Numerical Simulation of Dendritic Solidification Using a Phase Field Model

Christer Andersson

Stockholm 2000

Licentiate's Thesis
Royal Institute of Technology
Department of Numerical Analysis and Computer Science

TRITA-NA-0013
ISSN 0348-2952
ISRN KTH/NA/R--00/13--SE

© Christer Andersson, May 2000

Nada, Stockholm 2000
Abstract

Solidification of a pure material from its undercooled melt is considered, taking into account effects of surface tension, kinetic undercooling and anisotropy. The resulting crystals exhibit complicated dendritic shapes. To avoid explicit tracking of the solid-liquid interfaces a phase field model is employed. A diffuse interface of artificial width is introduced and the solidification is modeled by a set of reaction-diffusion equations valid in the entire computational domain.

The phase field equations are discretized in space using second order finite differences on uniform Cartesian grids. A semi-explicit first order accurate time stepping scheme based on equation splitting is compared to a more accurate fully implicit scheme. Numerical methods for efficient computer implementation are discussed. Convergence rates are experimentally validated for decreasing time steps and it is shown that for some cases the first order scheme may fail to capture the correct growth rate of the crystal.

A formal asymptotic analysis of the phase field model is presented in the limit of vanishing diffuse interface thickness. It was recently shown how an increased accuracy in the asymptotics can be used to obtain better agreement between the phase field model and the corresponding sharp interface formulation. This has implications on computational efficiency by allowing simulation with larger diffuse interface width which is verified by numerical experiments.

An implementation of the fully implicit scheme has been written for parallel distributed memory architectures. Experiments conducted on an IBM SP2 and networks of Sun Ultra 5 workstations show that the code scales well with the number of processors.

ISBN 91-7170-590-5 • TRITA-NA-0013 • ISSN 0348-2952 • ISBN KTH/NA/R-00/13--SE
## Contents

1 Introduction ............................................. 1
   1.1 Background ........................................ 1
   1.2 The Present Work ................................ 3

2 A Phase Field Model .................................. 5
   2.1 The Stefan Problem ................................ 5
   2.2 Formulation of the Phase Field Model ........... 6
      2.2.1 Non-dimensionalization ....................... 9
      2.2.2 Anisotropy .................................. 10
   2.3 A Formal Asymptotic Analysis ..................... 11
      2.3.1 Asymptotic Ansatz ............................. 12
      2.3.2 Zero-order Inner Solution ................... 14
      2.3.3 First Order Inner Solution ................... 14
      2.3.4 Second Order Inner Solution ................ 16
      2.3.5 Interface Conditions ......................... 18

3 Numerical Approximation ............................. 21
   3.1 Spatial Discretization ............................. 21
      3.1.1 One Space Dimension ......................... 21
      3.1.2 Two Space Dimensions ......................... 23
   3.2 Temporal Discretization ......................... 24
      3.2.1 Methods Based on Equation Splitting ....... 24
      3.2.2 Methods Based on Full Discretization ...... 25
   3.3 Non-linear Solver ................................ 26
      3.3.1 Newton Iteration ............................ 26
      3.3.2 Krylov Subspace Projection Methods ....... 28
   3.4 A Fast Poisson Solver .............................. 30

4 Parallel Implementation ............................. 33
   4.1 Parallelization Issues ............................ 33
      4.1.1 Distributed Parallel Computing ............... 33
      4.1.2 Data Distribution ............................ 34
   4.2 Implementation .................................... 35
   4.3 Computational Performance ....................... 37
5 Numerical Results 41
  5.1 A Comparison of Time Stepping Schemes 41
     5.1.1 Execution Time in Two Space Dimensions 45
  5.2 Sharp vs. Thin Interface Asymptotics 46
  5.3 Computed Dendrites 48
     5.3.1 A Reproduced Example 48
     5.3.2 Effects of anisotropy 50

6 Conclusions 55
Acknowledgments

First and foremost I would like to thank my advisor Gunilla Kreiss for invaluable support, guidance and encouragement throughout my research.

I would also like to thank Gustav Amberg, Irina Loginova, Robert Tönhardt and John Ågren for many stimulating and informative discussions on phase field methods.

Financial support from the Foundation for Strategic Research (SSF) through the National Graduate School in Scientific Computing (NGSSC) is gratefully acknowledged.
Chapter 1

Introduction

1.1 Background

We are interested in accurately simulating the solidification of a pure material from its undercooled melt taking into account effects of surface tension, kinetic undercooling and crystalline anisotropy. This is perhaps the simplest physical process in which one can observe the growth of dendritic crystals, characterized by complicated tree-like patterns. Since some properties of solids, such as electric conductivity, mechanical strength and ductility are determined by the microstructures produced upon solidification one would like to better understand the pattern formation in order to obtain the desired properties in manufacture. Applications can be found in, for example, casting and semiconductor production.

The classical mathematical description has the form of a Stefan problem, in which two diffusion equations govern the transport of heat in the solid and liquid phases, coupled by boundary conditions prescribed at the solid-liquid interface. Since the shape is changing in time it is a free boundary problem (or moving boundary problem) for which it has proved very difficult to find analytic solutions of practical use. For this reason a considerable effort has been put into developing numerical algorithms in the last twenty years.

Front-tracking techniques fit naturally in the framework of free boundary problems, and have been employed by several authors, see e.g. [1, 21]. The interface is represented by a discrete set of marker points, which can be connected by interpolating polynomials to obtain the location of the free boundary. The time-dependent problem can be solved by advecting the marker points according to the velocity given by boundary conditions, and solving the heat equations in solid and liquid given the location of the boundary. Due to the complicated interfacial shapes which arise it will be necessary to add and delete points during the course of the simulation. One can also expect problems with merging fronts, where the connectivity of the points will have to be redefined. The approach becomes increasingly difficult as one increases the number of space dimensions.

Some of the difficulties of front-tracking can be avoided by use of boundary integral methods. The physical domain is transformed so that the interface is mapped to a known location in the transformed space. The growth of the crystal can be determined by solving a heat equation coupled with an integral equation on the free boundary. Geometric properties such as curvature are readily available from the parameterization and boundary conditions can be accurately enforced, but merging interfaces is still a problem. In the current context it will also be necessary to use local parameterizations due to complicated geometries. Some of the largest computations to date have been based on boundary integral formulations [44].
Most of the difficulties in the methods discussed so far come from having to track the position of the free boundary. The phase field approach is another way of formulating the solidification problem which eliminates the need for explicit boundary representation \([37, 47]\). The key ingredient is to postulate the existence of a smooth approximation of an indicator function taking on distinct values in solid and liquid, the so-called phase field. Using ideas from continuum models of phase transitions, a set of reaction-diffusion equations valid in the entire computational domain can be derived. The sharp interface of width zero is replaced by a diffuse interface of width \(\epsilon\), in which the phase field variable changes value. One can prove by rigorous asymptotic analysis that the Stefan problem is recovered as \(\epsilon \to 0\) \([16]\).

The main advantage of the phase field methods is that the location of the solid-liquid interface is given implicitly by the phase field, which greatly simplifies the handling of merging interfaces. There is however a price to pay. The boundary has one dimension lower than the computational domain, but to avoid explicit representation a variable defined in the entire domain is introduced. Furthermore, in order to accurately represent the boundary the grid resolution must be at least of order \(\epsilon\), which implies large computational grids since \(\epsilon\) should be small compared to the smallest length scale of the dendrite in order to recover the Stefan problem.

Despite these drawbacks the phase field methods have become increasingly popular in numerical simulation over the past ten years. Since the first introduction by Langer 15 years ago \([32]\) the technique has been applied to a number of problems, including solidification of pure materials \([37, 47]\), solidification of binary alloys \([11, 34, 49]\), solidification under external flow \([45, 46]\), and solid to solid phase transformations \([48]\). Both qualitative \([29]\) and quantitative \([5, 25, 51]\) agreement between phase field models and sharp interface formulations have been investigated for the problem considered here. In recent years refined modeling \([2, 14, 23]\) and the use of adaptive methods \([5, 25, 38]\) has increased the parameter range available for simulation and allowed quantitative comparison with physical experiments \([24]\).

Level set methods are an alternative to phase field modeling which avoid explicit front tracking while being able to handle complicated geometries and merging interfaces \([12, 36]\). In contrast to phase field methods they are based on sharp interface formulations, and do not require any additional physical modeling. The underlying idea is to introduce a level set function, which is a signed distance function defined in the entire domain, measuring the distance from a point to the closest interface. Thus the interface is implicitly given by the zeros of the level set function.

In simulation the level set function is advected according to a velocity field defined through an extension of the interface conditions, and the temperature updated in such a way that the sharp interface formulation of the Stefan problem is satisfied. In practice it will also be necessary to reinitialize the level set function every few time steps to ensure that it remains a signed distance function.

