth of the meander amplitude.
- Stationary step profiles with a fixed amplitude.
- The step develops overhangs that eventually lead to a pinch-off of a vacancy island, that is a void of the depth of one atomic height.

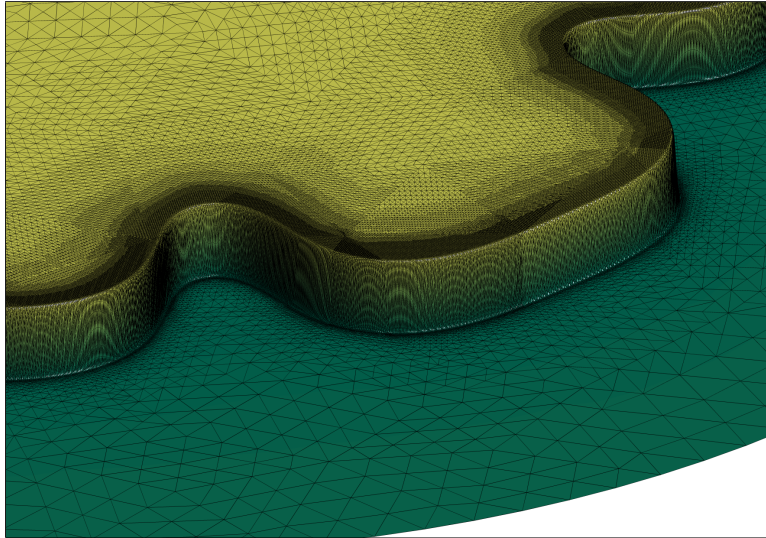


Figure 6.18: The diffuse-interface approximation corresponding to the orange step in Figure 6.17 together with the computational mesh.

Haußer and Voigt [2007] carried out a parameter study and found all three predicted behaviours. They used a front-tracking method for the simulations. This kind of method is based on the BCF model and calculates in each time step the solution of the Poisson problem given the boundary of the last step. Then, the step is promoted with the velocity determined by the density field. Two different meshes are necessary for the front-tracking method: a two-dimensional mesh to compute the density w and a one-dimensional mesh parameterizing the step. A major handicap of this type of method is that it breaks down if the step self-intersects as is the case with the pinch-off.

Since topological changes impose no problems for the diffuse-interface formulation, we can use our software to study the behavior after the pinch-off. We use the same parameters as Haußer and Voigt. In the nondimensional form, they translate into

$$\zeta^- \approx 94.1, \quad \hat{l} \approx 9.41 \quad \text{and} \quad \hat{\lambda} \approx 3.64.$$

In Figure 6.19, the step position extracted of our simulation is compared to the step position calculated with the front-tracking method just before the pinch-off, where the front-tracking has to stop. We see very good agreement.

In Figure 6.20 we can see what happens after the pinch-off: vacancies appear and are filled up again by the newly deposited atoms. They disappear just before the next step arrives, so that we get a periodic “production” of vacancies.

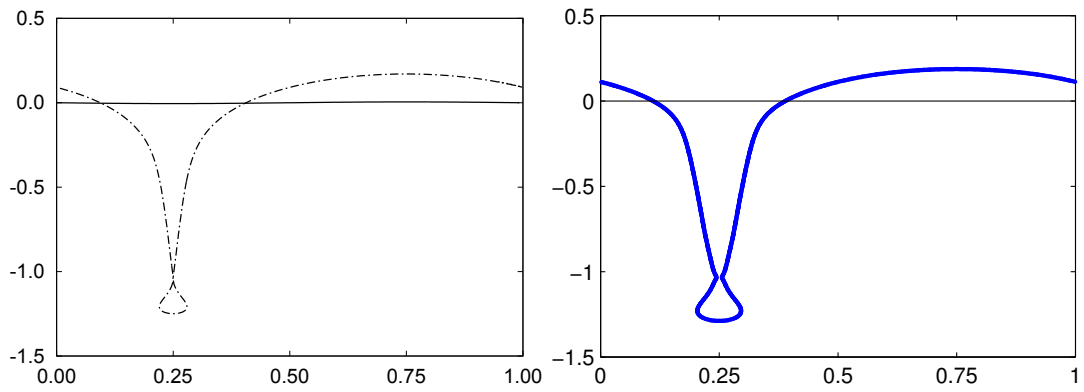


Figure 6.19: Comparison of the step position just before the pinch-off for the front-tracking method (left) and our diffuse-interface approximation (right).

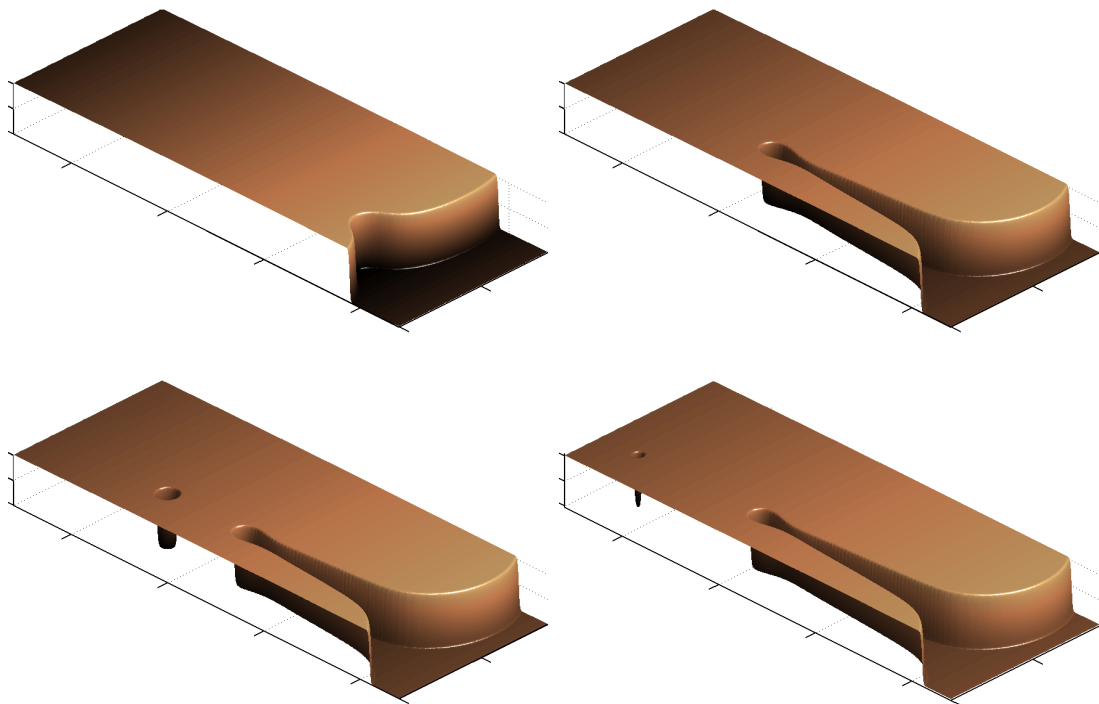


Figure 6.20: Nonlinear instability: the instability of a straight step grows larger (upper left) and develops overhangs (upper right) until a vacancy pinches off (lower left). The vacancy is filled up before the next step arrives (lower right), so that we see a periodic release of vacancy islands.

The finite element software

The simulations in this work were performed using our adaptive finite element software CHEST (Cahn Hilliard Equation Simulation Tool). The software was developed as part of this work. It was written mostly from scratch. There were two main reasons to develop a new software and not to use an already existant one. First, most finite element toolboxes do not include Raviart–Thomas finite elements. Second, and more important, the lack of support for skew-periodic meshes. Adding support for such meshes to an existing software is an extremely complex task. However without such meshes, simulating step trains as in Chapter 6 is hardly possible.

The software was written in C++ and consists of approximately 25 000 lines of code. Its organization is shown in Figure 7.1. We describe the components in the following sections.

There are a few parts that were not written by myself. The mesh library is based on code by Ralf Hiptmair developed during a seminar on the boundary element method. The Eikonal solver used in the mesh adaption is based on code by Jörg Drwenski. To accelerate the linear solver, we use the packages PETSc [Balay et al., 2001], CHOLMOD [Davis and Hager, 2005] and UMFPACK [Davis, 2004].

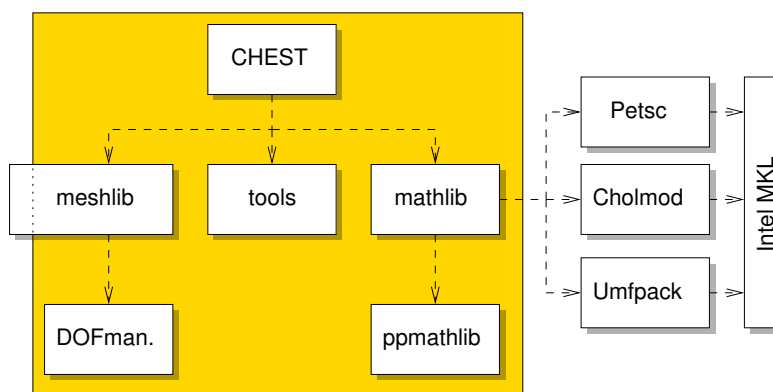


Figure 7.1: Components of the finite element software CHEST. The shaded components were written as part of this work.

7.1 The math library

First, we programmed our own math library consisting of classes for vectors, sparse and full matrices, and linear systems of equations.

Later, when the performance of the software became important, the wish to use external packages for the linear solver came up. With packages as PETSc, it is easy to try out different linear solvers and preconditioners. It turned then out that the performance of the Cholesky solver in the PETSc package (at least in version 2.3.3) is *much* slower and needs more memory than the solver in the Cholmod package. Finally, direct methods for non-symmetric problems seem to be quite fast with Umfpack. All three have in common that they rely on the basic linear algebra subrutines (BLAS). As our group uses computers with Intel processors, we use the Intel math kernel library (MKL) as implementation of BLAS.

Of course, all these packages use different classes for the matrices. Therefore, our math library is by now mostly used as an interface to the external packages, which also translates between the different matrix structures.

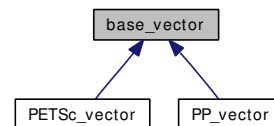
However, there are cases where our own implementation of a sparse matrix is faster due to its less advanced memory management, see Section 7.1.2.

Note that if we speak of arrays in the following, they are of course all dynamically allocated.

7.1.1 Vectors

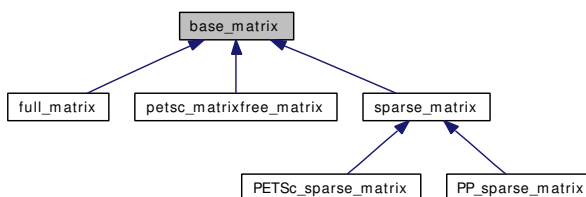
The base class for vectors has the non-surprising name `base_vector`. It is an abstract base class with virtual functions for the basic operations as data access and the like.

Our self-written vector class uses a simple double array for data storage and implements the functions from `base_vector`.



7.1.2 Matrices

The structure of the matrix class can be seen on the right. The `base_matrix` class is again an abstract base class with methods for data access, multiplication with a vector, scaling, and so on. The `full_matrix` class is again a simple double array.



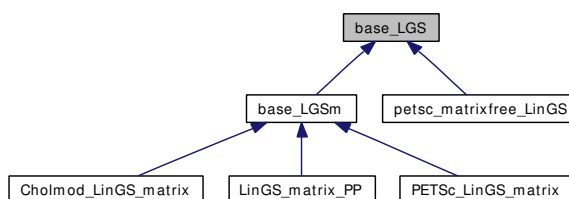
A little more effort was put into the `PP_sparse_matrix` class. It has a three-step structure: The matrix is an array of `sparse_matrix_rows` which in turn is an array of `sparse_matrix_entries`. The latter is a struct consisting of the column of the entry, the value and a used flag. We do not sort the entries, so accessing an entry requires to check the column numbers of all used

entries. In addition to the suboptimal access times, the usage flag wastes memory compared to other storage methods as the standard AIJ (or Yale) format.

The big advantage is that there is no need to perform any tasks between reading and writing entries. In Section 4.8.2 this is very useful since there a matrix is recursively filled. With the PETSc matrices, this constant alternation of reading and writing leads to a terrible performance. Thus, in this case our simple approach is faster.

7.1.3 Solving linear systems of equations

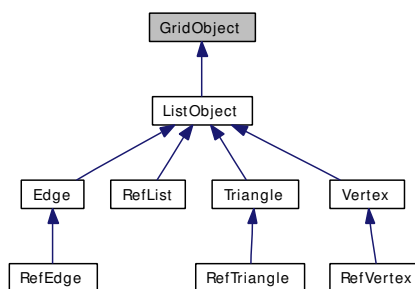
The classes for solving linear systems of equations (LSE) distinguish between LSE that are given by a matrix (`base_LGSm`) and those that are implicitly defined, i.e. only the result of a matrix multiplication is known. Both types can be solved using the iterative methods provided by PETSc. Our own solver can only use the conjugate gradient method with a given matrix as preconditioner.



The direct methods from Cholmod and Umfpack can of course only solve systems which are represented by a matrix.

7.2 The mesh library

The basis for the mesh library is a code from Ralf Hiptmair developed during a seminar on the boundary element method. It was intended for elliptic problems and therefore contained no DOF manager (DOF is short for degree of freedom), but could only load and refine a mesh. This base code was in parts rewritten and corrected, but the basic structure remained: there are linked lists (implemented by the class `ListObject`) for the Triangles, Edges and Vertices. The Ref classes take care of the “vertical” dependencies, i.e. of the father–son relations.



We largely extended this basis. The main points are the ability to communicate with a DOF manager, the addition of coarsening (more precisely unrefinement) and the support for periodic meshes, see below. An important point for the extension to periodic meshes was that the relation between triangles, edges and vertices is hierarchic and only the vertices store geometric information, see Figure 7.2. This made it easier to separate between (logical) neighbourhood relations and geometric location.

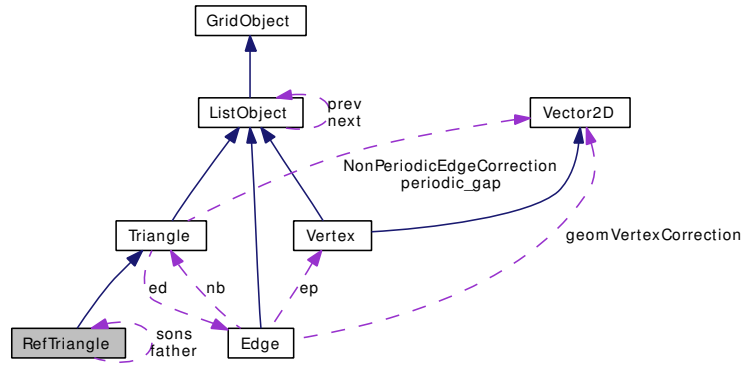
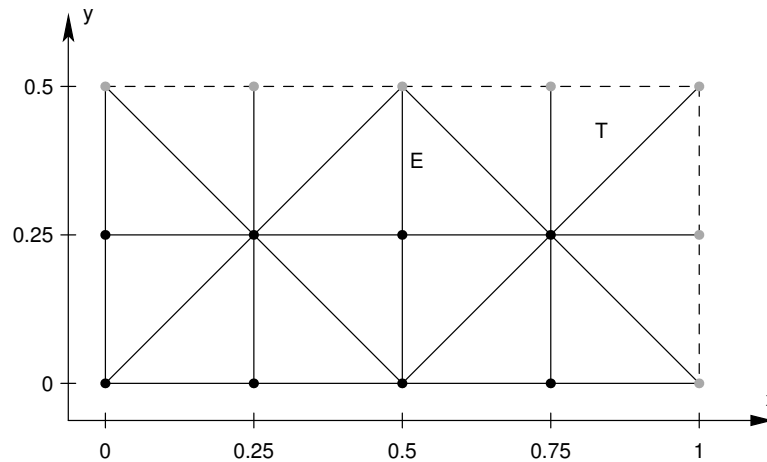


Figure 7.2: Collaboration between triangles, edges and vertices. Every triangle has three edges (*ed*) and every edge has two neighbours (*nb*), except for boundary edge who have only one neighbour. The edges have two endpoints (*ep*), the vertices. Geometric information is only stored in the vertices: a Vertex is also a Vector2D. The three properties *NonPeriodicEdgeCorrection*, *periodic_gap* and *geomVertexCorrection* are used for periodic meshes, see Section 7.2.1.