Level set methods do not seem to be as popular as phase field methods for solidification simulations and have not been considered in this work. The implementation requires some additional technicalities, and it seems more difficult to generalize the approach to similar physical problems such as solidification of alloys. Their main advantage over phase field methods is that convergence to the physical problem does not depend on the artificial parameter \(\epsilon\).
1.2 The Present Work

This work is primarily an investigation of numerical methods for simulation of dendritic solidification using a phase field model. The considered model consists of a singularly perturbed set of reaction-diffusion equations, where the variation of the solutions is highly localized in space. All computations have been performed using uniform finite difference discretizations in space, and constant time steps. The purpose is to find sufficiently general schemes to allow extension to adaptivity in space and time, as well as to more general physical problems.

Since the phase field variable is not conserved it is believed that an accurate numerical method must be used to obtain the correct propagation speed of the interface. A fully implicit time stepping scheme of up to sixth order accuracy is formulated based on backward difference formulas (BDF), and compared to a common first order accurate semi-explicit approach based on equation splitting [25, 51]. Asymptotic convergence rates in time are verified and it is shown that for cases where the crystal is growing fast the higher order accuracy is important. The fully implicit scheme can be more readily generalized to adaptive grids. The efficiency of the semi-explicit scheme relies heavily on the uniform grid due to the use of a fast Poisson solver. It may also be difficult to utilize the adaptivity to reduce the grid spacing in the vicinity of the interfaces due to stability restrictions.

A formal asymptotic analysis from [14] has been applied to the phase field model under consideration. In [22, 23] it was observed that reinterpreting the asymptotics and modifying the value of one dimensionless parameter leads to increased accuracy in simulation and enables computation on smaller grids. Numerical experiments are presented to validate this.

A second order accurate numerical scheme has been implemented for parallel distributed memory architectures. The parallel efficiency of the implementation exceeds 70% on 16 IBM SP2 processors even for grids as small as 200 × 200 points, thus significantly reducing run times and allowing simulation on large grids.

The outline of the paper is as follows: Chapter 2 discusses the formulation of the phase field model. A modified Stefan problem which includes effects of surface tension, kinetic undercooling and crystalline anisotropy is introduced in Section 2.1. The development of the phase field model used for the simulations is then outlined in Section 2.2. A formal asymptotic analysis is performed in Section 2.3, and the parameters of the phase field model are related to those of the Stefan problem. Implications of different interpretations of the asymptotics on computational efficiency based on observations from [23] are briefly discussed.

The numerical schemes used for simulation are described in Chapter 3. The spatial discretizations are based on finite differences on uniform Cartesian grids, and described in Section 3.1. Section 3.2 discusses two different approaches for the temporal discretization via the method of lines. One of these leads to large non-linear systems of equations, which are solved using techniques outlined in Section 3.3. In the other case it is sufficient to solve linear systems which can be done efficiently using a fast Poisson solver as described in Section 3.4.

An implementation has been written for parallel distributed memory architectures to enable simulation on large grids. In Section 4.1 some of the difficulties of parallel computing are briefly reviewed, and efficient implementation of the numerical schemes discussed. The actual implementation is described in Section 4.2, and its parallel performance investigated in Section 4.3. The implementation is verified against results previously published by Braun and Murray [5] in Section 3.3.1.
Chapter 5 presents some numerical results. The two different temporal discretizations are investigated with respect to accuracy and computational efficiency in Section 5.1. Implications of different asymptotics on computational efficiency and agreement with the Stefan problem are investigated in Section 5.2. Effects of grid anisotropy and random perturbation on computed dendrites are briefly examined in Section 5.3.
Chapter 2

A Phase Field Model

2.1 The Stefan Problem

It is instructive to first consider the sharp interface formulation of the solidification problem since it is more amenable to physical interpretation. The formulation will also be needed in Section 2.3 to relate microscopic parameters in the phase field model to macroscopic quantities which can be measured in physical experiments. More details on the underlying physics can be found in [30], [31] and [50].

To formulate the equations we split the computational domain $\Omega$ into two regions, $\Omega_+(t)$ and $\Omega_-(t)$ corresponding to the solid and liquid phases at a given time $t$, separated by the free boundary $\Gamma(t)$, see Figure 2.1. $\theta$ is defined as the angle between the outward normal and some reference direction, and is needed to account for preferred growth directions in the crystal. In three dimensions $\theta$ should be a vector of two components to properly account for anisotropy.

Given an initial interface location, temperature distribution, and outer boundary conditions, we wish to determine the interface location $\Gamma(t)$ and temperature $T(x, t)$. Taking into account effects of surface tension, kinetic undercooling and anisotropy one can formulate a Stefan problem in physical variables,

$$
\begin{align*}
\partial_t T &= D \nabla^2 T & x \in \Omega_+(t) \\
-cD\partial_n T &= -Lv_n & x \in \Gamma(t) \\
T &= T_M - \tilde{\sigma}(\theta) |\mathcal{K}|L - v_n/\mu(\theta) & x \in \Gamma(t)
\end{align*}
$$

(2.1)

where $T_M$ is the melting temperature, $D$ the thermal diffusivity, $c$ the heat capacity, $L$ the latent heat of fusion, $v_n$ the normal velocity of the interface, $\mathcal{K}$ the interfacial curvature taken to be positive if the center of curvature is inside the solid, $\mu$ the interfacial mobility and $\tilde{\sigma}(\theta)$ a function depending on the surface tension $\sigma(\theta)$.

In (2.1) it has been assumed that all physical parameters are constant and equal in the bulk solid and liquid phases. This is a valid assumption for certain materials, such as succinonitrile (SCN) and will reduce the amount of notation needed. The assumption is not vital for the phase field modeling [2].

The first equation of (2.1) describes the diffusion of heat inside the bulk solid and liquid phases. The second equation gives an expression for the interfacial velocity, proportional to the discontinuity in the normal derivative of the temperature across the interface. It corresponds to release and/or absorption of latent heat and is needed for conservation of
energy. The third equation, or the Gibbs-Thomson condition, accounts for the change of melting temperature according to kinetic and capillary effects.

The Gibbs-Thomson condition of (2.1) is not present in the classical Stefan problem often seen in mathematical textbooks. Instead the third equation is replaced by $T = T_M$, i.e. it is assumed that the phase transition takes place at equilibrium. The more complicated boundary condition is needed in order to obtain dendritic structures. In an attempt to motivate why the classical Stefan problem is not sufficient in the current context, and explain why the dendritic structures occur, the underlying physics of the “pattern selection process” will be briefly overviewed.

A pure liquid is held at a temperature below melting temperature. This is a metastable state and solidification can be initiated by disturbing the system, e.g. by adding a crystal seed. As solidification proceeds latent heat will be released at the solid-liquid interface, and the growth rate will depend on how fast this heat can be conducted away. From this point of view it is favorable to have a large interfacial area since it implies faster dissipation. On the other hand, the interface is associated with a surface energy which would in turn imply that a small interfacial area is to prefer. The actual shape of the crystal is determined by a compromise between these two competing effects. Surface energy has a stabilizing effect on the growth since it determines a smallest scale in the geometry. The process is unstable in the sense that small perturbations in initial data can significantly alter the time-dependent shape of the crystal.

The Stefan problem (2.1) is significant for applications in semiconductor production where one is interested in the growth of pure crystals. In order to simulate realistic casting (2.1) needs to be complemented with equations to account for the diffusion of alloying elements. For each additional component in the alloy a diffusion equation for a concentration is added to the Stefan problem, and also an interface condition prescribing a discontinuity in the concentration across the interface. The phase field approach has been successfully used to model solidification of binary alloys, see e.g. [11, 49].

### 2.2 Formulation of the Phase Field Model

This section gives a brief outline of the derivation of the phase field model used for computations. The presentation is based on the work of Wang et al. [47] where the derivation
2.2. Formulation of the Phase Field Model

is carried out in full detail, and to lesser extent on the more general work of Penrose and Fife [37]. For simplicity the isotropic case will be discussed here. The same approach can be used also for the anisotropic case [33], as will be discussed in Section 2.2.2.

We begin by postulating the existence of a phase field, which is an approximation of a non-conserved indicator function such that

$$
\phi(x, t) = \begin{cases} 
0 & x \in \Omega_-(t) \\
1 & x \in \Omega_+(t) 
\end{cases}.
$$

(2.2)

Furthermore, it is assumed that \(\phi(x, t)\) varies smoothly between 0 and 1 only in regions of width \(O(e)\) corresponding to diffuse solid-liquid interfaces. The small positive parameter \(e\) will later be related to physical quantities. To derive evolution equations for phase field and temperature, we assume that the entropy and internal energy of any given subvolume \(V \subset \Omega\) can be written

$$
S = \int_V \left[ s(e, \phi) - \frac{\delta^2}{2} (\nabla \phi)^2 \right] dV
$$

(2.3)

$$
E = \int_V e(T, \phi) dV
$$

(2.4)

where \(s\) is an entropy density and \(e\) an internal energy density. The second term in the integrand of (2.3) is a gradient entropy term needed to account for contributions from the solid-liquid interface, analogous to the Landau-Ginzburg energy term in the free energy [37]. According to the first and second laws of thermodynamics, we assume that \(T\) and \(\phi\) evolves in such a way that the energy is conserved and the local entropy production in \(V\) is non-negative. The second requirement is met if

$$
\tau \partial_t \phi = \frac{\delta S}{\delta \phi}.
$$

(2.5)

where \(\tau > 0\) can be interpreted as the relaxation time of the phase field [37, 47].