7.2.1 Periodic meshes

Periodicity only makes sense for rectangular meshes, so only those will be considered here. We distinguish between the topological properties like neighbourhood relations and geometrical properties. In the case of periodic meshes, we have geometrically a rectangle and topologically a torus.

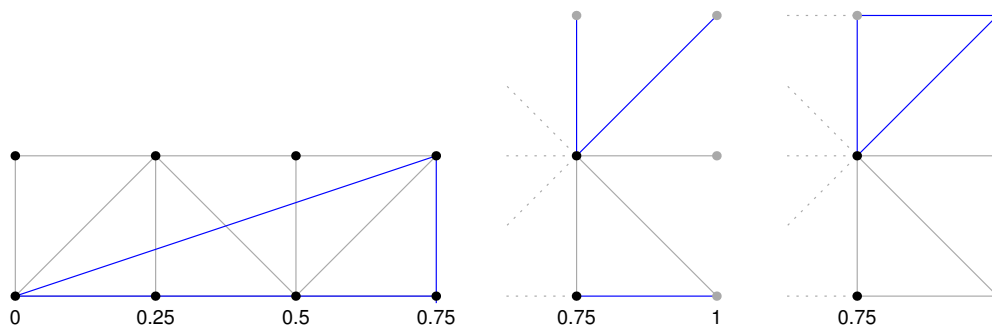
For non-periodic meshes, geometric information is only stored in the vertices, see Figure 7.2. For periodic meshes, we have to store geometric information in some other places. Consider the following periodic $[0, 1] \times [0, 0.5]$ mesh:



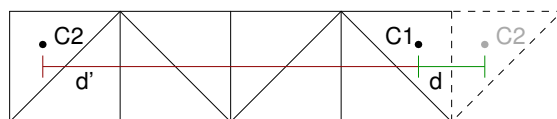
There are only eight points in the mesh, drawn black here. Now take the edge marked with *E*, connecting point $(0.5, 0.25)$ with point $(0.5, 0.5)$ in the picture. The latter point lies, by periodicity, at $(0.5, 0)$. Therefore, we add an information to the edge, named “geometric vertex

correction”, which tells us for the two endpoints how much we have to shift them to get their geometric position (the grey points).

But we need another adjustment. Consider the triangle marked T in the sketch picture. Its edges are drawn in blue in the picture below without any correction (left). After the geometric vertex correction, one edge is still at the wrong place (middle). Therefore, we add to each triangle a “geometric edge correction” for each of its three edges. This yields the final geometric position (right).



Additionally, we sometimes need to know if we jump from one side of the mesh to the other. For example, the distance from the point $C1$ to the point $C2$ in the sketch below should be d in a periodic mesh and not d' . Therefore, we add to each triangle another information for each of its three edges, the “periodic gap”.



Note that this is different from the geometric edge correction, since for example the triangles in the lower left corner both contain periodic-gap-information for one edge, but no geometric edge correction.

7.2.2 Skew-periodic meshes

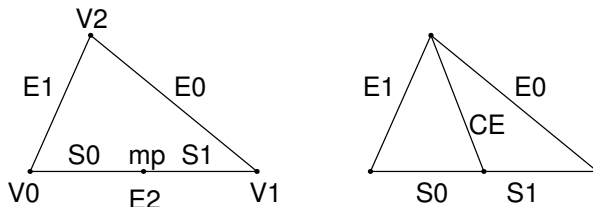
In the chapter on epitaxial growth, we have seen that it is worthwhile to investigate step-trains. This is like a infinite stair. We simulate a step-train by computing several steps on the grid and then continuing periodically. Of course, if we have n steps on the unit cell and all are steps down, then the values at $x = L$ are n units lower than the values at $x = 0$ to whom they should correspond.

We solve this problem by storing a third values for each edge of the triangle telling us by which amount the function jumps when going over the edge.

Remark. A feature of our discretization via the flux is that only ϕ jumps, but not J . So the correction for the jumps is only needed when calculating ϕ , not when solving a linear system of equations. \diamond

7.2.3 Mesh adaption

We explain here the refinement procedure at the hand of an example. To refine the triangle depicted below, we refine the longest edge, which is $E2$ in this example.



1. The edge $E2$ is split into $S0$ and $S1$, which generates a new vertex mp . The DOFmanager is notified of the refinement of the edge, so that the degrees of freedom are transferred to the sons $S0$ and $S1$. Then the edge $E2$ is removed from the vertices $V0$ and $V1$.
2. A new edge CE with endpoints $V2$ and mp is created.
3. The triangle $T0$ is removed as neighbour from edges $E0$ and $E1$.
4. Two new triangles $T1, T2$ with edges $E1, S0, CE$ and $E0, S1, CE$ are created.
5. The DOFmanager is notified of the new edge CE and a new DOF is computed as described in Section 4.9.2. This cannot happen before, because only now the edge CE has the triangles as neighbours.
6. The DOFmanager is notified of the refinement $T0$ to $T1$ and $T2$.
7. All edges of $T0$ are probed for hanging nodes. If hanging nodes are found, the neighbouring triangles are refined as well.

This procedure is repeated for all marked triangles.

Due to the refinement of neighbouring nodes to remove hanging nodes, also triangles that were not marked for refinement may be refined. This is the intended behaviour because *all* marked triangles have to be refined to improve the quality of the mesh. On the other hand, coarsening the mesh is only used to save computational work. So *at most* the marked triangles may be unrefined. Therefore, we have to avoid that triangles not marked for unrefinement are coarsened to remove hanging nodes.

The procedure to do this, named deep checking, is outlined below. The unrefinement procedure itself is similar to refinement and therefore not shown. During unrefinement, all edges and triangles that are no longer needed are destroyed.

1. All triangles marked for unrefinement are written into a list.
2. For each triangle T in the list, we iterate through the edges of the father of T . If an edge is refined, we ask if the neighbours of the sons of this edge are permitted to be unrefined, that is either

- they are marked for unrefinement or
- both sons are permitted to be unrefined.

If this recursive question is answered with yes, we continue. If the answer is no, triangle T is unmarked and we proceed.

3. If any triangle was unmarked in step 2, steps 1–3 are repeated.
4. All marked triangles are unrefined.

To improve the performance, we added a caching mechanism to the procedure.

7.3 The DOF manager

Using an adaptive mesh, one has to handle a number of objects: The mesh or grid consists (in the two-dimensional case) of triangles, edges and vertices. These will be named *grid objects* in the following. In the finite element setting, any *active* grid object may have assigned one or more degrees of freedom. Of course, inactive object like triangles belonging to a coarser grid do not have a DOF assigned. The DOF have to be continuously enumerated, so that the user can access a vector with data using the DOF numbers.

One purpose of the DOF manager is to provide the correspondence between the grid objects and the DOF numbers. A second purpose is to provide a mechanism to transfer a vector to a new grid.

For every kind of grid object, the user has to create a DOFmanager. When the DOFmanager is initialized, the user can decide if a vector with data has to be transferred to the new mesh and if so, he has to provide functions to be called upon transferring.

When a grid object is created, the user (in this case the mesh library) has to notify the DOFmanager. The object is then assigned a unique identifier (ID). When the object is destroyed, the DOFmanager has to be notified again and the ID is released for reuse.

Therefore, the core of the DOFmanager is a list which attaches to each ID a DOF. Additionally, a state like “active” or “inactive” is given to each ID.

The transfer of vectors to a new mesh takes place in three steps:

- In the first step, the user provides the old vector and starts the mesh refinement. The mesh library notifies the DOFmanager if a grid object is created, deleted, refined or coarsened. If it was refined or coarsened, then a new value is computed using the user-provided transfer functions and saved on a temporary stack.
- The second step consists only of counting the number of new DOF, so that the user can create the new vector.
- In the third step, the values belonging to unchanged grid objects are copied to the new vector and the temporary values from step one are inserted.

In our case, we have DOFmanagers for the triangles, the edges and the vertices.

In case of the PCE discretization method, we have three vectors to be transferred on the triangles: ϕ_h and the gradients in x - and y -direction. We need the gradients to compute the refined value of ϕ , see Section 4.9.2. The DOFmanagers for the edges and vertices do not transfer any data.

For any of the other discretization methods, we transfer ϕ_h with the DOFmanager on the triangles as before, but now the gradient lives on the edges, so that the DOFmanager for the edges transfers the gradient. As before, we have nothing to transfer on the vertices.

In the case of Neumann boundary conditions, we need to keep track of the boundary edges, since sometimes we need a DOF on boundary edges and sometimes not. Therefore, an additional flag was introduced.

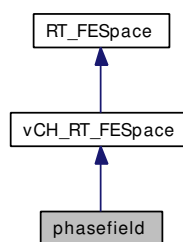
7.4 CHEST

CHEST is the main component of our software. It accesses all other components. A proper overview of the software is beyond the scope of this work, so we will restrict ourselves to giving some hints on the structure of the software and provide some typical examples.

The definitions for the potential G and the mobility G for the various equations, i.e. shallow quench, deep quench, variable quench, epitaxial growth and thin-film equation, are in separate files and for each equation, a separate executable is built.

7.4.1 Structure

The core of the software are the three classes depicted on the right. RT_FESpace contains everything that depends only on the Raviart–Thomas finite element space, like the the mass matrix \mathbf{B}_0 , its sparse approximate inverse, the basic operators (divergence, weak- and PCE-gradient, Laplacian), calculating the Dirichlet integral, solving Poisson’s equation, to name a few. There is no extra class for the finite-element space consisting of piecewise constant functions, since operations involving this space are simple enough to be performed where they are needed.



The software was developed for this work and is not intended to use other finite-element spaces, so the vCH_RT_FESpace class is derived from RT_FESpace and does not contain it as an object. vCH_RT_FESpace contains mainly the stiffness matrices of Section 4.7. Derived from this class is phasefield with the central member function perform_timestep. This function performs the sub-steps and Newton steps according to the chosen time-step scheme, computes the distance of the vertices after each step, triggers mesh adaption and everything else that has to be done to perform one time step.

As a typical example for what happens in many of the member functions, we show the code for assembling the stiffness matrix A_1 in Listing 7.1. The `ElementIterator` of the mesh is used to iterate over all active triangles. As it is faster to write several values into a matrix at once, we first collect the global edge numbers (lines 14–22) and values (lines 24–35) for the nine possible pairings of two edges of the triangle and then add them to the matrix (line 35). For a clearer presentation, we removed the parts exploiting the symmetry.

Listing 7.1: The function to assemble the stiffness matrix A_1

```

1  void vCH_RT_FESpace::assemble_A1() {
2      base_matrix* stiffness_matrix = ganzA1;
3      stiffness_matrix->clear();
4      // iterate over all triangles/elements
5      for (mesh::ElementIterator Tit=gitter->getFirstElement(); Tit!=gitter->ElementEnd(); ++Tit
6          ) {
7          Triangle* T = *Tit;
8          double Tvol = T->getVolume();
9          double phiT = phi->getval(T->getdofnr());
10         double mult = Gss( getphi(T) );
11         // iterate over all pairings of edges
12         int global_nr[3];
13         double values[9];
14         // record global edge numbers
15         for (int lokal_nr=0; lokal_nr<3; ++lokal_nr) {
16             Edge* E = T->getEdge(lokal_nr);
17             int edge_nr = E->getdofnr();
18             if (E->isBd()) {
19                 global_nr[lokal_nr] = -1;
20             } else {
21                 global_nr[lokal_nr] = edge_nr;
22             }
23         }
24         // calculate values for (A_1)_{ij}
25         for (int lokal_i=0; lokal_i<3; ++lokal_i) {
26             Edge* Ei = T->getEdge(lokal_i);
27             if (Ei->isBd()) continue;
28             double Eilen = Ei->length();
29             short muiT = T->getmu(lokal_i);
30             for (int lokal_j=0; lokal_j<3; ++lokal_j) {
31                 Edge* Ej = T->getEdge(lokal_j);
32                 if (Ej->isBd()) continue;
33                 double Ejlen = Ej->length();
34                 short mujT = T->getmu(lokal_j);
35                 double valij = mult * muiT * mujT * Eilen * Ejlen / Tvol;
36                 values[lokal_i*3+lokal_j] = valij;
37             }
38         }
39         stiffness_matrix->addtoentries( 3, global_nr, 3, global_nr, values );
40     }
41     stiffness_matrix->assemblyCompleted();
42 }

```

7.4.2 Resuming

Some of the simulations run for some weeks, so that the ability to continue a simulation that has been aborted by some reason is very helpful. Therefore, all relevant classes have a `serialize` member function, which writes the data to a file. The files are text files in XML style and all

double numbers are written as base64 encoded binary data. This resume data is written, say, every four hours and at the end of the simulation.

Upon restart, the user parameters are read again, so that the parameters can also be changed if necessary. There is also a menu which can be activated through sending the SIGINT signal and which provides direct access to some common parameters and allows tasks as writing all matrices and the like.

7.4.3 Data output

In intervals specified in the parameter file (see below), the current value of ϕ_h is written to disk. For each triangle T , the position of its vertices and the value of ϕ_h on T is written as binary data. This data can then be read in MATLAB and visualized with the patch function.

Since ϕ_h is piecewise constant, the resulting graphics are difficult to read. Therefore, when writing the data, we calculate for each vertex the mean of ϕ over all adjacent triangles. Using this data for visualization gives a much better impression of how ϕ_h looks like.

7.4.4 Parameter file

To set the various parameters for a simulation, the user writes a text file which is parsed at program start. An example for such a file is shown in Listing 7.2. The first line is an identifier which includes the version of the parameter file.