The key assumptions needed for deriving the phase field model are the existence of a phase field variable defined by (2.2) evolving according to (2.5), and a system entropy on the form (2.3). All other assumptions below can either be generalized or follow from standard physical modeling [37].

To formulate a set of partial differential equations the internal energy density is written

$$
e(T) = e_-(T) + p(\phi)L(T) = e_+(T) + (p(\phi) - 1)L(T),
$$

(2.6)

where \(e_\pm(T)\) are the classical internal energy densities in liquid and solid, and \(L(T) := e_-(T) - e_+(T)\) is their difference. \(p(\phi)\) is an unknown function to be specified below, required to satisfy the normalization \(p(0) = 0\) and \(p(1) = 1\). It will be assumed that the internal energy density in the liquid phase is linearly dependent on the temperature.

To identify the functional derivative in (2.5) the internal energy is related to the Helmholtz free energy through thermodynamic relations [37],

$$
J(T, \phi) = T \left( - \int_{T_M}^T \frac{e_+(\zeta)}{\zeta^2} d\zeta - (p(\phi) - 1) Q(T) + G(\phi) \right).
$$

(2.7)

where

$$
Q(T) = \int_{T_M}^T \frac{L(\zeta)}{\zeta^2} d\zeta.
$$

(2.8)
and $G(\phi)$ is another function to be specified. We require the free energy to be continuous with respect to temperature at the melting point $T_M$, and to have local minima in the solid and liquid phases. Thus we have the additional constraints

$$
G(0) = G(1) \quad \left[ G'(\phi) - \mu'(\phi) Q(T) \right]_{\phi=0,1} = 0 \quad \left[ G''(\phi) - \mu''(\phi) Q(T) \right]_{\phi=0,1} > 0
$$

(2.9)

Based on the assumptions above the dynamic equations become

$$
\begin{align*}
\tau \partial_t \phi &= \nabla \cdot \left[ \frac{\delta}{\delta \phi} p(\phi) - \tau \nabla \phi \right] + \delta^2 \nabla^2 \phi, \\
\partial_t \left( T + \frac{\Delta}{\beta} \frac{p(\phi)}{T} - 1 \right) &= D \nabla^2 T.
\end{align*}
$$

(2.10)

To fully specify a model expressions for $p(\phi)$ and $G(\phi)$ must be given. Following [47, Model I] we choose $G(\phi)$ to be a symmetric double well potential

$$
G(\phi) = \frac{1}{4a} \phi^2 (1 - \phi)^2 = \frac{a}{4a} g(\phi),
$$

(2.11)

where $a$ is the inverse depth of the potential, and

$$
p(\phi) = \int_0^\phi \frac{g(\zeta)}{g(\zeta)} d\zeta = \phi^3 (6\phi^4 - 15\phi + 10).
$$

(2.12)

These functions satisfy the necessary normalizations, and (2.9) for all temperatures. Hence the free energy (2.7) has local minima at $\phi = 0, 1$ for all values of $T$, corresponding to one of the solid or liquid phases being stable, and the other phase metastable, regardless of the supercooling of the liquid or superheating of the solid. This choice of $p$ has the advantage that the constant values taken in the bulk phases are given by (2.2) regardless of $T$ [37], but is not physically realistic for $T$ far from melting temperature.

We note that by letting the latent heat be a function of $T$ in (2.6) the heat capacities in the solid and liquid phases are allowed to depend differently on temperature. For simplicity it will now be assumed that the latent heat is constant, and that $L(T) = L(T_M) = L_0$, which corresponds to equal constant heat capacities $c_s = c_l = c$. Equation (2.8) becomes

$$
Q(T) = \frac{T_0}{T_M} T_{M} \left[ \frac{T - T_M}{T_M} + \mathcal{O} \left( \left( \frac{T - T_M}{T_M} \right)^2 \right) \right],
$$

(2.13)

after Taylor expansion around $T = T_M$. If $|T - T_M| \ll T_M$, (2.10) together with (2.13) gives the linearized approximation

$$
\begin{align*}
\tau \partial_t \phi &= \frac{\Delta_p}{c_p} (T - T_M) \phi'(\phi) - \frac{\Delta_p}{c_p} \phi'(\phi) + \delta^2 \nabla^2 \phi, \\
\partial_t \left( T + \frac{\Delta_p}{c_p} p(\phi) \right) &= D \nabla^2 T.
\end{align*}
$$

(2.14)

The linearization is relevant only for $T$ close to $T_M$, and it is therefore not a limitation to have local minima of the free energy (2.7) for all $T$.

The derivation outlined here ensures that the entropy of the system is monotonically increasing in time, resulting in a “thermodynamically consistent” model in the sense that the second law of thermodynamics holds. It is also common to derive phase field models by requiring a decreasing free energy, rather than an increasing entropy [8, 32]. The resulting partial differential equations are similar to (2.14), but do not a priori guarantee a positive
2.2. Formulation of the Phase Field Model

entropy production. The main differences arise due to the choices of constant values taken
by \( \phi \) in the solid and liquid phases, and the choice of \( p(\phi) \) describing the generation of latent
heat. \( G(\phi) \) will remain a double-well potential.

Since non-isothermal situations are considered it is more appropriate to work with en-
tropy than free energy from a physical point of view \[37\]. It is however generally agreed
that the phase field model takes on qualitative meaning only in the limit where the width
of the diffuse interfaces tend to zero, and the phase field equations approach the Stefan
problem (2.1). It is not clear from numerical simulation whether the thermodynamically
consistent models actually give better results than models based on derivation from free
energy \[28\].

2.2.1 Non-dimensionalization

To reduce the number of parameters the linearized phase field model (2.14) is rewritten in
dimensionless variables. For this purpose the dimensionless temperature

\[
    u = \frac{T - T_M}{T_M - T_0}
\]

is introduced, where \( T_0 \) is some reference temperature such as the initial temperature on
the boundaries. Space and time are scaled by

\[
    x = \frac{x_{dim}}{w}, \quad t = \frac{t_{dim}}{w^2/D}, \tag{2.15}
\]

where the subscript \( dim \) denotes a dimensional quantity, \( w \) is a reference length scale, and
\( w^2/D \) is the thermal diffusion time. Introducing the dimensionless variables

\[
    \epsilon = \frac{\sqrt{\rho_b}}{w}, \tag{2.16}
\]

\[
    m = \frac{\sigma_0^2}{D_T}, \tag{2.17}
\]

\[
    \alpha = \frac{\sqrt{\omega L_0^2}}{\delta c T_0^2}, \tag{2.18}
\]

\[
    \Delta = \frac{\epsilon(T_m - T_0)}{L_0}, \tag{2.19}
\]

(2.14) can be rewritten as

\[
    \frac{\epsilon^2}{m} \partial_t \phi = \epsilon \Delta u p'(\phi) - \frac{1}{4} g'(\phi) + \epsilon^2 \nabla^2 \phi, \tag{2.20}
\]

\[
    \partial_t \left( u + \frac{1}{\Delta} p(\phi) \right) = \nabla^2 u. \tag{2.21}
\]

\( \Delta \) is the dimensionless undercooling (or inverse Stefan number) which typically lies between
0 and 1.

It should be noted that the parameters \( \alpha, \delta \) and \( \tau \) in \(2.16 - 2.18\) are introduced in the
phase field modeling, and can not be measured through physical experiments. By a formal
asymptotic analysis performed in Section 2.3 these parameters can be related to the surface
tension and interface mobility in the sharp interface formulation \(2.1\). It will also follow
that the dimensionless width of the diffuse solid-liquid interfaces is $O(\epsilon)$, with $\epsilon$ given by (2.16). Since this is also an estimate of the grid resolution necessary to resolve the interfaces, the last relation needed to determine the three unknown physical parameters comes from the choice of $\epsilon$, which will be limited below by available computational resources, and above by accuracy requirements.

The scalings used here are slightly different from the ones chosen in [47]. Wang et al. first relates $a$ and $\delta$ to the physically measurable quantities interface width and surface tension by considering equilibrium conditions, which uniquely identifies all parameters in (2.16)–(2.19) except the relaxation time of the phase field, $\tau$. A second asymptotic analysis is then performed to express $\tau$ in the interfacial mobility $\mu$. It turns out that the analysis in Section 2.3 gives the same results to zero-order in $\epsilon$, assuming that all parameters except $\epsilon$ is of order one.

For future reference we note that (2.1) can be rewritten using the dimensionless scalings introduced above, and take the form

\[
\begin{align*}
\partial_t u = \nabla^2 u & \quad x \in \Omega_+ (t) \\
\partial_n u = -\frac{\epsilon_n}{\epsilon} & \quad x \in \Gamma (t) \\
u = -\frac{\Delta (u + \Delta)}{\epsilon} \mu - \frac{\Delta (u + \Delta)}{\epsilon} \nu_n & \quad x \in \Gamma (t),
\end{align*}
\]

(2.22)

where the quantities $\epsilon_n$ and $\mu$ are now dimensionless. The physical quantities appearing in the Gibbs-Thomson condition can be related to (2.16)–(2.19) once the asymptotic analysis has been performed.

### 2.2.2 Anisotropy

For many materials, including metals, the surface energy of the solid-melt interface and the kinetic coefficient depend on orientation. The anisotropy has a crucial impact on the shape of the dendrites, as will be seen in Section 5.3.2, and the phase field model needs to be modified in order to account for these effects. In [35] it is shown how to derive a phase field model incorporating the effects of anisotropy in two space dimensions using the approach summarized in Section 2.2. Using a formal asymptotic analysis to first order in $\epsilon$ the parameters of the phase field model can be related to those of the sharp interface formulation [35].

By letting the parameters $\tau$ and $\delta$ from (2.10) depend on $\theta$ (c.f. Figure 2.1) a model can be formulated which is equivalent to the Stefan problem (2.1) in the limit $\epsilon \to 0$. The modified version of (2.10) becomes

\[
\begin{align*}
\tau (\theta) \partial_t \phi = G (T) \phi & - G^\prime (\phi) + \overline{\mathbf{a}} \nabla \theta^2 \phi, \\
\partial_t (T + \frac{1}{\omega} (p (\phi) - 1) L (T)) = D \nabla T, \n\end{align*}
\]

where the ordinary Laplacian of the first equation has been replaced by

\[
\nabla^2 \theta (\theta) = -\frac{\partial}{\partial x} \left( \eta (\theta) \frac{\partial \theta}{\partial x} \right) + \frac{\partial}{\partial y} \left( \eta (\theta) \frac{\partial \theta}{\partial y} \right) + \nabla (\eta^2 (\theta) \nabla). \tag{2.23}
\]

Here $\eta (\theta)$ is a function chosen to model the angular dependence of the coefficient in front of the Landau-Ginzburg term in the entropy functional (2.3), which typically is chosen as

\[
\delta (\theta) \equiv \delta \eta (\theta) = \delta [1 + \gamma \cos (k \theta)], \tag{2.24}
\]

$\gamma$ is the relative strength of the anisotropy, the integer $k$ a mode number determining the number of symmetry axes, and $\delta$ a constant governing the size of the gradient contributions. The notation has been taken from [51].
2.3. A Formal Asymptotic Analysis

To determine the angle between the normal to the interface and the reference direction, we first identify the interface with level sets of the phase field, e.g. \( \Gamma(t) = \{ \mathbf{x} \in \Omega : \phi(\mathbf{x}, t) = 1/2 \} \). Thus the unit normal can be written

\[
\hat{n} = \frac{\nabla \phi}{\| \nabla \phi \|_2} = \cos(\theta) \hat{x} + \sin(\theta) \hat{y},
\]

where \( \theta \) is the angle between \( \hat{n} \) and the \( x \)-axis. If the angle between the reference direction and the \( x \)-axis is \( \theta_0 \), it follows that

\[
\theta = \theta - \theta_0 = \tan^{-1} \left( \frac{\partial_y \phi}{\partial_x \phi} \right) - \theta_0. \tag{2.25}
\]

From this expression it is clear that the modified Laplacian (2.23) is non-linear in \( \phi \), which is significant in the formulation of numerical schemes.

\( \nabla^2 \phi(\theta) \) is zero when \( \phi \) is constant just like for the ordinary Laplacian. Therefore the angular dependence will enter only in the vicinity of the interface. Since the angle is determined from the phase field variable the actual geometry of the interface does not have to be computed.

For the simulations, it has been assumed that \( \tau \) satisfies a relation similar to (2.24), \( \tau(\theta) = \tau(\theta) \), which means that the dimensionless parameter \( m \) of (2.20) is independent of \( \theta \). Thus the only modification of (2.20) is that the Laplacian should be replaced by (2.23).

There will be an additional contribution to the anisotropy from the discretization, not accounted for in the modeling above. Effects of grid anisotropy are investigated numerically in Section 5.3.2.

2.3 A Formal Asymptotic Analysis

In this section a matched formal asymptotic analysis of the dimensionless phase field model (2.20) (2.21) is performed in the limit \( \epsilon \to 0 \). There are two objectives with the analysis: to verify that the solutions of (2.20) and (2.21) approach the solutions of (2.1) with errors in the interface conditions of magnitude \( O(\epsilon^2) \), and to relate (2.16) (2.18) to physically measurable quantities. The analysis is based on ideas presented in [2] and [14] where more general physical settings are considered.

The analysis is performed in one space dimension to avoid some technicalities associated with orthogonal curvilinear coordinate systems, but can be generalized to two [2, 14] or three [22, 23] dimensions. Effects of crystalline anisotropy can be included [14, 22, 23, 35] but are not present here since the one-dimensional case is considered.

In the early asymptotic analysis approximate solutions of (2.20) and (2.21) were determined under assumption of constant temperature in the interfacial region \( 0 < \phi < 1 \) [8, 9, 10, 47]. As observed by Karma and Rappel [22, 23] this implies that the \( \epsilon \) must be much smaller than the dimensionless capillary length, as defined by

\[
d_0 = \frac{c_0 T_M}{\nu L_0^2} \tag{2.26}
\]

The capillary length is a measure of the smallest geometric scale in the problem since the dendrite tip radius is given by \( d_0 / f(\Delta) \), where \( f(\Delta) \) increases sharply with \( \Delta \) [23]. The constraint \( \epsilon \ll d_0 \) can therefore be interpreted as the grid resolution having to be much smaller than the geometry we wish to resolve.
By letting the temperature vary slowly within the interface the constraint can be removed, thus allowing larger ratios of $\epsilon$ to grid spacing. This is especially useful at small undercoolings where the dendrite tip radius is large. There will also be an $O(\epsilon^2)$ correction to the dimensionless mobility $m$ which allows simulation with $\mu \to \infty$. For certain materials, such as succinonitrile, this is a relevant limit.

### 2.3.1 Asymptotic Ansatz

The case of a crystal growing in one dimension under assumption of radial symmetry is considered. It is assumed that all dimensionless parameters except $\epsilon$ in the dimensionless phase field model (2.20), (2.21) are $O(1)$ as $\epsilon \to 0$, which corresponds to $a \sim \delta^2$ and $\tau \sim \delta^2$ in (2.14). Other scaling limits gives convergence of the phase field model to different sharp interface models, such as the Hales-Shaw and Cahn-Allen models [9, 14].

It will be convenient to work with the moving coordinate system $r(x, t; \epsilon) = x - \Gamma(t; \epsilon)$ attached to the interface, given implicitly by $\Gamma(t; \epsilon) := \{x \in \Omega : \phi(x, t; \epsilon) = 1/2\}$. Noting that $r$ is a signed distance function we can identify negative values with solid, and positive values with liquid, for $|r| \gg O(\epsilon)$. To go from Cartesian coordinates to the moving coordinate system the derivatives are transformed according to

$$
\nabla^2 \to \partial_{rr} + \mathcal{K}(t; \epsilon) \partial_r, \\
\partial_t \to \partial_t - (\partial \Gamma(t; \epsilon)) \partial_r,
$$

where $\mathcal{K}(t; \epsilon)$ denotes the mean curvature. The phase field model is a set of singularly perturbed reaction diffusion equations and thus one can expect layered solutions. The stretched coordinate system $\rho = r/\epsilon$ is used in the inner region $r = O(\epsilon)$, and the transformation corresponding to (2.27) becomes

$$
\nabla^2 \to \epsilon^{-2} \partial_{\rho\rho} + \epsilon^{-1} \mathcal{K}(t; \epsilon) \partial_{\rho}, \\
\partial_t \to \partial_t - \epsilon^{-1} (\partial \Gamma(t; \epsilon)) \partial_{\rho}.
$$

Assuming that solutions to the phase field equations approach their asymptotic limits smoothly as $\epsilon \to 0$ we can represent the inner solutions by power series expansions,

$$
\phi(\rho, t; \epsilon) = \sum_{j=0}^{\infty} \epsilon^j \phi_j(\rho, t; \epsilon), \\
u(\rho, t; \epsilon) = \sum_{j=0}^{\infty} \epsilon^j u_j(\rho, t; \epsilon).
$$

A similar ansatz is used for the outer solutions with dependence on $r$ instead of $\rho$, and using superscripts $-$ and $+$ to denote solid and liquid respectively. If the asymptotic ansatz holds, the interfacial properties satisfy

$$
\Gamma(t; \epsilon) = \sum_{j=0}^{\infty} \epsilon^j \Gamma_j(t; \epsilon), \\
\mathcal{K}(t; \epsilon) = \sum_{j=0}^{\infty} \epsilon^j \mathcal{K}_j(t; \epsilon).
$$

Derivatives of $\Gamma(t; \epsilon)$ are needed in (2.27) and (2.28) so we introduce the propagation speed

$$
v_j(t; \epsilon) := \partial_t \Gamma_j(t; \epsilon)
$$

for notational convenience.
2.3. A Formal Asymptotic Analysis

The inner and outer solutions should agree in the common domain of validity, $\varepsilon \ll |r| \ll 1$, leading to the matching conditions

$$
\lim_{\rho \to 0^+} u_k(\rho, t; \varepsilon) = \sum_{j=0}^{k} \frac{1}{\varepsilon^j} \partial^j u_{k-j}(0, t; \varepsilon) \rho^j,
$$

$$
\lim_{\rho \to 0^+} \phi_k(\rho, t; \varepsilon) = \begin{cases} 
1/2 & \text{if } k = 0 \\
0 & \text{otherwise}.
\end{cases}
$$

(2.30)

The first matching condition is obtained by Taylor expanding the outer solutions to the left and right of $\Gamma$ and identifying powers of $\varepsilon$, and holds if the $u_j$ are sufficiently smooth which in the analysis below means at least three times differentiable at $r = 0$ as $\varepsilon \to 0$. The second condition ensures that the phase field is constant and equal to 0 or 1 away from the interface. In addition to (2.30) it follows from the definition of $\Gamma(t; \varepsilon)$ that

$$
\phi_k(0, t; \varepsilon) = \begin{cases} 
1/2 & \text{if } k = 0 \\
0 & \text{otherwise}.
\end{cases}
$$

(2.31)

Boundary conditions should be prescribed at the boundaries of the domain in order to determine the outer solutions. In the present context this is not necessary since the assumed existence of outer solutions is sufficient to determine the interfacial conditions. The analysis is heuristic in the sense that the validity of (2.29) is not proved. A rigorous asymptotic analysis has been performed by Caginalp and Chen under assumptions of radial symmetry [10].