Listing 7.2: Parameter file for the simulation of the coarsening process.

```

1  cahnpamE06
2  ##### Parameter of the equation #####
3
4  ## Inverse of the temperature / width of transition layer
5  eps 1
6  beta 4
7  stepzpos 0
8
9  ## Deposition
10 F 0
11
12 ## only EG:
13 ## zetaminus=0 => constant mobility => no barrier
14 ## zetaminus=oo => Neumann BC at a step down
15 zetaminus 0
16
17 ## Initial values
18 initial_data white_noise 0 0.01
19 initial_data_use_midpoint 1
20 initial_data_min_energy 0
21
22 ##### Parameter of the discretization #####
23
24 ## time-step scheme (eulerbw, cn, theta, trbdf2)
25 time_discretisation trbdf2
26
27 ## gradient method (spasi, pcl, abt, sine)
28 gradient_method abt

```



```
29
30 ## final time
31 zeit 350000
32 ## enter menu at the end?
33 wait_at_end 0
34 ## wait until w very small before starting?
35 let_w_relax 0
36 ## stop if w very small?
37 stop_if_w_smaller_than 0
38
39 ## Initial time-step size
40 DT 4e-6
41 use_adaptive_timestep 1
42 max_timestep 4000
43 min_timestep 1e-8
44 ats_maxchange 0.26
45 ats_minchange -1
46 error_estimator trbdf2
47
48 ## Output options
49 use_exponential_writetimes 1
50 alt_exp_writetimes_tstart 0.15
51 alt_exp_writetimes_tend 350000
52 writeinterval 350
53 write_steppos 128
54 calc_wdiff 0
55 write_data 5
56 write_contvec_valjump 0
57 resume_mins 240
58 force_exact_writetimes 0
59
60 ## Mesh
61 meshdir /home/penzler/07 data/meshes
62 meshname mesh_cross_N78x78_L73.4x73.4_P0.pergrid
63
64 ## Refine initial mesh?
65 globally_refine_initial_mesh 0
66 refine_initial_mesh_max_prefactor 0
67 refine_initial_mesh 1
68
69 ## Adaptive mesh un-/refinement?
70 adapt_mesh 1
71 max_refinements 10
72 max_coarsenings 10
73 flatzone_width 0.8
74 hmin_prefactor 0.15
75 coarsening_factor 9999
76 coarsening_min_fraction 0.02
77 refine_min_energy 0
78
79 ## calculate  $H^{-1}$ -norm?
80 calc_negnorm 1
81
82 ## Use Cholesky up to #DOF?
83 cholesky_max_edges 600000
84
85 ## Extract submatrix (->DQ) ?
86 submatrix_beve 0
87 submatrix_borderfrac 0.999
88
89 ## Number of Newton steps
90 newton_steps 3
91
92 ## Parameter p in EG
93 aniso 20
```


Appendix A

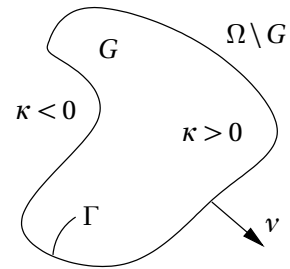
Complements

In this appendix, we provide complementary information to several topics which did not fit into the main text.

A.1 Gradient-flow structure of the sharp-interface limits

We show here the gradient-flow structure of the limit-models of the deep- and shallow-quench equations, namely motion by surface diffusion and the Mullins–Sekerka free boundary problem. For surface diffusion, this was first shown by Taylor and Cahn [1994]. The derivation for Mullins–Sekerka can be found in Niethammer and Otto [2001].

A number of properties is common to both models. The manifold \mathcal{M} consists of all subsets of Ω with given total area and the tangent space consists of all possible movements of the boundary that keep the mass constant:



$$\mathcal{M} = \left\{ G \subset \Omega \mid \int_G 1 = \text{const} \right\}, \quad T_G \mathcal{M} = \left\{ V : \Gamma \rightarrow \mathbb{R} \mid \int_\Gamma V = 0 \right\},$$

where we denoted the boundary of G with Γ . Moreover, the energy functional is the length of the interface

$$E(G) = \int_\Gamma 1.$$

The gradient flow is given by

$$g(V, \tilde{V}) + \text{diff } E(G) \tilde{V} = 0 \quad \forall \tilde{V} \in T_G \mathcal{M}.$$

It is well-known that the first variation of the interface length is the curvature of the interface, so the gradient flow becomes

$$g(V, \tilde{V}) + \int_\Gamma \kappa \tilde{V} = 0 \quad \forall \tilde{V} \in T_G \mathcal{M}. \quad (\text{A.1})$$

The models differ by the choice of the metric. For surface diffusion, the metric is the H^{-1} -norm of the velocities

$$g_G^{\text{SD}}(V, \tilde{V}) = \langle V, \tilde{V} \rangle_{H^{-1}(\Gamma)} = \int_{\Gamma} \nabla_s w \cdot \nabla_s \tilde{w},$$

where w and \tilde{w} are solutions of $-\Delta_s w = V$ and $-\Delta_s \tilde{w} = \tilde{V}$, respectively. Partial integration yields

$$g_G^{\text{SD}}(V, \tilde{V}) = \int_{\Gamma} w(-\Delta_s \tilde{w}) = \int_{\Gamma} w \tilde{V},$$

so that we get with (A.1)

$$w = -\kappa \quad \implies \quad V = -\Delta_s w = \Delta_s \kappa. \quad (\text{A.2})$$

For Mullins–Sekerka the metric is similar to the H^{-1} -norm, but on the whole of Ω and not on the interface.

$$g_G^{\text{MS}}(V, \tilde{V}) = \int_{\Omega} \nabla w \cdot \nabla \tilde{w},$$

where w is a solution of

$$-\Delta w = 0 \quad \text{in } \Omega \setminus \Gamma, \quad (\text{A.3a})$$

$$\left(\nabla w \Big|_{\Omega \setminus G} - \nabla w \Big|_G \right) \cdot \nu = V \quad \text{on } \Gamma, \quad (\text{A.3b})$$

and analogously for \tilde{w} . Here ν is the outer normal to G . Since \tilde{w} is harmonic, integration by parts only yields boundary terms

$$g_G^{\text{MS}}(V, \tilde{V}) = \int_{\Gamma} w \left(\nabla \tilde{w} \Big|_G - \nabla \tilde{w} \Big|_{\Omega \setminus G} \right) \cdot \nu = - \int_{\Gamma} w \tilde{V}.$$

From (A.1), we then get

$$w = \kappa, \quad (\text{A.3c})$$

which completes the Mullins–Sekerka free boundary problem.

A.2 Energy minimization during refinement

During the refinement of the mesh, when a triangle is bisected, the distribution of ϕ onto the two newly created triangles is not unique, see Section 4.9.2. When the refinement is finished, we optimize the distribution by minimizing the energy.

It is convenient to take a purely discrete approach: Consider the function

$$E : \mathbb{R}^{N_T} \rightarrow \mathbb{R}, \quad E(\underline{\phi}) = \frac{\varepsilon}{2} \mathbf{B}_0^{-1} \mathbf{G} \underline{\phi} \cdot \mathbf{G} \underline{\phi} + \varepsilon^{-1} \mathbf{V} \mathbf{G}(\underline{\phi}). \quad (\text{A.4a})$$

The function E was obtained by writing the discrete energy (4.51) in matrix-form.

We minimize this energy with the constraint that the mass on each refined triangle is conserved, which yields the linear constraint

$$\mathbf{P}\underline{\phi} = \underline{m}, \quad (\text{A.4b})$$

see Section A.2.1 below. In Section A.2.2 we solve the constrained problem using the augmented Lagrangean method.

Remark A.1. In Section 4.9.3 we mentioned that minimizing the energy means to find a critical point of the functional (4.52). This is the same as minimizing the discrete functional E . Indeed, the optimality conditions of (4.52) are

$$\begin{aligned} \varepsilon \int_{\Omega} \mathbf{g}_h \cdot \tilde{\mathbf{g}}_h + \int_{\Omega} (\nabla \cdot \tilde{\mathbf{g}}_h) \phi_h &= 0 \quad \forall \tilde{\mathbf{g}}_h \in \mathcal{RT}_0(\mathcal{E}_h), \\ \int_{\Omega} (\nabla \cdot \mathbf{g}_h) \zeta_h - \varepsilon^{-1} \int_{\Omega} G'(\phi_h) \zeta_h &= 0 \quad \forall \zeta_h \in \mathcal{L}_0(\mathcal{T}_h), \end{aligned}$$

which is in matrices

$$\varepsilon \mathbf{B}_0 \underline{\mathbf{g}} - \mathbf{G} \underline{\phi} = 0 \quad \text{and} \quad \mathbf{D} \underline{\mathbf{g}} - \varepsilon^{-1} \mathbf{V} G'(\underline{\phi}) = 0.$$

Solving the first equation for $\underline{\mathbf{g}}$ and inserting it into the second yields the condition

$$-\varepsilon \mathbf{D} \mathbf{B}_0^{-1} \mathbf{G} \underline{\phi} + \varepsilon^{-1} \mathbf{V} G'(\underline{\phi}) = 0$$

which is identical to the optimality condition of (A.4a). \diamond

A.2.1 Setting up the constraint

It would be very convenient if we could determine the new values of ϕ_h on the two new, refined triangles immediately during refinement. This is not possible, since the energy is nonlocal due to the inverse of the mass matrix. Even if the inverse would use only a few neighbours this wouldn't be possible, since there are usually hanging nodes created and so the matrices cannot be assembled. Therefore, during refinement, we keep track of the refinements taking place and build the constraint after the refinement.

The constraint says that mass is to be kept constant and is not moved between triangles. As an example for the constraint, consider the following one-dimensional example of a mesh refinement

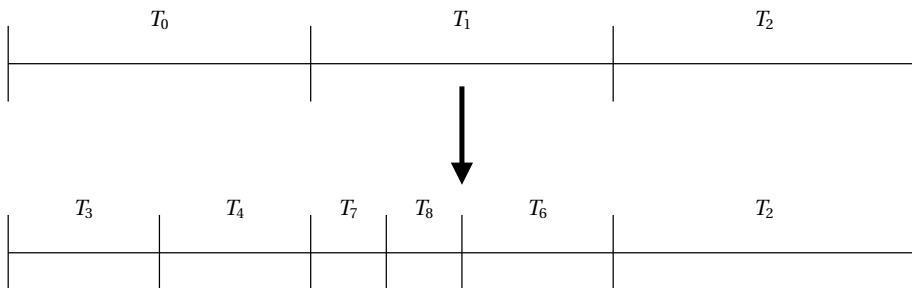
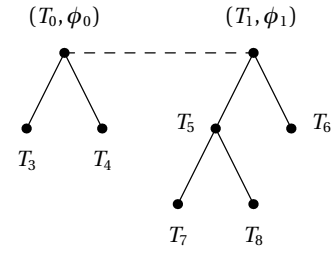


Figure A.1: During refinement the refined triangles, along with the old value of ϕ on that triangle, are written in a list, each building the root of a tree structure. The newly created children are added to the tree. Every tree in the list corresponds to one row in the matrix \mathbf{P} . To setup the matrix, we iterate the leaves of each tree and write 2^{-d} in the column corresponding to the triangle in that leaf, where d is the depth of the leaf.



The constraints are then

$$\begin{aligned} |T_3| \underline{\phi}_3 + |T_4| \underline{\phi}_4 &= |T_0| \underline{\phi}_0 \\ |T_7| \underline{\phi}_7 + |T_8| \underline{\phi}_8 + |T_6| \underline{\phi}_6 &= |T_1| \underline{\phi}_1. \end{aligned}$$

Since we always (also in the two-dimensional case) refine by bisection, we can divide by $|T_0|$ and $|T_1|$, respectively, and get as constraints

$$\begin{aligned} \frac{1}{2} \underline{\phi}_3 + \frac{1}{2} \underline{\phi}_4 &= \underline{\phi}_0 \\ \frac{1}{4} \underline{\phi}_7 + \frac{1}{4} \underline{\phi}_8 + \frac{1}{2} \underline{\phi}_6 &= \underline{\phi}_1. \end{aligned}$$

The technical part of setting up \mathbf{P} is described in Figure A.1.

A.2.2 Solving the constrained minimization problem

To solve the minimization problem (A.4), we use the *augmented Lagrangean* method. Also under consideration were

- Lagrange–Newton method. Since we want the constraint to be fulfilled exactly (or at least to very high accuracy), this method is not suitable.
- Penalty methods are not suitable for the same reason.
- Reduction or Projection methods need a basis of the kernel or image of the matrix \mathbf{P} , which is not available at low cost.

The augmented Lagrangean method will be described only roughly in the following, see for example Geiger and Kanzow [2002] for details.

Define the augmented Lagrangean function by

$$L_{\text{aug}}(\underline{x}, \underline{\mu}, \alpha) := E(\underline{x}) + \frac{\alpha}{2} \|\mathbf{P}\underline{x} - \underline{m}\|^2 + \underline{\mu} \cdot (\mathbf{P}\underline{x} - \underline{m}).$$

In each step of the algorithm, the unconstrained minimum $\underline{x}^{(k+1)}$ of the problem

$$\min_{\underline{x} \in \mathbb{R}^{N_T}} L_{\text{aug}}(\underline{x}, \underline{\mu}^{(k)}, \alpha^{(k)}) \quad (\text{A.5})$$

is computed, see the next section. At the minimum, we have

$$0 = \nabla_x L_{\text{aug}}(\underline{x}^{(x+1)}, \underline{\mu}^{(k)}, \alpha^{(k)}) = \nabla E(\underline{x}^{(k+1)}) + \mathbf{P}^t \left(\alpha^{(k)}(\mathbf{P}\underline{x}^{(k+1)} - \underline{m}) + \underline{\mu}^{(k)} \right).$$

On the other hand, at a minimum $(\underline{x}^*, \underline{\mu}^*)$ of (A.4), we have

$$\nabla E(\underline{x}^*) + \mathbf{P}^t \underline{\mu}^* = 0.$$

Comparing these two suggests to set

$$\underline{\mu}^{(k+1)} := \alpha^{(k)}(\mathbf{P}\underline{x}^{(k+1)} - \underline{m}).$$

The parameter α is increased if the value of $\|\mathbf{P}\underline{x} - \underline{m}\|$ doesn't decrease fast enough. One feature of the augmented Lagrangean method is that the constraint already holds exactly for finite α .

A.2.3 Solving the unconstrained minimization problem

For the minimization problem (A.5), we use a globalized Newton method. The starting value of $\underline{\phi}^{(0)}$ is created in the refinement process, see Section 4.9.2.

In every Newton step, we solve the system

$$D_x^2 L_{\text{aug}}(\underline{x}, \underline{\mu}, \alpha) \underline{d}^{(k+1)} = -\nabla_x L_{\text{aug}}(\underline{x}, \underline{\mu}, \alpha)$$

for the Newton direction $\underline{d}^{(k+1)}$. In matrices, this is

$$\left(-\varepsilon \mathbf{D} \mathbf{B}_0^{-1} \mathbf{G} + \varepsilon^{-1} \mathbf{N} + \alpha \mathbf{P}^t \mathbf{P} \right) \underline{d}^{(k+1)} = - \left(-\varepsilon \mathbf{D} \mathbf{B}_0^{-1} \mathbf{G} \underline{x}^{(k)} + \varepsilon^{-1} \mathbf{V} \mathbf{G}(\underline{x}^{(k)}) \right),$$

where \mathbf{N} is the nonlinear diagonal matrix

$$\mathbf{N}_{kk} = |T_k| G(\underline{x}^{(k)}).$$

Since there appears \mathbf{B}_0^{-1} in the matrix expression and since the matrix $\mathbf{P}^t \mathbf{P}$ can be rather dense, we use a matrix-free conjugate gradient method to solve the system. It turned out that it is important to adapt the tolerance of the solver. We found that setting

$$\tilde{tol} := \frac{10^{-5}}{k+1} \|\nabla_x L_{\text{aug}}\|^2, \quad tol = \min \left(10^{-5}, \max \left(\tilde{tol}, \frac{10^{-20}}{k+1} \right) \right)$$

works quite well.