Inserting the asymptotic ansatz (2.29) in (2.20), and applying the change of coordinates by (2.28) we obtain

$$
0 = \partial^0 \left[ \partial_{\rho} \phi_0 - \frac{1}{4} g'(\phi_0) \right] + e^1 \left[ \partial_{\rho} \phi_1 - \frac{1}{4} g''(\phi_0) \phi_1 + \alpha \Delta u_0 p'(\phi_0) + \left( \frac{\tau_0}{m} + K_0 \right) \partial_{\rho} \phi_0 \right] + \frac{\tau_1}{m} \partial_{\rho} \phi_1 + \left( \frac{\tau_1}{m} + K_1 \right) \partial_{\rho} \phi_2 - \frac{1}{m} \partial_{\phi} \phi_0 + O(\varepsilon^2),
$$

(2.32)

and similarly by inserting (2.29) in (2.21),

$$
0 = \partial^0 \left[ \partial_{\rho} u_0 \right] + e^1 \left[ \partial_{\rho} u_1 + \left( \tau_0 + K_0 \right) \partial_{\rho} u_0 + \frac{\tau_0}{8} \partial' \phi_0 \partial_{\rho} \phi_0 \right] + e^2 \left[ \partial_{\rho} u_2 + \left( \tau_0 + K_0 \right) \partial_{\rho} u_1 + \left( \tau_1 + K_1 \right) \partial_{\rho} u_0 + \frac{\tau_0}{8} \partial' \phi_0 \partial_{\rho} \phi_1 + \frac{\tau_1}{4} \partial' \phi_0 \partial_{\rho} \phi_0 + \frac{\tau_0}{8} \partial''(\phi_0) \partial_{\rho} \phi_0 - \partial_{t} u_0 \right] + O(\varepsilon^3).
$$

(2.33)

The unknown functions in the asymptotic expansion can be determined by identifying powers of $\varepsilon$ in (2.32) and (2.33) and solving the corresponding equations for $\phi_j$ and $u_j$. For brevity of notation we will refer to the polynomials $p(\phi)$ and $g(\phi)$, and their derivatives below instead of the explicit expressions given by (2.11) and (2.12). All integrals have however been evaluated for these specific choices. The constants of integration which appear are denoted by $c_\ast$ and $C_\ast$ for the temperature and phase field respectively, and numbered according to their introduction. Explicit expressions will be given only when needed.
2.3.2 Zero-order Inner Solution

The equation for \( u_0 \) is a linear ordinary differential equation in \( \rho \), and can be integrated twice to obtain

\[
u_0(\rho, t; \epsilon) = c_0(t) + c_1(t)\rho \tag{2.34}\]

From (2.30) we expect the temperature to be bounded as \( \rho \to \pm \infty \) and thus \( c_1(t) = 0 \). The constant \( c_0(t) \) is determined below.

Using the chain rule of differentiation we note that

\[
\partial_{\rho^2} \phi_0 = \frac{1}{2} \frac{d}{d\rho} \left( \partial_{\rho} \phi_0 \right)^2.
\]

Inserting this expression in the leading-order equation from (2.32) and integrating with respect to \( \phi_0 \) yields

\[
(\partial_{\rho} \phi_0)^2 = \frac{1}{2} \rho(\phi_0) + C_0(t). \tag{2.35}
\]

According to (2.30) the constant of integration must be zero, and the equation is separable with solution

\[
\phi_0(\rho, t; \epsilon) = \frac{C_1(t)}{1 + e^{-\rho/\sqrt{2}}} + \frac{C_2(t)}{1 + e^{\rho/\sqrt{2}}},
\]

where the two different terms are the result of taking the square root of (2.35). Applying the matching conditions again, and enforcing \( \phi_0(0, t; \epsilon) = 1/2 \) the zero-order inner solution can be written

\[
\phi_0(\rho, t; \epsilon) = \frac{1}{1 + e^{-\rho/\sqrt{2}}} = \frac{1}{2} \left( \tanh \frac{\rho}{2\sqrt{2}} + 1 \right). \tag{2.36}
\]

Figure 2.2 shows \( \phi_0 \) in the interval \(-20 \leq \rho \leq 20\).

2.3.3 First Order Inner Solution

From (2.34) we know that the leading order term in the temperature expansion does not depend on \( \rho \), and the \( \mathcal{O}(\epsilon) \)-equation of (2.33) can be written

\[
\partial_{\rho} \left( \partial_{\rho} u_1 + \frac{u_0}{\Delta} \rho(\phi_0) \right) = 0.
\]

Integrating twice with respect to \( \rho \) yields

\[
u_1(\rho, t; \epsilon) = c_2(t) + c_3(t)\rho - \frac{u_0}{\Delta} \int_0^\rho \rho(\phi_0(z))dz = c_2(t) + c_3(t)\rho
\]

\[
- \frac{\sqrt{2\pi} u_0}{\Delta} \left( \frac{11}{32} (12\phi_0(1 - \phi_0) - 11)(2\phi_0 - 1)^2 - \ln (2 - 2\phi_0) \right).
\]

Expressing the evaluated integral in \( \phi_0 \) facilitates identification of limits as \( \rho \to \pm \infty \). We obtain

\[
\lim_{\rho \to -\infty} u_1(\rho, t; \epsilon) \sim c_2(t) + c_3(t)\rho - \frac{\sqrt{2\pi} u_0}{\Delta} \left( \frac{11}{32} - \ln 2 \right)
\]

\[
\lim_{\rho \to \infty} u_1(\rho, t; \epsilon) \sim c_2(t) + c_3(t)\rho - \frac{\sqrt{2\pi} u_0}{\Delta} \left( \frac{11}{32} - \ln 2 + \frac{\rho}{\sqrt{2}} \right).
\]
2.3. A Formal Asymptotic Analysis

which after comparison with (2.30) gives

\[
\begin{align*}
\partial_r u_0^-(0) &= c_3(t), \\
\partial_r u_0^+(0) &= c_3(t) - \frac{\nu}{\Delta}, \\
\phi_1^+(0) &= c_2(t) - \left(\frac{11}{32} - \ln 2\right) \frac{\sqrt{2\nu}}{\Delta}.
\end{align*}
\]

(2.38)

Thus the temperature is continuous across the interface to zero-order, and the normal derivative satisfies a jump condition according to (2.22). A plot of the integral from (2.37) is shown in Figure 2.4.

The \(O(\varepsilon)\)-equation of (2.32) is

\[
\alpha \Delta u_0 \phi' (\phi_0) + \left(\frac{\nu}{m} + K_0\right) \partial_\rho \phi_0 = - \left(\partial_\rho - \frac{1}{4} \phi' (\phi_0)\right) \phi_1 =: \mathcal{P} \phi_1,
\]

(2.39)

where the differential operator \(\mathcal{P}\) will also appear in the equation for \(\phi_2 (\rho, t; \varepsilon)\). Note that \(\partial_\rho \phi_0\) is a null-function to \(\mathcal{P}\),

\[
\mathcal{P} \partial_\rho \phi_0 = - \partial_\rho \left(\partial_\rho \phi_0 - \frac{1}{4} \phi' (\phi_0)\right) = 0,
\]

because the expression in the parenthesis is exactly the equation we solved to get the zero-order approximation \(\phi_0 (\rho, t; \varepsilon)\). The left-hand side of (2.39) must be orthogonal to \(\partial_\rho \phi_0\) in order for a solution to exist, which leads to the solvability condition

\[
0 = \int_{-\infty}^{\infty} \left[\alpha \Delta u_0 \phi' (\phi_0) + \left(\frac{\nu}{m} + K_0\right) \partial_\rho \phi_0\right] (\partial_\rho \phi_0) d\rho
\]

\[
= \alpha \Delta u_0 + \frac{1}{6 \sqrt{2}} \left(\frac{\nu}{m} + K_0\right),
\]

(2.40)

from which the unknown constant in (2.34) can be determined,

\[
u_0 (\rho, t; \varepsilon) = \nu_0 (t; \varepsilon) = - \frac{1}{6 \sqrt{2} \alpha \Delta} \left(\frac{\nu_0 (t; \varepsilon)}{m} + K_0 (t; \varepsilon)\right).
\]

(2.41)

A comparison to (2.22) at this point would allow us to express the dimensionless parameters in known physical constants [47]. We are interested in pushing the error in the interface conditions to higher order in \(\varepsilon\) and will therefore continue with the asymptotics.