If the computed direction is good enough, determined by

$$\nabla_x L_{\text{aug}}(\underline{x}, \underline{\mu}, \alpha) \cdot \underline{d}^{(k+1)} < -10^{-8} \|\underline{d}^{(k+1)}\|^{2.1},$$

we take this direction to calculate $\underline{x}^{(k+1)}$. If not, we perform a gradient step by setting

$$\underline{d}^{(k+1)} := \nabla L_{\text{aug}}(\underline{x}, \underline{\mu}, \alpha).$$

The step size t used in

$$\underline{x}^{(k+1)} = \underline{x}^{(k)} + t \underline{d}^{(k+1)}$$

is computed using the well-known Amijo rule: find the smallest $l \in \mathbb{N}_0$ such that with $t = 2^{-l}$

$$L_{\text{aug}}(\underline{x}^{(k)} + t \underline{d}^{(k+1)}, \underline{\mu}, \alpha) \leq L_{\text{aug}}(\underline{x}^{(k)}, \underline{\mu}, \alpha) + \sigma t \nabla L_{\text{aug}}(\underline{x}^{(k)}, \underline{\mu}, \alpha) \cdot \underline{d}^{(k+1)},$$

where we use $\sigma = 10^{-4}$.

A.3 Determining the step position

To compare the results of the diffuse-interface models, i.e. the Cahn–Hilliard-type equations, with their sharp-interface limits, we have to extract the position of the interface from the order parameter ϕ . In the case of epitaxial growth, the interface- or step position is given by the level set

$$\{\phi = \frac{1}{2}\}.$$

Our goal is therefore to find points $S \in \Omega$ with $\phi(S) = 1/2$. Since ϕ_h is piecewise constant, we won't have $\phi_h(S) = 1/2$. The strategy is therefore to locally approximate ϕ_h with a continuous function.

As a first step, we search those edges which have $\phi_h \geq 1/2$ on one neighbouring triangle and $\phi_h < 1/2$ on the other. These “step-edges” will be marked red in subsequent figures. Then we approximate ϕ_h by a continuous function. As we know that the transition profile at the step is close to the function

$$\frac{1}{2} \tanh\left(\frac{3}{\varepsilon}(x \cdot \nu - \xi)\right) + \frac{1}{2} \approx \frac{3}{2\varepsilon}(x \cdot \nu - \xi) + \frac{1}{2} \text{ for } x \cdot \nu - \xi \ll \varepsilon,$$

it is natural to use an affine function for the approximation. We present in the following three different methods to choose this affine function.

A.3.1 Least squares (LSQ) approximation

To define the affine function

$$L_1(x) = \nu \cdot x + \beta_1$$

with $\nu \in \mathbb{R}^2$ and $\beta_1 \in \mathbb{R}$, we require that the restriction of L_1 to $\mathcal{L}_0(\mathcal{T}_h)$ is equal to ϕ_h on T_E^\pm , that is

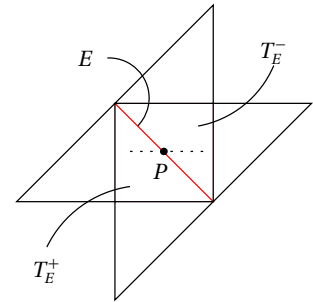
$$\int_{T_E^+} L_1 = \int_{T_E^+} \phi_h \text{ and } \int_{T_E^-} L_1 = \int_{T_E^-} \phi_h$$

Since this gives only two conditions, we take all the neighbours of T_E^\pm into account and use least squares in this approach:

$$\sum_{i=1\dots 6} \left(\int_{T_i} L_1 - \phi_h \right)^2 = \min! \quad (\text{A.6})$$

Inserting the ansatz function L_1 into (A.6) yields

$$\frac{1}{2} \mathbf{A} \underline{z} \cdot \underline{z} - \underline{b} \cdot \underline{z} = \min!$$



with

$$\mathbf{A} = \begin{pmatrix} \sum_i |T_i|^2 M_{i,1}^2 & \sum_i |T_i|^2 M_{i,1} M_{i,2} & \sum_i |T_i|^2 M_{i,1} \\ \sum_i |T_i|^2 M_{i,1} M_{i,2} & \sum_i |T_i|^2 M_{i,2}^2 & \sum_i |T_i|^2 M_{i,2} \\ \sum_i |T_i| M_{i,1} & \sum_i |T_i| M_{i,2} & \sum_i |T_i|^2 \end{pmatrix},$$

$$\underline{b} = \begin{pmatrix} \sum_i |T_i|^2 \phi_i M_{i,1} \\ \sum_i |T_i|^2 \phi_i M_{i,2} \\ \sum_i |T_i|^2 \phi_i \end{pmatrix} \quad \text{and} \quad \underline{z} = \begin{pmatrix} v_1 \\ v_2 \\ \beta_1 \end{pmatrix},$$

where M_i are the centres of the triangles. We therefore have to solve the 3×3 system

$$\mathbf{A}\underline{z} = \underline{b}$$

to get v and β_1 . This is done using Gauß' scheme. Then we search λ_1 such that

$$L_1(P + \lambda_1 v) = \frac{1}{2}.$$

A disadvantage of this approach is that we do not use any other quantities than ϕ_h to get information on the shape of ϕ .

A.3.2 Gradient approximation

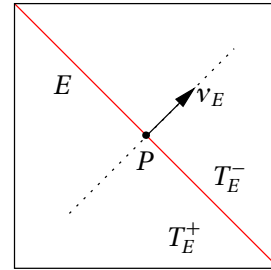
A quantity that can give us more information on the shape of ϕ is of course the discrete gradient $\nabla_h \phi_h$. As the discrete gradient is a Thomas–Raviart vector field, the normal component of $\nabla_h \phi_h$ coincides with the continuous gradient on the edges. Therefore, we choose the affine function

$$L_2(x) = \alpha v_E \cdot x + \beta_2$$

with $\alpha, \beta_2 \in \mathbb{R}$ and the conditions

$$(i) \quad \nabla L_2(P) \cdot v_E = \nabla_h \phi_h(P) \cdot v_E,$$

$$(ii) \quad \int_{T_E^+ \cup T_E^-} L_2 = \int_{T_E^+ \cup T_E^-} \phi_h.$$



This gives $\alpha = \nabla_h \phi_h \cdot v_E = \underline{g}_E$ and

$$\beta_2 = \frac{|T_E^+|(\phi^+ - \alpha v_E \cdot M^+) + |T_E^-|(\phi^- - \alpha v_E \cdot M^-)}{|T_E^+| + |T_E^-|},$$

where M^\pm are, again, the centres of the triangles.

Then we search λ_2 such that

$$L_2(P + \lambda_2 v_E) = \frac{1}{2},$$

i.e. we are searching along the dotted line in the sketch above. This seems to be a bit odd since the step front will in general not be parallel to the edge E . Therefore, we consider a third method which uses both the direction of the step front v and the weak gradient.

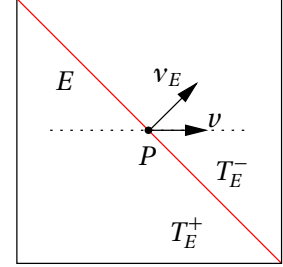
A.3.3 Mixed approximation.

Here, we use an Ansatz similar to the previous one, but use v from the LSQ approximation instead of v_E :

$$L_3(x) = \alpha v \cdot x + \beta_3$$

with $\alpha, \beta_3 \in \mathbb{R}$ and the conditions

- (i) $\nabla L_3(P) \cdot v_E = \nabla_h \phi_h(P) \cdot v_E$,
- (ii) $\int_{T_E^+ \cup T_E^-} L_3 = \int_{T_E^+ \cup T_E^-} \phi_h$.



This yields

$$\alpha = \frac{\nabla_h \phi_h \cdot v_E}{v \cdot v_E} = \frac{g_E}{v \cdot v_E},$$

$$\beta_3 = \frac{|T_E^+|(\phi^+ - \alpha v \cdot M^+) + |T_E^-|(\phi^- - \alpha v \cdot M^-)}{|T_E^+| + |T_E^-|},$$

and we search λ_3 such that

$$L_3(P + \lambda_3 v) = \frac{1}{2}.$$

A.3.4 Comparison of the approximations

For a first test, we consider a straight step growing from left to right on a domain $\Omega = [0, 1] \times [0, 1/2]$. Since we want to test here the step-finding algorithm and not the simulation, in each time step we set

$$\phi_h = R_h(f(t)) \quad \text{with} \quad f(t) = \frac{1}{2} \left(1 - \tanh \left(\frac{3}{\epsilon} (x_1 - \xi(t)) \right) \right)$$

with the restriction operator (4.6) and step position $\xi(t)$. Although this function is only correct for an infinite domain, the solution for a finite domain is very close.

We set the deposition rate to one, so the speed of the step is $f L_x = 1$ and therefore $\xi(t) = \xi_0 + t$, where we choose $\xi_0 = 0.25$. We then measure the step position for each step-edge using the above approximations. Since the variations of the position for the different step-edges is virtually zero (of the order 10^{-14} , which is the accuracy of a double precision variable), we have a unique position $\xi_h(t)$ for each time step. We then compute

$$d\xi(t) := \xi_h(t) - \xi(t) = \xi_h(t) - t - 0.25. \quad (\text{A.7})$$

The results can be seen in Figure A.2. For comparison, we have added a fourth approximation “prescribedstep”, which is the mixed approximation with $v = (1, 0)$.

All the approximations are satisfactory, while “gradstep” and “mixedstep” are a bit better than “lsqstep”. Further testing showed that “mixedstep” is better suited for curved step fronts.

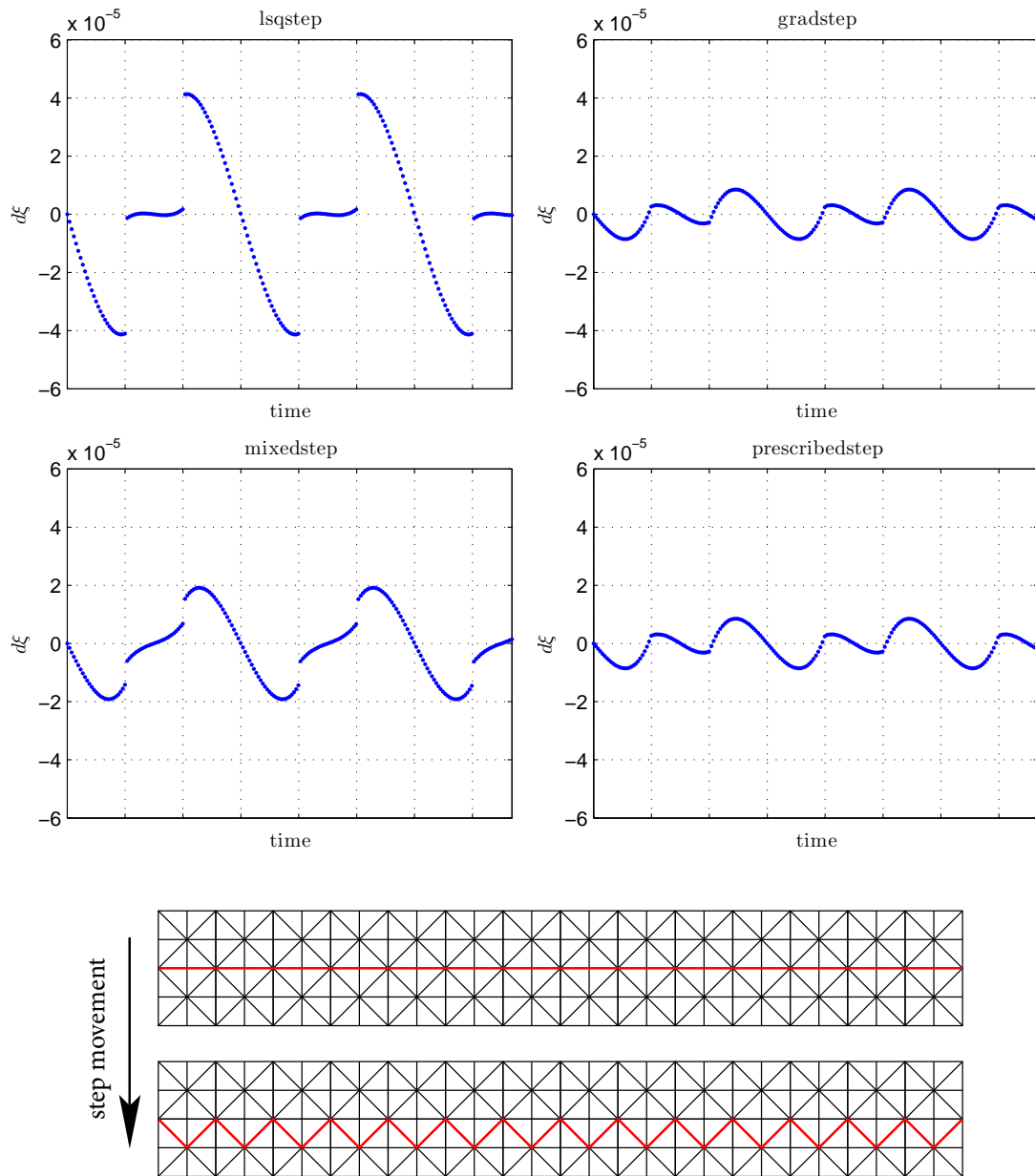


Figure A.2: Error in measured step position for least squares (top-left), gradient (top-right), mixed (lower-left) and prescribed (lower-right) approximations. Note that the scale is 10^{-5} . The jumps occur when the edges between triangles with $\phi_h \geq 1/2$ and $\phi_h < 1/2$ change from a straight line to a zick-zack pattern, see the bottom images.

A.3.5 Approximation with piecewise constant gradient

In the case of piecewise constant gradients (see Section 4.8.5), we know not only the normal component of the gradient, but the whole gradient on each triangle. Therefore, we choose

$$L_4(x) = a \cdot x + \beta_4$$

with $a \in \mathbb{R}^2, \beta_4 \in \mathbb{R}$ and the conditions

$$(i) \quad \nabla L_4(P) = \frac{1}{2} (\nabla_h \phi_h|_{T_E^+} + \nabla_h \phi_h|_{T_E^-}),$$

$$(ii) \quad \int_{T_E^+ \cup T_E^-} L_4 = \int_{T_E^+ \cup T_E^-} \phi_h.$$

As before, we then search $\lambda_4 \in \mathbb{R}$ such that

$$L(P + \lambda_4 v) = \frac{1}{2},$$

where v comes from the least-squares approximation.

A.4 Time adaptivity

In this section, we first show how to derive an error estimator from the TR-BDF2 scheme. For the definition of the scheme, see Section B.5. Then, we apply the error estimator to our equation in Section A.4.2 and show how we use the error estimator for time adaptivity. Finally, we propose in Section A.4.3 a quite naïve way to measure the error for the other time-step schemes.

A.4.1 Derivation of the error estimator

To derive the error estimator we note that TR-BDF2 can be viewed as a diagonally implicit Runge–Kutta (DIRK) scheme with an embedded third-order scheme [Hosea and Shampine, 1996]. The Butcher array is

$$\begin{array}{c|ccc} 0 & 0 & & \\ \theta & a & a & \\ 1 & \beta & \beta & a \\ \hline & \beta & \beta & a \\ \hline & \hat{b}_1 & \hat{b}_2 & \hat{b}_3 \end{array}$$

with

$$\begin{aligned} \theta &= 2 - \sqrt{2}, & a &= 1 - \frac{1}{\sqrt{2}}, & \beta &= \frac{\sqrt{2}}{4}, \\ \hat{b}_1 &= \frac{4 - \sqrt{2}}{12}, & \hat{b}_2 &= \frac{3\sqrt{2} + 4}{12}, & \hat{b}_3 &= \frac{2 - \sqrt{2}}{6}. \end{aligned}$$

This is to be read as follows: In the left column are the points in time where the evaluation takes place. The matrix indicates how the auxiliary results $z_n, z_{n+\theta}, z_{n+1}$ are computed, namely

$$\begin{aligned} z_n &= F(t_n, \phi^n), \\ z_{n+\theta} &= F(t_n + \theta\tau, \phi^n + \tau(az_n + az_{n+\theta})), \\ z_{n+1} &= F(t_n + \tau, \phi^n + \tau(\beta z_n + \beta z_{n+\theta} + az_{n+1})). \end{aligned}$$

Using the notion

$$\phi^* = \phi^n + \tau(az_n + az_{n+\theta}) \quad \text{and} \quad \phi^{n+1} = \phi^n + \tau(\beta z_n + \beta z_{n+\theta} + az_{n+1}) \quad (\text{A.8})$$

and dropping the time argument in F , we get $z_n = F(\phi^n)$, $z_{n+\theta} = F(\phi^*)$ and $z_{n+1} = F(\phi^{n+1})$. Then, (A.8) reads as

$$\begin{aligned} \frac{1}{a\tau}(\phi^* - \phi^n) &= F(\phi^n) + F(\phi^*), \\ \frac{1}{a\tau}(\phi^{n+1} - \phi^* + \phi^* - \phi^n) &= \frac{\beta}{a}(F(\phi^n) + F(\phi^*)) + F(\phi^{n+1}), \end{aligned}$$

which yields, after inserting the first equation into the second, Equations (B.3):

$$\begin{aligned} \frac{1}{a\tau}(\phi^* - \phi^n) &= F(\phi^n) + F(\phi^*), \\ \frac{1}{a\tau}(\phi^{n+1} - \phi^*) &= F(\phi^{n+1}) + \left(\frac{\beta}{a} - 1\right)(F(\phi^*) + F(\phi^n)) \end{aligned}$$

with $(\beta/a) - 1 = (\sqrt{2} - 1)/2$.

Let \mathfrak{D} and $\hat{\mathfrak{D}}$ denote the discrete evolution given by the second-order TR-BDF2 and the third-order embedded scheme, respectively. The two lowermost columns of the Butcher array define how to evaluate these discrete evolutions:

$$\begin{aligned} \mathfrak{D}^\tau \phi^n &= \phi^n + \tau(\beta z_n + \beta z_{n+\theta} + az_{n+1}), \\ \hat{\mathfrak{D}}^\tau \phi^n &= \phi^n + \tau(\hat{b}_1 z_n + \hat{b}_1 z_{n+\theta} + \hat{b}_1 z_{n+1}). \end{aligned}$$

Furthermore, let \mathfrak{C} be the continuous (exact) evolution defined through $\mathfrak{C}^\tau \phi(t, x) = \phi(t + \tau, x)$. Then the errors of the discrete evolutions are

$$\eta = \mathfrak{C}^\tau \phi - \mathfrak{D}^\tau \phi \quad \text{and} \quad \hat{\eta} = \mathfrak{C}^\tau \phi - \hat{\mathfrak{D}}^\tau \phi.$$

Since the third-order evolution is “better”, we have

$$\vartheta := \frac{\|\hat{\eta}\|}{\|\eta\|} < 1.$$

We define the error estimator $[\eta]$ by

$$[\eta] := \hat{\mathfrak{D}}^\tau \phi - \mathfrak{D}^\tau \phi = \hat{\mathfrak{D}}^\tau \phi - \mathfrak{C}^\tau \phi + \mathfrak{C}^\tau \phi - \mathfrak{D}^\tau \phi = \eta - \hat{\eta}.$$

Inserting the discrete evolutions, we get

$$[\eta] = \frac{\tau}{3} \left((1 - \sqrt{2})F(\phi^n) + F(\phi^*) + (\sqrt{2} - 2)F(\phi^{n+1}) \right).$$

Note that if ϕ is stationary, i.e. $F(\phi^n) = F(\phi^*) = F(\phi^{n+1})$, then $[\eta] = 0$.

To see that this is an effective and robust error estimator, we calculate

$$\begin{aligned} \|\eta\| - \|[\eta]\| &\leq \|\eta - [\eta]\| = \|\hat{\eta}\| = \vartheta\|\eta\| \\ &\implies (1 - \vartheta)\|\eta\| \leq \|[\eta]\|. \end{aligned}$$

and analogously

$$\begin{aligned} \|[\eta]\| - \|\eta\| &\leq \|[\eta] - \eta\| = \|\hat{\eta}\| = \vartheta\|\eta\| \\ &\implies (1 + \vartheta)\|\eta\| \geq \|[\eta]\|. \end{aligned}$$

This gives together

$$\frac{1}{1 + \vartheta}\|[\eta]\| \leq \|\eta\| \leq \frac{1}{1 - \vartheta}\|[\eta]\|.$$

To adjust the time-step size, consider the following. Given a tolerance tol , a time-step size τ_* would be considered optimal if it yields an error exhausting the given tolerance:

$$\|\eta_*\| \approx tol.$$

Since we have a second-order scheme, we know

$$\|\eta_*\| = \mathcal{O}(\tau_*^3) \quad \text{and} \quad \|\eta\| = \mathcal{O}(\tau^3).$$

Additionally, the error estimator approximates the real error,

$$\|[\eta]\| \approx \|\eta\|,$$

so we get

$$\frac{tol}{\|[\eta]\|} \approx \frac{\|\eta_*\|}{\|\eta\|} \approx \left(\frac{\tau_*}{\tau} \right)^3$$

and therefore we can approximate the optimal time-step size by

$$\tau_* \approx \tau \left(\frac{tol}{\|[\eta]\|} \right)^{\frac{1}{3}}.$$

The problem is of course that we first need to perform the time step to be able to calculate the error estimator and in turn the optimal time-step size. Therefore, an explicit strategy is used:

- 1: Perform time step with step size τ
- 2: Calculate $\bar{\eta} := \|[\eta]\|$ and τ_*
- 3: Set $\tau \leftarrow \tau_*$

- 4: **if** $\bar{\eta} > \eta_{\max}$ **then**
- 5: repeat time step
- 6: **end if**
- 7: perform next time step

We added to the calculation of τ_* a “safety factor” $\rho \in [0, 1)$:

$$\tau_* := \tau \left(\rho \frac{\text{tol}}{\bar{\eta}} \right)^{\frac{1}{3}}.$$

Usually, we use $\rho = 0.8$, which is quite large. For some calculations it is necessary to reduce ρ in order to avoid too many discarded steps. The maximum error η_{\max} is set by the user.

A.4.2 Computation of error estimator

In the previous section we found that using TR-BDF2 to discretize $\partial_t \phi = F(\phi)$ in time, we get an error estimator

$$[\eta] = \tau \left(\frac{1 - \sqrt{2}}{3} F(\phi^n) + \frac{1}{3} F(\phi^*) + \frac{\sqrt{2} - 2}{3} F(\phi^{n+1}) \right).$$

Remember the numbering of the time steps in Chapter 3. From the point of view of the second sub-step of TR-BDF2, the indices translate to

$$n \mapsto k - 1, \quad * \mapsto k \quad \text{and} \quad n + 1 \mapsto k + 1.$$

Using furthermore

$$F(\phi) := \nabla \cdot (M(\phi^k) \nabla w) + f,$$

we get

$$[\eta] = \tau \nabla \cdot \hat{J},$$

where

$$\hat{J} = -M(\phi^k) \nabla \hat{w} \quad \text{and} \quad \hat{w} = \frac{\sqrt{2} - 1}{3} w^{k-1} - \frac{1}{3} w^k + \frac{2 - \sqrt{2}}{3} w^{k+1}.$$

To compute $[\eta]$ in the spatially discrete case, we first compute \hat{J}_h from

$$\int_{\Omega} \frac{1}{M(\phi_h^k)} \hat{J}_h \cdot \tilde{J}_h = \int_{\Omega} \hat{w}_h (\nabla \cdot \tilde{J}_h) \quad \forall \tilde{J}_h \in \mathcal{RT}_0(\mathcal{E}_h), \quad (\text{A.9})$$

which yields in matrices the linear system of equations

$$\mathbf{B}_1 \underline{\hat{J}} = \mathbf{G} \underline{\hat{w}}.$$

Finally, we calculate

$$\underline{[\eta]} = \tau \mathbf{D} \underline{\hat{J}} \quad \text{and} \quad \bar{\eta} := \|[\eta]\|_{L^2(\Omega)}.$$

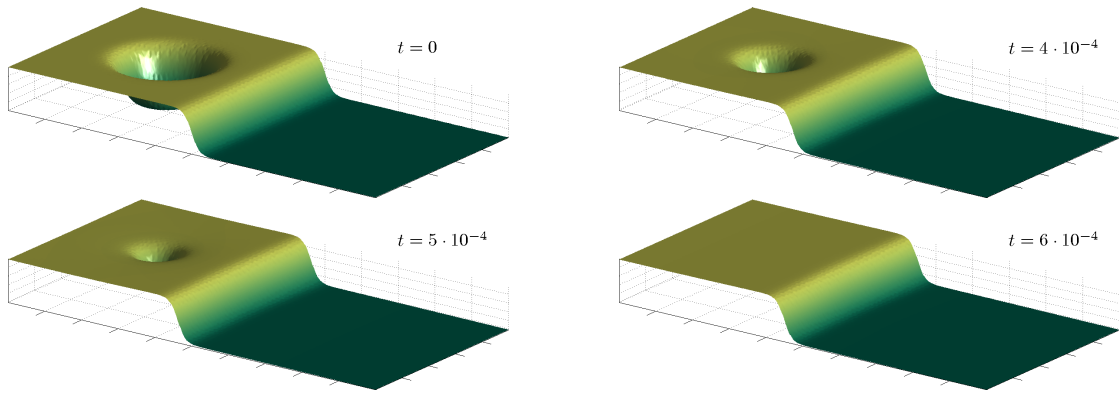


Figure A.3: Snapshots of the simulation at the beginning of the simulation and at three times around when the hole disappears. Compare to the value of the error estimator in Figure A.4 on the next page.

A.4.3 Error estimator versus change of the solution

For the other time-step schemes, we use a much simpler approach to adaptively define the time-step size. We just take

$$\|\mathcal{D}^\tau \phi - \phi\|_{L^2(\Omega)}$$

as an error indicator. This reflects the simple idea, that we can use larger time steps when not much happens.

To compare the error estimator described in the previous sections with the naïve measurement of the change of ϕ , we start a simulation with a straight step train having a hole in the upper terrace. Due to deposition, the step moves forward and the hole is filled up. When the hole gets smaller, we have higher curvature and expect a reduction of the time step size. After the hole has disappeared, we expect the time steps to become bigger again. Looking at the snapshots in Figure A.3 together with the values of the error indicators and time-step sizes in Figure A.4, we see the desired behaviour, even for the simple error indicator.

A.5 Solving Poisson's equation

To compute the H^{-1} norm, see the next section, it is necessary to solve Poisson's equation

$$-\Delta_h u_h = \phi_h, \tag{A.10}$$

with periodic or Neumann boundary conditions. This equation was also used as a benchmark for the different gradient methods in Section 4.8.

We presented two ways to define a discrete Laplacian: one was to take the divergence of the discrete gradient and the other was derived from a piecewise constant energy. The solving procedures for these two cases differ.

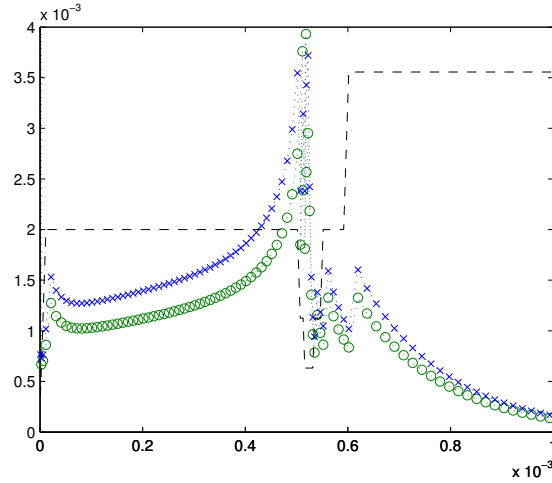


Figure A.4: Comparison of error indicators: the green circles show the values of the error estimator and the blue crosses show the change of ϕ in the L^2 -norm. The dashed line shows the resulting time-step size (on another scale).

A.5.1 ... using the discrete gradient

As in Section 4.1, we split Equation (A.10) into

$$J = \nabla u \quad \text{and} \quad -\nabla \cdot J = \phi.$$

The (standard) mixed formulation is: Find $(J, u) \in H(\nabla \cdot, \Omega) \times L^2(\Omega)$ such that

$$\begin{aligned} \int_{\Omega} J \cdot \tilde{J} + \int_{\Omega} u(\nabla \cdot \tilde{J}) &= 0 & \forall \tilde{J} \in H(\nabla \cdot, \Omega), \\ \int_{\Omega} (\nabla \cdot J)\zeta &= - \int_{\Omega} \phi \zeta & \forall \zeta \in L^2(\Omega). \end{aligned}$$

Using the finite-dimensional subspaces $\mathcal{RT}_0(\mathcal{E}_h)$ and $\mathcal{L}_0(\mathcal{T}_h)$ of $H(\nabla \cdot, \Omega)$ and $L^2(\Omega)$, respectively, we get the following discrete problem: Find $(J_h, u_h) \in \mathcal{RT}_0(\mathcal{E}_h) \times \mathcal{L}_0(\mathcal{T}_h)$ such that

$$\int_{\Omega} J_h \cdot \tilde{J}_h + \int_{\Omega} u_h(\nabla \cdot \tilde{J}_h) = 0 \quad \forall \tilde{J}_h \in \mathcal{RT}_0(\mathcal{E}_h), \quad (\text{A.11a})$$

$$\int_{\Omega} (\nabla \cdot J_h)\zeta_h = - \int_{\Omega} \phi_h \zeta_h \quad \forall \zeta_h \in \mathcal{L}_0(\mathcal{T}_h), \quad (\text{A.11b})$$

where ϕ_h is the restriction of ϕ to $\mathcal{L}_0(\mathcal{T}_h)$.

With the matrices of Section 4.7, we get

$$\begin{aligned} \mathbf{B}_0 \underline{J} - \mathbf{G} \underline{u} &= 0, \\ \mathbf{D} \underline{J} &= -\mathbf{V} \underline{\phi}. \end{aligned}$$

Since $\mathbf{D} = -\mathbf{G}^t$, the above system is symmetric, but indefinite.