To solve (2.39), \(\phi_1\) is rewritten as

\[
\phi_1 (\rho, t; \varepsilon) = \zeta_1 (\rho, t; \varepsilon) \partial_\rho \phi_0 (\rho, t; \varepsilon)
\]

(2.42)

where \(\zeta_1\) is an unknown function to be determined below. It turns out that \(\zeta_1\) can be found by solving an ordinary differential equation of the first order in \(\zeta_1\), which is simpler than solving the second order equation (2.39) directly. Inserting (2.42) into (2.39) gives

\[
- \partial_\rho \left((\partial_\rho \zeta_1)^2 \cdot \partial_\rho \zeta_1\right) = \alpha \Delta u_0 \phi' (\phi_0) \partial_\rho \phi_0 + \left(\frac{\nu}{m} + K_0\right) \left(\partial_\rho \phi_0\right)^2,
\]

This expression can be integrated once to obtain a first order equation in \(\zeta_1\),

\[
\partial_\rho \zeta_1 = - \frac{1}{(\partial_\rho \phi_0)^2} \left[ C_3 (t) + \alpha \Delta u_0 \phi (\phi_0) + \frac{1}{6 \sqrt{2}} \left(\frac{\nu}{m} + K_0\right) (3 \phi_0^2 - 2 \phi_0^3)\right]
\]
Noting that $\partial_p \phi_0 = \phi_0(1 - \phi_0)/\sqrt{\pi}$, and that some of the physical constants can be eliminated by (2.41), the expression simplifies to

$$\partial_p \zeta_1 = -\frac{2C_3(t)}{\phi_0(1 - \phi_0)^2} + 6\alpha \Delta u_0(1 - 2\phi_0),$$

and after integration

$$\zeta_1 = C_4(t) - 6\alpha \Delta u_0 \left(2\sqrt{2} \ln(\phi_0^{-1}) + \rho\right) - 2C_3(t) \left(6\rho + \sqrt{2} \left(\sinh(\sqrt{2}\rho) + 8\sinh(\rho/\sqrt{2})\right)\right),$$

Inserting this expression in (2.42), and investigating the limiting values of $\phi_1$ as $\rho \to \pm \infty$, one finds that $\phi_1$ is bounded if and only if $C_3(t) = 0$. The coefficient $C_4(t)$ can be determined by (2.31), and we arrive at

$$\phi_1(\rho, t, \epsilon) = \alpha \Delta u_0 \phi_0^2 \left(12 \ln(2\phi_0) - 3\sqrt{\pi}\right) e^{-\rho^2/\sqrt{\pi}}. \quad (2.43)$$

The function is plotted in Figure 2.3.

### 2.3.4 Second Order Inner Solution

The analysis in this section will not be carried through all the way, and explicit expressions for $\phi_2$ and $u_2$ will not be determined. The objective is to find interfacial relations such that the errors can be made of size $\mathcal{O}(\epsilon^2)$. From the analysis in Section 2.3.3 we expect to get information on the jump condition from matching conditions for the temperature, and information about the interfacial temperature from a solvability condition similar to (2.40).

Integrating the $\mathcal{O}(\epsilon^2)$-term of equation (2.33) once with respect to $\rho$ we obtain

$$\partial_\rho \mu_2 + (\nu_0 + K_0)u_1 + \frac{\mu_0}{\Delta} j'(\phi_0) \phi_1 + \frac{\nu_1}{\Delta} p(\phi_0) - \partial_\rho u_0 \cdot \rho + c_4(t) = 0. \quad (2.44)$$

A matching condition similar to (2.30) can be formulated by Taylor-expanding the first derivatives of the outer temperature to the left and right of the interface,

$$\lim_{\rho \to \pm \infty} \partial_\rho \mu_2 = \partial_\rho u_1^+(0) + \partial_\rho u_0^+(0)\rho. \quad (2.45)$$
which after insertion of (2.36), (2.37) and (2.43) in (2.44) gives,

\[
\partial_\tau u^+_1(0) = -(t_0 + K_0) \left[ c_2(t) - \frac{\sqrt{2} v_0}{\Delta} \left( \frac{11}{32} - \ln 2 \right) \right] - c_4(t) - \frac{v_1}{\Delta};
\]

\[
\partial_\tau u^-_1(0) = -(t_0 + K_0) \left[ c_2(t) - \frac{\sqrt{2} v_0}{\Delta} \left( \frac{11}{32} - \ln 2 \right) \right] - c_4(t). \tag{2.46}
\]

By using the matching condition (2.45) instead of (2.30) some difficulties in integrating (2.44) can be avoided. We lose information on \( u^+_2(0, t; \epsilon) \), but apart from this the same results are obtained.

The operator \( \mathcal{P} \) appearing in (2.39) is present also in the equation for \( \phi_2 \). By the same argument as above we require the null-function \( \partial_\tau \phi_0 \) to be orthogonal to the right hand side. Hence we have a solvability condition,

\[
0 = \alpha \left[ \Delta c_2(t) - \sqrt{2} v_0 \left( \frac{2827}{3360} - \ln 2 \right) + \frac{6}{\sqrt{2} \Delta} \left( \frac{v_1}{m} + K_1 \right) \right].
\]

Solving this equation for \( c_2(t) \) we obtain

\[
c_2(t) = \left( \frac{2827}{3360} - \ln 2 \right) \frac{\sqrt{2} v_0}{\Delta} - \frac{1}{6 \sqrt{2} \Delta} \left( \frac{v_1}{m} + K_1 \right), \tag{2.47}
\]

and the temperature at the interface (2.38) has been determined to first order in \( \epsilon \).
2.3.5 Interface Conditions

Assuming that the asymptotic ansatz holds, the interface conditions for the outer solutions can be expressed using (2.38), (2.46) and (2.47),

\[
\begin{align*}
\partial_
u u^+ \bigg|_- &= -\frac{v_0 + \varepsilon v_1}{\Delta} + O(\varepsilon^2), \\
u^+(0) &= -\frac{1}{6\sqrt{2} \Delta \alpha} \left( \frac{v_0 + \varepsilon v_1}{m} + (K_0 + \varepsilon K_1) \right) + \frac{209 \sqrt{2} v_0}{420 \Delta} + O(\varepsilon^2). 
\end{align*}
\] (2.48)

A comparison to the dimensionless Stefan problem (2.22) shows that the jump condition is satisfied with errors of size \( \varepsilon^2 \). \( \phi \) is constant away from the interface and \( p(\phi) \) takes the constant values 0 (solid) or 1 (liquid). Thus the term depending on \( \phi \) disappears from (2.21), and the ordinary heat equation is satisfied in the bulk phases in agreement with (2.22).

It remains to determine the parameters \( m \) and \( \alpha \) such that (2.48) is equivalent to the dimensionless Gibbs-Thomson condition. To this end it will be necessary to compute the dimensional surface tension, given at melting temperature \( T_M \) by [47]

\[
\sigma = T_M \int_{-\infty}^{\infty} \delta^2 \left( \partial_{x \text{ dim}} \phi(x, \text{ dim}, t) \right)^2 dx \text{ dim}
\]

\[
= \frac{\sigma}{\alpha} \int_{-\infty}^{\infty} \left[ \varepsilon\left( \partial_\rho \phi_0 \right)^2 + 2\varepsilon^2 \partial_\rho \phi_0 \partial_\rho \phi_1 + O(\varepsilon^3) \right] \rho d\rho,
\]

where the final expression is obtained from substitution using (2.15) and (2.16), and insertion of the asymptotic ansatz (2.32). Noting that the \( \varepsilon^3 \)-term in the integrand is odd, the integral can easily be evaluated,

\[
\sigma = \frac{\sigma}{\alpha} \frac{\delta T_M}{6 \sqrt{2} \alpha} + O(\varepsilon^3) = \frac{1}{6 \sqrt{2} \Delta \alpha} \frac{\delta T_M}{\alpha} + O(\varepsilon^3),
\] (2.49)

where regularity assumptions from the asymptotic ansatz and matching conditions ensure that the higher order terms in the integrand grow at most by a constant. Rewriting (2.18) using (2.26) and (2.49) we find

\[
\alpha = \frac{w L_0^2}{6 \sqrt{2} \sigma T_M} = \frac{1}{6 \sqrt{2} \delta d_0},
\] (2.50)

which is an expression in known quantities. The curvature-dependent terms in (2.48) agree with (2.22) to first order in \( \varepsilon \).