To get a symmetric problem, we use the Arnold–Brezzi method, compare to Section 4.8.3. The problem (A.11) is equivalent to the problem: Find $(\hat{J}_h, u_h, \lambda_h) \in \mathcal{RT}_{-1}(\mathcal{T}_h) \times \mathcal{L}_0(\mathcal{T}_h) \times \mathcal{L}_0(\mathcal{E}_h)$ such that

$$\begin{aligned} \int_{\Omega} \hat{J}_h \cdot \tilde{J}_h + \sum_{T \in \mathcal{T}_h} \int_T u_h (\nabla \cdot \tilde{J}_h) &= \sum_{T \in \mathcal{T}_h} \int_{\partial T} \lambda_h \tilde{J} \cdot \nu \quad \forall \tilde{J}_h \in \mathcal{RT}_{-1}(\mathcal{T}_h), \\ \int_{\Omega} (\nabla \cdot J_h) \zeta_h &= - \int_{\Omega} \phi_h \zeta_h \quad \forall \zeta_h \in \mathcal{L}_0(\mathcal{T}_h), \\ \sum_{T \in \mathcal{T}_h} \int_{\partial T} \mu_h \hat{J} \cdot \nu &= 0 \quad \forall \mu_h \in \mathcal{L}_0(\mathcal{E}_h). \end{aligned}$$

Using additionally the matrices from Section 4.8.3 and $\hat{\mathbf{G}}: \mathcal{L}_0(\mathcal{T}_h) \rightarrow \mathcal{RT}_{-1}(\mathcal{T}_h)$ defined by

$$\hat{\mathbf{G}}_{sk} = \hat{\mathbf{G}}_{3n+m,k} = - \sum_{T \in \mathcal{T}_h} \int_T Z_k (\nabla \cdot \hat{\Psi}_{3n+m}),$$

this reads as

$$\begin{aligned} \hat{\mathbf{B}}_0 \hat{\underline{J}} - \hat{\mathbf{G}} \underline{u} - \mathbf{C} \hat{\underline{\lambda}} &= 0, \\ -\hat{\mathbf{G}}^t \hat{\underline{J}} &= -\mathbf{V} \underline{\phi}, \\ -\mathbf{C}^t \hat{\underline{J}} &= 0. \end{aligned}$$

Now remember that $\hat{\mathbf{B}}_0$ is block-diagonal, so that it can be easily inverted. Therefore, we can insert

$$\hat{\underline{J}} = \hat{\mathbf{B}}_0^{-1} (\hat{\mathbf{G}} \underline{u} + \mathbf{C} \hat{\underline{\lambda}})$$

into the second equation and get

$$\begin{aligned} -\hat{\mathbf{G}}^t \hat{\mathbf{B}}_0^{-1} \hat{\mathbf{G}} \underline{u} - \hat{\mathbf{G}}^t \hat{\mathbf{B}}_0^{-1} \mathbf{C} \hat{\underline{\lambda}} &= -\mathbf{V} \underline{\phi}, \\ -\mathbf{C}^t \hat{\mathbf{B}}_0^{-1} \hat{\mathbf{G}} \underline{u} - \mathbf{C}^t \hat{\mathbf{B}}_0^{-1} \mathbf{C} \hat{\underline{\lambda}} &= 0. \end{aligned}$$

To simplify notation, we set

$$\hat{\mathbf{D}} := \hat{\mathbf{G}}^t \hat{\mathbf{B}}_0^{-1} \hat{\mathbf{G}}, \quad \hat{\mathbf{E}} := \hat{\mathbf{G}}^t \hat{\mathbf{B}}_0^{-1} \mathbf{C} \quad \text{and} \quad \hat{\mathbf{F}} := \mathbf{C}^t \hat{\mathbf{B}}_0^{-1} \mathbf{C}$$

which yields

$$\begin{aligned} -\hat{\mathbf{D}} \underline{u} - \hat{\mathbf{E}} \hat{\underline{\lambda}} &= -\mathbf{V} \underline{\phi}, \\ -\hat{\mathbf{E}}^t \underline{u} - \hat{\mathbf{F}} \hat{\underline{\lambda}} &= 0. \end{aligned}$$

Now the matrix $\hat{\mathbf{D}}$ (note that this matrix has nothing to do with the divergence matrix \mathbf{D}) is diagonal, so that we can easily solve the first equation for \underline{u} ,

$$\underline{u} = \hat{\mathbf{D}}^{-1} (\mathbf{V} \underline{\phi} - \hat{\mathbf{E}} \hat{\underline{\lambda}}), \tag{A.12}$$

and insert this into the second equation, so that we finally get

$$(-\hat{\mathbf{E}}^t \hat{\mathbf{D}}^{-1} \hat{\mathbf{E}} + \hat{\mathbf{F}}) \hat{\lambda} = -\hat{\mathbf{E}}^t \hat{\mathbf{D}}^{-1} \mathbf{V} \phi. \quad (\text{A.13})$$

The matrix on the left-hand side is positive semi-definite and the kernel consists of constant $\hat{\lambda}_h$.

Equation A.13 is solved using the conjugate gradient method with incomplete Cholesky as preconditioner. The solver is capable of handling matrices where the kernel consists of the constant functions. Then, u_h is calculated using Equation (A.12).

A.5.2 ...using the piecewise constant energy

In this second case, we simply use the matrix \mathbb{T} , see (4.42) on page 67, for the discrete Laplacian and solve

$$\mathbb{T} \underline{u} = \underline{\phi}.$$

A.6 Computing the negative norm

To calculate the H^{-1} -norm of ϕ , we have to solve

$$-\Delta u = \phi$$

with either periodic or Neumann boundary conditions and then calculate

$$\|\nabla u\|_{L^2(\Omega)},$$

see Section 2.2.

Solving Poisson's problem was covered in the previous section, so it remains to compute the gradient of u_h in $\mathcal{RT}_0(\mathcal{E}_h)$ via

$$\mathbf{B}_0 \underline{g} = \mathbf{G} \underline{u},$$

and get the norm through

$$\|u_h\|_{H^{-1}(\Omega)}^2 = \|g_h\|_{L^2(\Omega)}^2 = \mathbf{B}_0 \underline{g} \cdot \underline{g}.$$

In the case of the piecewise constant energy, we use \mathbf{dx} and \mathbf{dy} to compute the gradient.

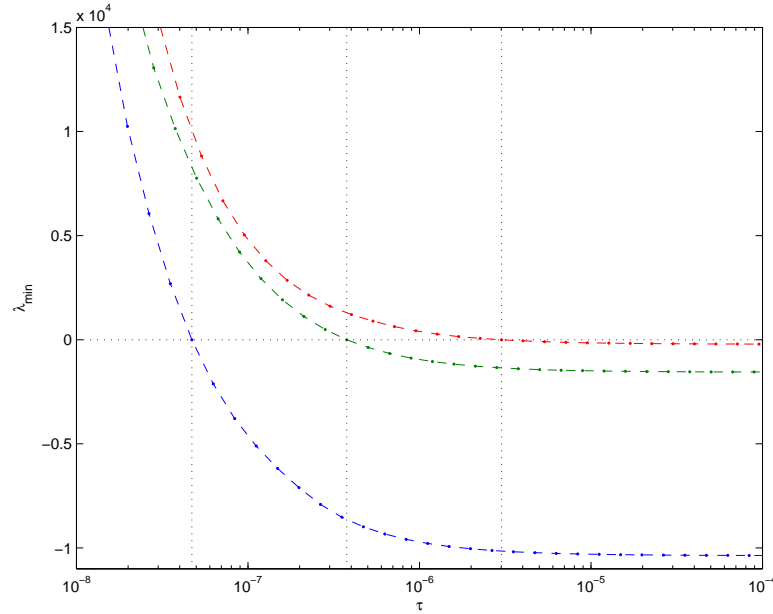


Figure A.5: Smallest eigenvalue of the matrix (A.15) versus time-step size for $\varepsilon = 1/16, 1/32$ and $1/64$. The vertical lines show the restriction (A.14).

A.7 Eigenvalues of the discrete operator

In Section 3.2.2, we investigated for which values of τ the operator is positive definite. A sufficient condition was

$$\tau < \frac{4}{\beta^2} \varepsilon^3. \quad (\text{A.14})$$

Here we present a simple numerical example where this condition is indeed necessary. We use the equations for epitaxial growth and use $\phi \equiv 1/2$, which yields $G''(\phi) \equiv -18$. The matrices defined in Section 4.7 were set up and the mass matrix was inverted. Since the mesh is very coarse (1536 degrees of freedom), it is possible to compute the full (dense) inverse. Then, the operator

$$\frac{1}{\tau} \mathbf{B}_0 + \varepsilon \mathbf{A}_0 \mathbf{B}_0^{-1} \mathbf{A}_0 + \varepsilon \mathbf{A}_1, \quad (\text{A.15})$$

see (4.24), was assembled for different values of τ and the eigenvalues were computed in Matlab. In Figure A.5, we plotted the smallest eigenvalue versus τ for different values of ε . We see that the constraint (A.14) is sharp.

A.8 Number of required Newton steps

To get an impression of how many Newton steps I are necessary in the algorithm in Table 4.1, we forced the software to use ten Newton steps in an exemplary simulation. In each Newton step, the system (4.24)–(4.26) was solved using the conjugate gradient (cg) method and the

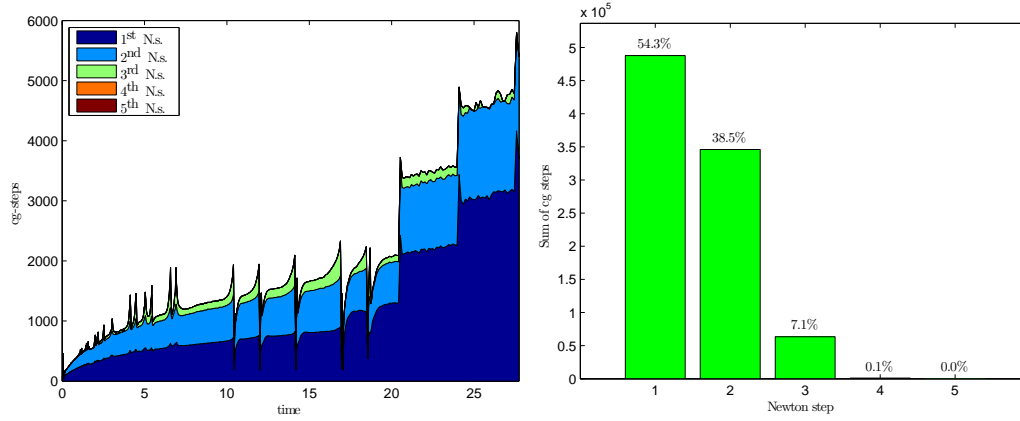


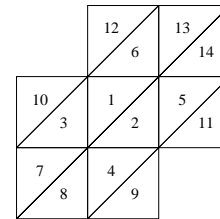
Figure A.6: Number of cg-steps per Newton-step. On the left, the number of cg-steps is shown for the first 1000 steps of a simulation. On the right, the sum of these cg-steps is shown. We see that Newton’s method almost always converged after three steps.

cg steps were counted. The result is shown in Figure A.6: three Newton steps are almost always enough.

A.9 Constructing the piecewise constant gradient

In Section 4.8.5 we constructed a gradient matrices \mathbf{dx} and \mathbf{dy} . We illustrate here the procedure using the simple mesh depicted on the right.

To be able to write the conditions (4.44a)–(4.44d) as a matrix, we enumerate the nonzero entries of \mathbf{dx} (remember that there are four nonzero entries per row) to get a vector



$$dx_{4k+\#l} = \mathbf{dx}_{kl},$$

where l is the $\#l$ -th nonzero entry in row k .

The system for the gradients on T_1 and T_2 , i.e. for the first two rows of \mathbf{dx} , consists of the eight conditions

$$\begin{pmatrix} \frac{Mx_1}{My_1} & \frac{Mx_6}{My_{-6}} & \frac{Mx_3}{My_3} & \frac{Mx_2}{My_2} & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{Mx_2}{My_2} & \frac{Mx_5}{My_5} & \frac{Mx_1}{My_1} & \frac{Mx_4}{My_4} \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ |T_1| & 0 & 0 & 0 & 0 & |T_2| & |T_2| & |T_2| \\ 0 & |T_1| & |T_1| & |T_1| & |T_2| & 0 & 0 & 0 \end{pmatrix} dx = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

The conditions (4.44a)–(4.44c) are represented in rows 1–3 and 4–6 for triangles T_1 and T_2 , respectively. Condition (4.44d) is represented in lines 7 and 8 for the two triangles.

It is now also apparent that the matrix has not full rank. Indeed, multiplying the third row with $|T_1|$ and adding the sixth row multiplied with $|T_2|$ is the same as the sum of the last two rows.

Using this information, we can transform the above system into a system of the form

$$\begin{pmatrix} \mathbf{M} & \underline{b} \\ 0 & 0 \end{pmatrix} \begin{pmatrix} \underline{x} \\ c \end{pmatrix} = \begin{pmatrix} \underline{r} \\ 0 \end{pmatrix}, \quad (\text{A.16})$$

where \mathbf{M} is a 7×7 Matrix, \underline{b} is the upper part of the last column and c is a scalar. For any value of c , we can now solve the system

$$\mathbf{M} \underline{x} = \underline{r} - c \underline{b},$$

where $\underline{x} = \underline{x}(c)$. Thus, solving the system for the values, say, $c = 0$ and $c = 1$, we can define

$$\underline{dx} = \begin{pmatrix} \underline{x}(0) \\ 0 \end{pmatrix} \quad \text{and} \quad \underline{zx} = \begin{pmatrix} \underline{x}(1) \\ 1 \end{pmatrix} - \begin{pmatrix} \underline{x}(0) \\ 0 \end{pmatrix}.$$

This yields a family of solutions $\underline{dx} + \lambda_x \underline{zx}$. Using the optimal λ_x as calculated in Section 4.8.5 yields the gradient and Laplacian shown in Figure A.7. For a different mesh, gradient and Laplacian are shown in Figure A.8.

A.10 Quadrature

In a finite element software, there are numerous occasions where the integration of a function over a triangle has to be carried out. First of all, this is necessary when assembling the matrices, but integration is also needed for the restriction operator, computation of norms and so on.

To compute

$$\int_T x^m y^n \, dx \, dy,$$

there are explicit formulas for $m, n \leq 2$ which only use the position of the vertices. Since our discrete functions are either piecewise constant or Raviart–Thomas vector fields, polynomials of order higher than two do not appear.

The only cases where we need to approximate integration by quadrature rules is when using the restriction operator. This is the case when setting the initial values or when comparing a discrete solution to a given continuous function, e.g. in Section 4.11.1.

In these cases we use the quadrature rules by Dunavant [1985] recommended to us by Jörg Drwenski.

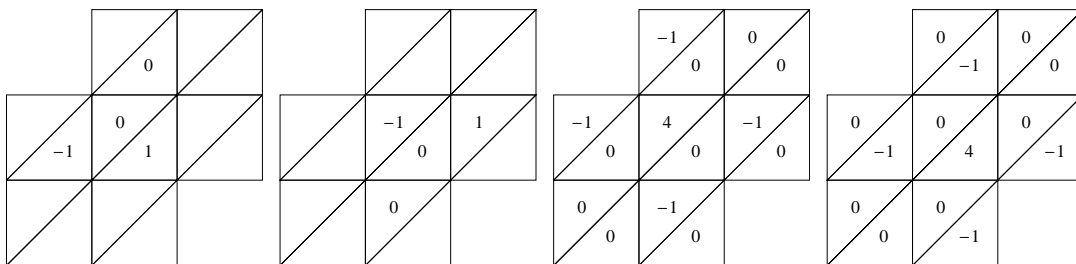


Figure A.7: From left to right: coefficients of the gradient \mathbf{dx} for triangles T_1 and T_2 (scaled by h) and coefficients for the Laplacian ∇ for triangles T_1 and T_2 (scaled by h^2). For each of the periodic sub-meshes, we get the coefficients known from finite differences.

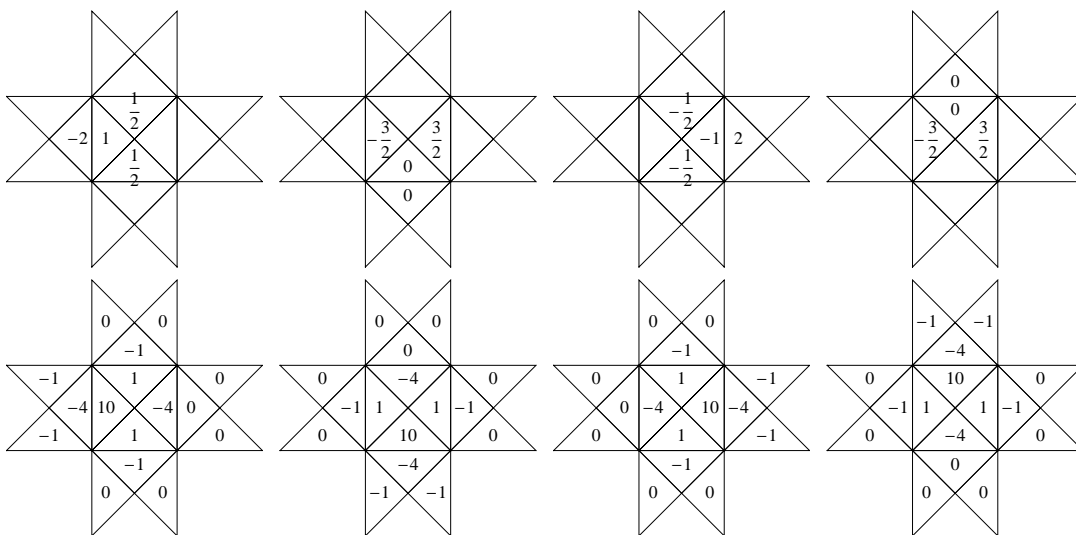


Figure A.8: Coefficients of \mathbf{dx} (top row) and ∇ (bottom row) for a different mesh. The Laplacian inherits the symmetry of the mesh.

A.11 Hardware, software versions and compiler options

Most of the simulations, including the method comparison in Section 4.11, were run on Intel® Core™ 2 Duo 1.86GHz CPUs (only one core was used) with 2GB RAM.

The operating system was Debian GNU/Linux 4.0 and the program was compiled with gcc 4.1.2 using the (standard) compiler options `-O3 -march=nocona -mfpmath=sse -msse3 -fPIC -fomit-frame-pointer`.

As external packages, we used PETSc 2.3.3p6 which in turn uses the BLAS routines of the Intel® MKL 8.0.1. For the Cholesky decomposition CHOLMOD 1.0.2 was used. Additionally, for the indefinite system (A.16) in PCE, we used UMFPACK 4.6.

For the few occasions where two GB of memory were not enough, we had an Intel® Xeon® 2GHz machine with 4GB RAM and the 64bit version of Debian 4.0.

Many Figures in this work were created using MATLAB® by The MathWorks in versions R2007a–R2008b, see also Section 7.4.3. Other figures were drawn using xFig 3.2.5-*a5*. The inheritance and collaboration diagrams in Chapter 7 were created by Doxygen 1.5.1. Also, we used GIMP 2.2.13, mainly to add transparency and for the title image.

The software was developed with the help of Eclipse with the CDT plugin. The TeXlipse plugin was used to write this work. Typesetting was done using pdfL^AT_EX.

Appendix B

Time-step schemes

We present here the time-step schemes we took under consideration together with their stability properties.

Table B.1: Values of a and γ for the different time-step schemes. Note that the second step of TR–BDF2 is not one-step in each substep, therefore correction (B.5) to Equation (3.7) applies.

	a	γ
Euler backward	1	0
Crank–Nicolson	1/2	1
BGP (step 1 and 3)	$3 - 2\sqrt{2}$	$1/\sqrt{2}$
BGP (step 2)	$3 - 2\sqrt{2}$	$\sqrt{2}$
TR–BDF2 (step 1)	$1 - 1/\sqrt{2}$	1
TR–BDF2 (step 2)	$1 - 1/\sqrt{2}$	$\sqrt{2} - 1$

B.1 Euler backward

For the first-order Euler backward scheme, we have $a = 1$ and $\gamma = 0$. Therefore $J_* = J^{k+1}$ and the right hand side is just f in Equation (3.4).

B.2 Crank–Nicolson

The simplest choice for a second-order scheme is the Crank–Nicolson (CN) scheme

$$\frac{\phi^{n+1} - \phi^n}{\tau} = \frac{1}{2}F(\phi^{n+1}) + \frac{1}{2}F(\phi^n),$$

i.e. $a = 1/2$ and $\gamma = 1$. Unfortunately, this scheme is not suited for Cahn–Hilliard-type equations due to its lack of strong A-stability: set $F(\phi) = A\phi$ with a self-adjoint linear operator A . Then, the scheme can be written as

$$\phi^{n+1} = (\text{id} - \frac{1}{2}\tau A)^{-1}(\text{id} + \frac{1}{2}\tau A)\phi^n.$$

Taking a basis of eigenfunctions of A , we have for the coefficient ϕ_i^{n+1} of the i -th eigenfunction

$$\phi_i^{n+1} = R_{\text{CN}}(\tau\lambda_i)\phi_i^n \quad \text{with} \quad R_{\text{CN}}(x) := \frac{(1 + \frac{1}{2}x)}{(1 - \frac{1}{2}x)}$$

and λ_i are the eigenvalues. As we can see, we have

$$\lim_{x \rightarrow \infty} |R_{\text{CN}}(x)| = 1.$$

Therefore, for stiff problems, some components of ϕ are always only poorly damped. Of course taking other values than $1/2$ for the operator splitting will lead to a first-order scheme, so this is no option.

B.3 Peaceman–Rachford-type

As a next step, we divide the time interval into two subintervals

$$[n\tau, (n+1)\tau] = [n\tau, (n+\theta)\tau] \cup [(n+\theta)\tau, (n+1)\tau]$$

$$\begin{array}{c} |-----|-----| \\ n \qquad \qquad n+\theta \qquad \qquad n+1 \end{array}$$

with $\theta \in (0, 1)$, where in the second subinterval the coefficients of the explicit and the implicit part are swapped. This yields Peaceman–Rachford (PR) type schemes

$$\begin{aligned} \frac{\phi^* - \phi^n}{\theta\tau} &= \alpha F(\phi^*) + (1 - \alpha)F(\phi^n), \\ \frac{\phi^{n+1} - \phi^*}{(1 - \theta)\tau} &= (1 - \alpha)F(\phi^{n+1}) + \alpha F(\phi^*). \end{aligned}$$

Note that with $\theta = 1/2$ and the extreme value $\alpha = 0$, this is the Crank–Nicolson scheme.

Again setting $F(\phi) = A\phi$ and using a basis of eigenfunctions yields

$$\phi_i^{k+1} = R_{\text{PR}}(\tau\lambda_i)\phi_i^k \quad \text{with} \quad R_{\text{PR}}(x) := \frac{(1 + \alpha(1 - \theta)x)(1 + (1 - \alpha)\theta x)}{(1 - (1 - \alpha)(1 - \theta)x)(1 - \alpha\theta x)}.$$

These schemes are of second order if

$$\alpha = \frac{1 - 2\theta}{2(1 + 2\theta)},$$

but we have again

$$\lim_{x \rightarrow \infty} |R_{\text{PR}}(x)| = 1,$$

so this class of schemes is also not suitable.

B.4 BGP

To overcome the limitations of the Crank–Nicolson and Peaceman–Rachford schemes, we follow Weikard [2002] and investigate a scheme by Bristeau, Glowinski, and Périaux [1987] (BGP scheme, called θ -scheme in their paper). The idea of the BGP scheme is to divide the time interval into *three* subintervals

$$[n\tau, (n+1)\tau] = [n\tau, (n+\theta)\tau] \cup [(n+\theta)\tau, (n+1-\theta)\tau] \cup [(n+1-\theta)\tau, (n+1)\tau]$$



with $\theta \in (0, 1/2)$. In the first two subintervals, the procedure is the same as in the PR scheme and in the third interval, the coefficients are once again swapped. Therefore, the BGP-scheme is

$$\frac{1}{\theta\tau}(\phi^* - \phi^n) = \alpha F(\phi^*) + (1-\alpha)F(\phi^n), \quad (\text{B.1a})$$

$$\frac{1}{(1-2\theta)\tau}(\phi^{**} - \phi^*) = (1-\alpha)F(\phi^{**}) + \alpha F(\phi^*), \quad (\text{B.1b})$$

$$\frac{1}{\theta\tau}(\phi^{n+1} - \phi^{**}) = \alpha F(\phi^{n+1}) + (1-\alpha)F(\phi^{**}). \quad (\text{B.1c})$$

Here, we find the stability function

$$R_{\text{BGP}}(x) = \frac{(1 + (1-\alpha)\theta x)^2 (1 + \alpha(1-2\theta)x)}{(1 - \alpha\theta x)^2 (1 - (1-\alpha)(1-2\theta)x)}$$

which yields in the limit

$$\lim_{x \rightarrow \infty} |R_{\text{BGP}}(x)| = \frac{1-\alpha}{\alpha}.$$

Thus, we need $\alpha > 1/2$ to get strong A-stability. Additionally, we want the scheme to be of second order in time. Expanding the stability function yields

$$R_{\text{BGP}}(x) = 1 + x + \frac{1}{2} \{ 1 + (1-2\alpha)(1-4\theta+2\theta^2) \} x^2 + \mathcal{O}(x^3).$$

Since we wanted $\alpha > 1/2$, the scheme is only second order accurate if

$$1 - 4\theta + 2\theta^2 = 0 \iff \theta = 1 \pm \frac{1}{\sqrt{2}}$$

and since $\theta \in (0, 1/2)$ we get $\theta = 1 - 1/\sqrt{2} \approx 0.29$.

We have still one free parameter: α . To choose α , we require the implicit parts in all three steps to be equal, i.e.

$$\alpha\theta = (1-\alpha)(1-2\theta) \iff \alpha = 2 - \sqrt{2} \approx 0.59.$$

Then, with $\theta\alpha = 3 - 2\sqrt{2} =: a$, we can rewrite scheme (B.1) as

$$\frac{1}{a\tau}(\phi^* - \phi^n) = F(\phi^*) + \gamma_1 F(\phi^n) \quad (\text{B.2a})$$

$$\frac{1}{a\tau}(\phi^{**} - \phi^*) = F(\phi^{**}) + \gamma_2 F(\phi^*) \quad (\text{B.2b})$$

$$\frac{1}{a\tau}(\phi^{n+1} - \phi^{**}) = F(\phi^{n+1}) + \gamma_1 F(\phi^{**}) \quad (\text{B.2c})$$

with $\gamma_1 = 1/\sqrt{2}$ and $\gamma_2 = \sqrt{2}$.

Concluding, we have a second-order scheme which is suited for stiff problems and the implicit part of the scheme is the same in all steps. Of course, the price we pay for these properties are two extra steps per time step.

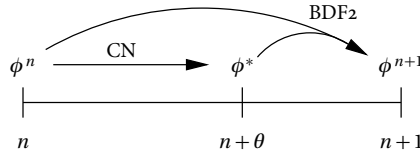
B.5 TR-BDF2

As we have seen, the above benefits cannot be achieved using an operator-splitting time stepping scheme which uses the splitting $\alpha F(\phi^{k+1}) + (1 - \alpha)F(\phi^k)$ in every substep and has less than three steps.

One option would be to change the operator splitting. Instead, we drop the requirement of being a one-step scheme in every substep and take a look at the TR-BDF2 scheme. It was developed by Bank et al. [1985] in the context of device simulation. It uses first a Crank-Nicolson step (or trapezoidal step, therefore TR) to produce an intermediate value ϕ^* at $t = t_n + \theta\tau$, $\theta \in (0, 1)$

$$\frac{1}{\frac{\theta}{2}\tau}(\phi^* - \phi^n) = F(\phi^*) + F(\phi^n)$$

and then uses the second-order backward-difference formula (BDF2) to compute the final value ϕ^{n+1} :



We briefly review the BDF2 formula. It uses a quadratic polynomial $p(t) = at^2 + bt + c$ with a, b, c such that

$$p(n\tau) = \phi^n, \quad p(t_n + \theta\tau) = \phi^* \quad \text{and} \quad p'(t_n + \tau) = F(\phi^{n+1}).$$

Then ϕ^{n+1} is given by $p(t_n + \tau)$. This yields

$$\phi^{n+1} = \phi^n + \frac{1}{\theta(2-\theta)}(\phi^* - \phi^n) + \tau \left(1 - \frac{1}{2-\theta}\right) F(\phi^{n+1})$$

and by inserting the CN step

$$\begin{aligned}\phi^{n+1} &= \phi^n + \tau \left[\frac{1}{2(2-\theta)} \left(F(\phi^n) + F(\phi^*) \right) + \frac{1-\theta}{2-\theta} F(\phi^{n+1}) \right] \\ \Leftrightarrow \frac{1}{\frac{1-\theta}{2-\theta}\tau} (\phi^{n+1} - \phi^n) &= \frac{1}{2(1-\theta)} \left(F(\phi^n) + F(\phi^*) \right) + F(\phi^{n+1}).\end{aligned}$$

The stability function is

$$R_T(x) = \frac{1 + \frac{2x}{(2-\theta)(2-\theta x)}}{1 - \frac{1-\theta}{2-\theta}x},$$

so for the limit we have

$$\lim_{x \rightarrow \infty} |R_T(x)| = 0,$$

i.e. L-stability, independent of θ . Expanding the stability function yields

$$R_T(x) = 1 + x + \frac{1}{2}x^2 + \mathcal{O}(x^3),$$

so the scheme is of second order for any θ . However, an accuracy of third order can not be attained since this would require $\theta = (1 \pm \sqrt{26})/3 \notin (0, 1)$.