To determine \( m \) we first consider the sharp interface limit, in which the \( O(\varepsilon) \)-terms in (2.48) are neglected. We obtain

\[
m_0 = \frac{\mu T_M}{L_0 D}
\] (2.51)

after comparison with (2.22). This expression was also obtained in the early asymptotic analysis where the variation of temperature across the interface was neglected [47]. \( m \) has been given the subscript zero to distinguish it from the value used in computation.
Following observations in [2, 14, 23] we note that by letting
\[ \frac{1}{m} = \frac{1}{m_0} + \frac{209}{35} \alpha, \]  
the Gibbs-Thomson condition (2.48) becomes
\[ u^\pm(0) = -\frac{1}{6\sqrt{2}\alpha \Delta} \left( \frac{m_0 + \epsilon m}{m_0} + \left( \mathcal{K}_0 + \epsilon \mathcal{K}_1 \right) \right) + \mathcal{O}(\epsilon^2). \]
i.e. the \( \mathcal{O}(\epsilon) \)-correction in (2.52) gives agreement to first order in the Gibbs-Thomson condition for interfacial mobility \( m_0 \). Neglecting the last term in (2.52) is equivalent to assuming that \( \epsilon \alpha \sim \epsilon/\alpha \ll 1/m_0 \). By incorporating the thin interface correction a larger ratio of interface thickness to capillary length is allowed for a given mobility \( m_0 \). As will be seen in Section 5.2 this has significant implications on computational efficiency. The thin interface asymptotics also enable simulation with \( m_0 = 0 \), or equivalently \( \mu \rightarrow \infty \), by choosing
\[ \frac{1}{m} = \frac{209}{35} \epsilon \alpha, \]
which is not possible in the sharp interface limit.

The actual phase field equations do not change in the thin interface limit, merely the value of the dimensionless parameter \( m \). In the sharp interface limit \( u(0, t; \epsilon) = u^\pm(0, t; \epsilon) \) as given by (2.41). This is not true when the temperature is allowed to vary within the interface, as seen from (2.37) and (2.38). While this would seem to be an inconsistency, the aim of the phase field model is to describe the macroscopic features of solidification, and the microscopic internal structure of the interface is not necessarily represented correctly.

It is possible to continue the asymptotic analysis to higher orders in \( \epsilon \) but that would require solving partial differential equations to find \( u_3 \) and \( \phi_3 \).
Chapter 3

Numerical Approximation

This chapter treats the numerical schemes used for solving the phase field equations. The partial differential equations (PDEs) are solved using the method of lines. Via a sequence of approximations we will see that approximate solutions to the PDEs can be determined by solving linear systems.

Section 3.1 discusses discrete finite difference approximations to the spatial derivatives in one and two dimensions on uniform Cartesian grids. Inserting these approximations into the PDEs a system of ordinary differential equations (ODEs) for the grid values is obtained. The ODEs can then be solved using standard numerical methods [19]. Two specific choices are discussed in Section 3.2. Schemes based on equation splitting are the most widely used [25, 51] and are typically formulated so that no non-linear systems need to be solved, see Section 3.2.1. Due to simple geometry and discretization this can be done efficiently using a Fast Poisson solver [33] as described in Section 3.4.

A more accurate scheme based on backward difference formulas is proposed in Section 3.2.2. It is based on implicit discretization of the ODEs and results in a large sparse non-linear system every time step, which is solved by a backtracking line search modification of Newton’s method [13]. The arising linear systems are solved iteratively using the Generalized Minimal Residual Method (GMRES) [40, 42], making the non-linear solver a Newton-Krylov algorithm [6]. The non-linear solver is discussed in greater detail in Section 3.3.

3.1 Spatial Discretization

3.1.1 One Space Dimension

We begin by considering the spherically symmetric case. This problem is to simple to be of interest from a physical point of view since the interface is given by a single point and not associated with a geometry. The curvature, and hence also the contribution from capillary effects, is prescribed by the position of the solid-liquid interface. Nevertheless the problem is useful for investigation of numerical schemes. The computer implementation is simpler and simulation times are shorter than for the two- and three-dimensional cases.
We wish to find solutions $\phi(x, t)$ and $u(x, t)$ to the dimensionless phase field equations for $(x, t) \in [x_{min}, x_{max}] \times [0, T]$, given initial values at time $t = 0$, and boundary conditions at $x = x_{min}, x_{max}$. Initial values are set to

$$\phi(x, 0) = \frac{1}{2} \left[ \tanh \left( \frac{x - r_0}{2\sqrt{2\epsilon}} \right) + 1 \right],$$

$$u(x, 0) = -1,$$  \hspace{1cm} (3.1)

which corresponds to a uniform temperature distribution equal to the undercooling, and a phase field given by the zero-order asymptotic approximation (2.36). Here $r_0$ can be interpreted as the initial radius of the crystal. Boundary conditions are discussed below.

Approximate solutions to (2.20) and (2.21) are determined by considering finite difference discretizations on uniform Cartesian grids,

$$\Omega_h^{(1)} := \{x_i = x_{min} + i \Delta x, \; i = 0, 1, \ldots, N - 1\}.$$  \hspace{1cm} (3.2)

Let $f$ be a gridfunction defined on $\Omega_h^{(1)}$ whose value in $x_i$ is denoted $f_i$. Spatial derivatives of $f$ can be approximated by the difference operators

$$D_+ f_i := \frac{f_{i+1} - f_i}{\Delta x} = f'(x_i) + O(\Delta x),$$

$$D_- f_i := \frac{f_i - f_{i-1}}{\Delta x} = f'(x_i) + O(\Delta x),$$

$$D_0 f_i := \frac{f_{i+1} - f_{i-1}}{2\Delta x} = f'(x_i) + O(\Delta x^2),$$

Higher order derivatives are approximated by repeated application of the difference operators. Special care has to be taken for $i$ close to 0 or $N - 1$ since the expressions in (3.2) may need values $f_i$ outside of $\Omega_h^{(1)}$. Such values are obtained by discretizing the boundary conditions of the PDE.

The action of the Laplacian can be approximated by

$$\frac{1}{x_i^2} D_+ \left( x_i^2 \frac{1}{2} D_- f_i \right) = \nabla^2 f_i + O(\Delta x^2).$$  \hspace{1cm} (3.3)

From the spherical symmetry it follows that $\phi$ and $u$ must satisfy zero Neumann-conditions at $x = 0$: $\phi_x(0, t) = u_x(0, t) = 0$. Zero Neumann conditions are prescribed for the phase field also at the outer boundary, whereas the temperature is required to satisfy $u_x(x_{max}) + (u(x_{max}) + 1)/x_{max} = 0$ [31]. The second order accurate discretized boundary conditions become

$$\phi_{-1} = \frac{4}{3} \phi_0 - \frac{1}{3} \phi_1, \hspace{0.5cm} \phi_N = \frac{4}{3} \phi_{N-1} - \frac{1}{3} \phi_{N-2},$$

$$u_{-1} = \frac{4}{3} u_0 - \frac{1}{3} u_1, \hspace{0.5cm} u_N = \frac{4u_{N-1} - u_{N-2} - 2h/x_{max}}{3 + 2h/x_{max}}.$$  \hspace{1cm} (3.4)

By replacing spatial derivatives in (2.20) and (2.21) using (3.3) and (3.4), a system of ODEs is obtained. This system is discussed in Section 3.1.2 below.

The one dimensional spherically symmetric phase field equations have also been discretized using fourth order accurate finite differences to investigate whether it would allow an increased ratio $\epsilon/\Delta x$, which would lead to reduced grid sizes. This is not the case. Choosing an $\epsilon$ larger than the grid spacing leads to oscillations in the interfacial velocity.
3.1.2 Two Space Dimensions

For the discretization in two space dimensions we introduce the grid

\[
\Omega_h := \{(x_i, y_j) = (x_{\min} + i \Delta x, y_{\min} + j \Delta y), \ 0 \leq i < N_x, \ 0 \leq j < N_y\},
\]

and analogous to the one dimensional case \( f_{i, j} \) denotes the value of a gridfunction defined on \( \Omega_h \) in the point \((x_i, y_j)\). Difference operators equivalent to (3.2) are used, with additional subscripts \( x \) or \( y \) to denote the approximated derivative.