As we want $\phi^{n+1} - \phi^*$ on the left hand side, we use

$$\frac{1}{\frac{1-\theta}{2-\theta}\tau} (\phi^* - \phi^n) = \frac{\theta(2-\theta)}{2(1-\theta)} \frac{1}{\frac{\theta}{2}\tau} (\phi^* - \phi^n) = \frac{\theta(2-\theta)}{2(1-\theta)} (F(\phi^*) + F(\phi^n))$$

and get for the BDF2 step

$$\frac{1}{\frac{1-\theta}{2-\theta}\tau} (\phi^{n+1} - \phi^*) = \underbrace{\left(\frac{1}{2(1-\theta)} - \frac{\theta(2-\theta)}{2(1-\theta)} \right)}_{=(1-\theta)/2} (F(\phi^n) + F(\phi^*)) + F(\phi^{n+1}).$$

To fix a θ , we again request the implicit parts in the TR and BDF2 steps to be the same, i.e.

$$a := \frac{1-\theta}{2-\theta} = \frac{\theta}{2} \Leftrightarrow \theta = 2 \pm \sqrt{2}.$$

Since $\theta \in (0, 1)$, we get

$$\theta = 2 - \sqrt{2} \approx 0.59 \quad \text{and} \quad a = 1 - \frac{1}{\sqrt{2}}.$$

This yields the TR-BDF2 scheme

$$\frac{1}{a\tau} (\phi^* - \phi^n) = F(\phi^*) + F(\phi^n) \tag{B.3a}$$

$$\frac{1}{a\tau} (\phi^{n+1} - \phi^*) = F(\phi^{n+1}) + \frac{\sqrt{2}-1}{2} (F(\phi^*) + F(\phi^n)). \tag{B.3b}$$

Note that no information from previous time-steps is needed. Now remember our notation: k in Formula (3.4) counts the substeps *relative* to the current substep and n are the (top-level) time-steps, so in the first step

$$\phi^k = \phi^n \quad \text{and} \quad \phi^{k+1} = \phi^* = \phi^{n+\theta}$$

and in the second step

$$\phi^{k-1} = \phi^n, \quad \phi^k = \phi^* = \phi^{n+\theta} \quad \text{and} \quad \phi^{k+1} = \phi^{n+1}.$$

Thus, we can write the two steps as

$$\frac{1}{a\tau}(\phi^{k+1} - \phi^k) = F(\phi^{k+1}) + \frac{\gamma}{2} (F(\phi^k) + F(\phi^{k-1})). \quad (\text{B.4})$$

with

$$\begin{array}{ll} \gamma = 1, & \phi^{k-1} := \phi^k & \text{in the first substep,} \\ \gamma = \sqrt{2} - 1 & & \text{in the second substep.} \end{array}$$

To be able to use this scheme in our setup, we change Equation (3.1) to the new definition (B.4). Repeating the calculations of Section 3.1, we can continue to use Equation (3.7) with the new definition

$$w_{\text{TR-BDF}}^* = w^i + \frac{\gamma}{2} (w^k + w^{k-1}). \quad (\text{B.5})$$

In total, we have more advantages than in BGP (namely L-stability) while only having to perform one extra step instead of two.

Symbols

$E(\phi)$	Energy functional, page 12.
$E_h(\phi_h)$	Discrete energy, page 79.
$G(\phi)$	Potential term in the energy, page 15.
$H(\nabla\cdot, \Omega)$	L^2 vector fields with divergence in $L^2(\Omega)$, page 38.
$H(\nabla\nabla\cdot, \Omega)$	$L^2(\Omega)^2$ vector fields with divergence in $H^1(\Omega)$, page 29.
$H^m(\nabla\cdot, \Omega)$	$H^m(\Omega)^2$ vector fields with divergence in $H^m(\Omega)$, page 45.
$H^m(\nabla\nabla\cdot, \Omega)$	$H^m(\Omega)^2$ vector fields with divergence in $H^{m+1}(\Omega)$, page 51.
K	$-\nabla\nabla\cdot J_*^{i+1}$, page 38.
$M(\phi)$	Mobility, page 15.
N_E	Number of edges in the mesh, page 40.
N_T	Number of triangles in the mesh, page 40.
N_V	Number of vertices in the mesh, page 40.
P_h^E	L^2 -orthogonal projection onto Raviart–Thomas vector fields, page 41.
P_h^T	L^2 -orthogonal projection onto piecewise constant functions, page 41.
R_h^T	Restriction operator $R_h^T : L^2(\Omega) \rightarrow \mathcal{L}_0(\mathcal{T}_h)$, page 41.
R_h^E	Restriction operator $R_h^E : H(\nabla\cdot, \Omega) \cap L^s(\Omega)^2 \rightarrow \mathcal{RT}_0(\mathcal{E}_h)$, page 44.
T_i^\pm	Triangles adjacent to E_i , page 43.

$\Psi_i(x)$	Basis function on $\mathcal{RT}_0(\mathcal{E}_h)$ to edge i , page 43.
$ E_i $	Length of edge E_i , page 40.
$ T_k $	Volume of triangle T_i , page 40.
$\delta E(\phi)/\delta \phi$	Variational derivative of E , page 6.
$\delta^2 E(\phi)/\delta \phi^2$	Second variational derivative of E , page 6.
\mathcal{E}_h	Set of all edges in the mesh, page 40.
∇_h	Discrete gradient, page 45.
κ	Curvature, page 21.
$\ \cdot\ _{H^{-1}(\Omega)}$	Norm in the dual of $H^1(\Omega)$, page 9.
$\ \cdot\ _{H_M^{-1}(\Omega)}$	Weighted H^{-1} norm, page 10.
ν	Outer normal, page 21.
$\mathcal{L}_0(\mathcal{T}_h)$	Space of piecewise constant functions, page 41.
$\mathcal{RT}_0(\mathcal{E}_h)$	Space of lowest-order Raviart–Thomas finite elements, page 43.
\mathcal{T}_h	Set of all triangles of the mesh, page 40.
g^\pm	Bulk equilibrium values, page 18.
h_T	Half the diameter of the triangle T , page 75.
w	Chemical potential, short for $\delta E(\phi)/\delta \phi$, page 28.
(G.i)–(G.iv)	Properties of the discrete gradient, page 46.

Bibliography

- R. A. Adams. *Sobolev spaces*. Academic Press, second edition, 2003. Pure and Applied Mathematics, Vol. 140.
- N. Alikakos, P. Bates, and X. Chen. Convergence of the Cahn–Hilliard equation to the Hele–Shaw model. *Arch. Ration. Mech. An.*, 128:165–205, 1994.
- P. R. Amestoy, T. A. Davis, and I. S. Duff. An approximate minimum degree ordering algorithm. *SIAM J. Matrix Anal. Appl.*, 17(4):886–905, 1996.
- D. N. Arnold and F. Brezzi. Mixed and nonconforming finite element methods: implementation, postprocessing and error estimates. *RAIRO Modél. Math. Anal. Numér.*, 19(1):7–32, 1985.
- S. Balay, K. Buschelman, W. D. Gropp, D. Kaushik, M. G. Knepley, L. C. McInnes, B. F. Smith, and H. Zhang. PETSc Web page, 2001. <http://www.mcs.anl.gov/petsc>.
- G. Bales and A. Zangwill. Morphological instability of a terrace edge during step-flow growth. *Phys. Rev. B*, 41:5500, 1990.
- R. Bank, W. Coughran, W. Fichtner, E. Gross, D. Rose, and R. Smith. Transient simulation of silicon devices and circuits. *IEEE T. Comput. Aid. D.*, 4:436–451, 1985.
- F. Bornemann and C. Rasch. Finite-element discretization of static Hamilton-Jacobi equations based on a local variational principle. *Comput. Vis. Sci.*, 9(2):57–69, 2006.
- F. Brezzi and M. Fortin. *Mixed and hybrid finite element methods*, volume 15 of *Springer Series in Computational Mathematics*. Springer-Verlag, New York, 1991.
- M. O. Bristeau, R. Glowinski, and J. Périaux. Numerical methods for the Navier-Stokes equations. Applications to the simulation of compressible and incompressible viscous flows. In *Finite elements in physics (Lausanne, 1986)*, pages 73–187. North-Holland, Amsterdam, 1987.
- W. K. Burton, N. Cabrera, and F. C. Frank. The growth of crystals and the equilibrium of their surfaces. *Phil. Trans. Roy. Soc. London Ser. A*, 243(866):299–358, 1951.
- J. Cahn and J. Hilliard. Free energy of a nonuniform system. 1. Interfacial free energy. *J. Chem. Phys.*, 28:258–267, 1958.

- J. W. Cahn, C. M. Elliott, and A. Novick-Cohen. The Cahn-Hilliard equation with a concentration dependent mobility: motion by minus the Laplacian of the mean curvature. *European J. Appl. Math.*, 7(3):287–301, 1996.
- X. Chen. Spectrum for the Allen-Cahn, Cahn-Hilliard, and phase-field equations for generic interfaces. *Comm. Partial Differential Equations*, 19(7-8):1371–1395, 1994.
- G. Danker, O. Pierre-Louis, K. Kassner, and C. Misbah. Interrupted coarsening of anisotropic step meander. *Phys. Rev. E*, 68:020601, 2003.
- T. A. Davis. A column pre-ordering strategy for the unsymmetric-pattern multifrontal method. *ACM Trans. Math. Softw.*, 30(2):165–195, 2004.
- T. A. Davis and W. W. Hager. Row modifications of a sparse Cholesky factorization. *SIAM J. Matrix Anal. Appl.*, 26(3):621–639 (electronic), 2005.
- P. Deuffhard and F. Bornemann. *Numerische Mathematik. II.* de Gruyter Lehrbuch. Walter de Gruyter & Co., Berlin, 1994. Integration gewöhnlicher Differentialgleichungen. [Integration of ordinary differential equations].
- C. Domb and J. L. Lebowitz, editors. *Phase transitions and critical phenomena. Vol. 8.* Academic Press Inc., London, 1983.
- D. Dunavant. High degree efficient symmetrical gaussian quadrature-rules for the triangle. *Int. J. Numer. Methods Eng.*, 21(6):1129–1148, 1985.
- G. Ehrlich and F. G. Hudda. Atomic view of surface diffusion: tungsten on tungsten. *J. Chem. Phys.*, 44:1036–1099, 1966.
- C. M. Elliott and H. Garcke. On the Cahn-Hilliard equation with degenerate mobility. *SIAM J. Math. Anal.*, 27(2):404–423, 1996.
- C. M. Elliott and Z. Songmu. On the Cahn-Hilliard equation. *Arch. Rational Mech. Anal.*, 96(4):339–357, 1986.
- C. Geiger and C. Kanzow. *Theorie und Numerik restringierter Optimierungsaufgaben.* Springer-Verlag Berlin Heidelberg, 2002.
- F. Haußer and A. Voigt. Step meandering in epitaxial growth. *J. Cryst. Growth*, 303(1):80–84, 2007.
- F. Haußer, F. Otto, P. Penzler, and A. Voigt. Numerical methods for the simulation of epitaxial growth and their application in the study of a meander instability. In *Mathematics—Key Technology for the Future, Joint Projects between Universities and Industry, 2003–2007.* Springer, 2008.
- M. E. Hosea and L. F. Shampine. Analysis and implementation of TR-BDF2. *Appl. Numer. Math.*, 20(1-2):21–37, 1996. Workshop on the method of lines for time-dependent problems (Lexington, KY, 1995).
- A. Karma and M. Plapp. Spiral surface growth without desorption. *Phys. Rev. Lett.*, 81:4444–4447, 1998.

- R. V. Kohn and F. Otto. Upper bounds on coarsening rates. *Comm. Math. Phys.*, 229(3): 375–395, 2002.
- L. Y. Kolotilina and A. Y. Yeremin. Factorized sparse approximate inverse preconditionings. I. Theory. *SIAM J. Matrix Anal. Appl.*, 14(1):45–58, 1993.
- J. Krug. Introduction to step dynamics and step instabilities. In A. Voigt, editor, *Multiscale modeling of epitaxial growth*, volume 149 of *ISNM*. Birkhäuser, 2005.
- B. Li, A. Rätz, and A. Voigt. Stability of a circular epitaxial island. *Physica D*, 198(3-4):231–247, 2004.
- C. C. Liu, Y. W. Qi, and J. X. Yin. Regularity of solutions of the Cahn-Hilliard equation with non-constant mobility. *Acta Math. Sin. (Engl. Ser.)*, 22(4):1139–1150, 2006.
- F. Liu and H. Metiu. Stability and kinetics of step motion on crystal surfaces. *Phys. Rev. E*, 49:2601–2616, 1997.
- T. Maroutian, L. Douillard, and H. Ernst. Wavelength selection in unstable homoepitaxial step flow growth. *Phys. Rev. Lett.*, 83:4353, 1999.
- W. McCray. MBE deserves a place in the history books. *Nat. Nanotechnol.*, 2:259–261, May 2007.
- L. Modica and S. Mortola. Un esempio di Γ -convergenza. *Boll. Un. Mat. Ital. B (5)*, 14(1): 285–299, 1977.
- B. Niethammer and F. Otto. Ostwald ripening: the screening length revisited. *Calc. Var. Partial Differential Equations*, 13(1):33–68, 2001.
- A. Novick-Cohen. The Cahn-Hilliard equation: mathematical and modeling perspectives. *Adv. Math. Sci. Appl.*, 8(2):965–985, 1998.
- A. Oron, S. H. Davis, and S. G. Bankoff. Long-scale evolution of thin liquid films. *Rev. Mod. Phys.*, 69(3):931–980, 1997.
- F. Otto, P. Penzler, A. Rätz, T. Rump, and A. Voigt. A diffusive-interface approximation for step flow in epitaxial growth. *Nonlinearity*, 17(2):477–491, 2004.
- F. Otto, P. Penzler, and T. Rump. Discretisation and numerical tests of a diffuse-interface model with Ehrlich-Schwoebel barrier. In *Multiscale modeling in epitaxial growth*, volume 149 of *Internat. Ser. Numer. Math.*, pages 127–158. Birkhäuser, Basel, 2005.
- P. Papon, J. Leblond, and P. H. Meijer, editors. *The Physics of Phase Transitions*. Springer, Heidelberg, 2002.
- R. Pego. Front migration in the nonlinear Cahn-Hilliard equation. *P. Roy. Soc. A*, 422:261–278, Apr 1989.
- O. Pierre-Louis, C. Misbah, Y. Saito, J. Krug, and P. Politi. New nonlinear evolution equation for steps during molecular beam epitaxy. *Phys. Rev. Lett.*, 80:4221, 1998.

- O. Pierre-Louis, G. Danker, J. Chang, K. Kassner, and C. Misbah. Nonlinear dynamics of vicinal surfaces. *J. Cryst. Growth*, 275:56, 2005.
- P. Politi, G. Grenet, A. Marty, A. Ponchet, and J. Villain. Instabilities in crystal growth by atomic or molecular beams. *Phys. Rep.*, 324:271, 2000.
- C. Polop, S. Bleikamp, and T. Michely. I. Phys. Institut, RWTH Aachen.
- A. Rätz. *Modelling and Numerical Treatment of Diffuse Interface Models with Applications in Epitaxial Growth*. PhD thesis, Universität Bonn, 2007.
- P.-A. Raviart and J. M. Thomas. A mixed finite element method for 2nd order elliptic problems. In *Mathematical aspects of finite element methods (Proc. Conf., Consiglio Naz. delle Ricerche (C.N.R.), Rome, 1975)*, pages 292–315. Lecture Notes in Math., Vol. 606. Springer, Berlin, 1977.
- J. S. Rowlinson. Translation of J. D. van der Waals' "The thermodynamic theory of capillarity under the hypothesis of a continuous variation of density". *J. Stat. Phys.*, 20(2):197–244, Feb 1979.
- T. Rump. *Coarsening processes in thin liquid films: Analysis and numerics*. PhD thesis, Universität Bonn, 2008.
- J. Taylor and J. Cahn. Linking anisotropic sharp and diffuse surface motion laws via gradient flows. *J. Stat. Phys.*, 77(1-2):183–197, 1994.
- A. Voigt, editor. *Multiscale modeling in epitaxial growth*, volume 149 of *International Series of Numerical Mathematics*. Birkhäuser Verlag, Basel, 2005. Selected papers from the workshop held in Oberwolfach, January 18–24, 2004.
- U. Weikard. *Numerische Lösungen der Cahn-Hilliard-Gleichung und der Cahn-Larché-Gleichung*. Dissertation, Rheinische Friedrich-Wilhelms-Universität Bonn, Oct 2002.