The Laplacian in (2.21) can be approximated to second order using the standard five point stencil,

\[
\nabla^2 u_{i, j} \approx \left( D_{+x} D_{-x} + D_{+y} D_{-y} \right) u_{i, j}.
\]

An approximation to the anisotropic Laplacian (2.23) should be used in (2.20) to account for preferred growth directions in the crystal. We wish to minimize the width of the difference operator for two reasons: the function \( \eta(\theta) \) used to model anisotropy has significance only in the vicinity of the interface, and in a parallel implementation the amount of communication increases with the width, see Chapter 4. For this purpose approximations to certain gridfunctions will be needed between gridpoints, as denoted by non-integer indices in the formulas below. We obtain

\[
\nabla^2 \phi_{i, j} \approx -D_{-x} \left( (\eta_y')_{i+1/2, j} D_{0x} \frac{\phi_{i, j} + \phi_{i+1, j}}{2} \right) + D_{-y} \left( (\eta_x')_{i, j+1/2} D_{0y} \frac{\phi_{i, j} + \phi_{i, j+1}}{2} \right) + D_{-x} \left( \eta^2 / \phi_{i+1/2, j} D_{+x} \phi_{i, j} \right) + D_{-y} \left( \eta^2 / \phi_{i, j+1/2} D_{+y} \phi_{i, j} \right),
\]

where \( \eta(\theta) \) is defined by (2.24), and subscripts denote the gridpoint in which the function is evaluated. The angle \( \theta \), as given by (2.25), is approximated by

\[
\theta_{i+1/2, j} = \arctan \left( \frac{D_{0y}(\phi_{i, j} + \phi_{i+1, j})}{2D_{+x} \phi_{i, j}} \right) - \theta_0,
\]

\[
\theta_{i, j+1/2} = \arctan \left( \frac{2D_{+y} \phi_{i, j}}{D_{0x}(\phi_{i, j} + \phi_{i+1, j+1})} \right) - \theta_0
\]

between gridpoints. It can be shown by Taylor expansion that (3.7) - (3.8) is a second order accurate approximation to the anisotropic Laplacian. Eight neighboring gridpoints are needed to evaluate the anisotropic Laplacian which is four more than for the standard five-point stencil. Zero Neumann conditions are prescribed on all boundaries, \( \nabla \phi \cdot \hat{n} = \nabla u \cdot \hat{n} = 0 \).

From (3.8) one might expect floating point exceptions when computing \( \theta \) between certain gridpoints due to division by zero or expressions on the form \( \pi/0 \). For implementations written in C or Fortran these difficulties can be avoided by computing the arctangent using the \( \text{atan2} \)-routine instead of \( \text{atan} \), thus avoiding explicit division. When \( \phi \) is approximately constant, cancellation may result in poor accuracy in the computed angles. This occurs only away from the interface where the contribution from the Laplacian should be zero. No practical consequences have been observed in simulation, and the Newton iteration converges quadratically as expected.

The difference operator (3.7) has been expressed using \( \eta \) and \( \theta \), instead of expanding derivatives in the analytic expression (2.23) and expressing them in terms of difference
operators in $\phi$. The approach chosen here is likely less sensitive to round-off errors and
cancellation due to the use of $\tan(2)$.

In the following development of the numerical schemes it will be convenient to define a
one-to-one mapping from gridfunctions defined on (3.5) to column vectors of length $N_xN_y$.
Let such a mapping be given by

$$
\phi_{i,j} \rightarrow \Phi_{i+jN_x+1},
$$

which is sometimes referred to as natural ordering of the gridpoints [40]. With this represen-
tation the action of difference operators on gridfunctions can be interpreted as matrix-vector
products.

The phase field model can now be approximated by a system of ODEs for the grid values
of $\phi$ and $u$,

$$
\frac{d}{dt} \left[ U + p(\Phi) / \Delta \right] = \left[ \begin{array}{cc} m \nabla^2(\Phi) & 0 \\ 0 & \tilde{\nabla}^2 \end{array} \right] \left[ \Phi \right] + \left[ F(\Phi, U) \right].
$$

Terms denoted by $\tilde{\nabla}$ are matrix representations of the difference operators (3.6) - (3.8).
For brevity of notation, the polynomial terms in (2.20) have been collected in $F$ which
also depends on dimensionless parameters in the phase field model. A system on the same
form is obtained from the one dimensional discretization as well, but without the angular
dependence. Equation (3.10) is the starting point for the temporal discretization discussed
in the following section.

3.2 Temporal Discretization

To solve the system of ODEs we introduce a discrete representation of time and determine
approximate solutions at equidistant time levels \( \{n \Delta t\}_{n=1,2,...} \), given initial values at $t = 0$.
At time level $n$ the approximations for the phase field and temperature are denoted by $\Phi^n$ and $U^n$ respectively.

3.2.1 Methods Based on Equation Splitting

The most common approach is based on equation splitting, which is motivated by a wish
to avoid non-linear systems. Given solutions $\Phi^{n-1}$ and $U^{n-1}$, solutions at time level $n$ are
determined by:

1. Computing $\Phi^n$ from the upper part of (3.10) using an explicit scheme, e.g. forward
   Euler,

   $$
   \Phi^n = \Phi^{n-1} + m \Delta t \tilde{\nabla}^2(\Phi^{n-1}) \Phi^{n-1} + \Delta t F(\Phi^{n-1}, U^{n-1})
   $$

2. Computing $U^n$ from the lower part of (3.10) using an implicit scheme, e.g. Crank-
   Nicholson or backward Euler,

   $$
   U^n = \left( I - \Delta t \tilde{\nabla}^2 \right)^{-1} U^{n-1} - \frac{1}{\Delta} \left( p(\Phi^n) - p(\Phi^{n-1}) \right)
   $$

The time derivative of $p(\phi)$ is approximated using the value of $\Phi^n$ computed in step
1, and written on conservative form to ensure conservation of energy.
3.2 Temporal Discretization

Other numerical schemes can be used, but the update in step 1 should be explicit to avoid non-linear equations. We note that employing the forward Euler scheme imposes step size restrictions on $\Delta t$. Ignoring non-linear terms in (3.11), and setting the anisotropy to zero, the well known stability requirement

$$\Delta t \left( \frac{1}{\Delta x^2} + \frac{1}{\Delta y^2} \right) \leq \frac{1}{2m}$$

(3.13)

holds in two space dimensions. If the forward Euler scheme were to be applied in step 2 as well, $m$ would be replaced by one in (3.13) (again neglecting non-linear terms). Since $m$ is typically smaller than one, e.g. $m \approx 0.05$ for nickel, this would imply a more strict step size restriction. Equation (3.13) is an upper bound for the largest allowed time step since the non-linear functions will be zero away from the interface. We note that the value of $m$ given by the thin interface model (2.52) allows larger time steps than the value predicted by the sharp interface model (2.51).

This scheme is limited to first order accuracy in time, even if more accurate methods are used in steps 1 and 2 above. A similar approach based on operator splitting can be employed to formulate a second order accurate numerical method. The underlying idea is to apply the linear and non-linear operators in (3.10) one at a time, rather than updating the temperature and phase field separately. In one dimension the solution is advanced one time step by solving two uncoupled heat equations, and two scalar non-linear equations in each gridpoint. The stability region and the accuracy increase while maintaining a low computational cost.

The operator splitting approach can not be as successfully employed in two dimensions, since the inclusion of anisotropy introduces a non-linear coupling between values in different gridpoints. The splitting approach has been used only as a predictor, see Section 3.2.2.

3.2.2 Methods Based on Full Discretization

We wish to relax the stability restriction (3.13) and increase the order of accuracy. To do this we shall consider standard numerical methods developed for ODEs for advancing the solution to the semi-discrete system (3.10) in time. Schemes based on BDF-methods, explicit and implicit Adams methods as well as explicit and singly diagonally implicit Runge-Kutta methods have been implemented for the one dimensional spherically symmetric case. After numerical experiments the BDF-methods were chosen due to both good stability properties and high orders of accuracy.

For a general ODE $y'(t) = f(t, y)$ the BDF-methods take the form

$$\sum_{i=0}^{k} \alpha_i^{(k)} y((n - i) \Delta t) = \Delta t f(n \Delta t, y(n \Delta t)),$$

where $1 \leq k \leq 6$ is the order of accuracy and $\alpha_i^{(k)}$ are known constants. Generalization to adaptive time stepping is relatively straight-forward, and will introduce a dependence of $\alpha_i^{(k)}$ on the time steps [19]. The BDF-methods have desirable stability properties. In the sense of theory of numerical methods for ODE, the first and second order methods are $A$-stable, and methods of order up to and including six are $A(\alpha)$-stable [19, 20]. BDF(1) is the well-known backward Euler method.

BDF$(k)$ is a $k$-step multistep method and the order of accuracy is determined by the number of previous solutions used to approximate the time derivative. Thus one can obtain
a high order of accuracy in the temporal discretization with little extra computational work at the expense of increased storage requirements. As will be seen in Section 4.2 the extra storage required is small compared to that used by the non-linear solver. Solutions must be known at \( k \) previous time levels in order to apply the BDF\((k)\)-method. To avoid having to specify initial data on several time levels, simulation is typically initiated with \( k = 1 \). In each successive time step \( k \) is then increased until the desired accuracy is obtained.

Applying BDF-methods to the semi-discrete system (3.10) results in the expression

\[
\sum_{i=0}^{k} c_i \Phi^{n-i} - \partial_t \left[ -\frac{\Phi^n}{\Delta t} \right] = \Delta t \left[ F(\Phi^n, U^n) \right].
\]  

(3.14)

As in Section 3.2.1 we choose to write the approximate time derivative of \( p(\phi) \) on conservative form to ensure conservation of energy. The BDF-methods are implicit and \( \Phi^n \) and \( U^n \) at time level \( n \) appears as unknowns in Equation (3.14). The non-linear system must be solved once per time step to advance the solution in time.

A predictor is used to determine an initial guess to the solution of (3.14), and will to some extent determine how fast the non-linear system can be solved. In two dimensions a predictor based on a first order operator splitting discretization of (3.10) for \( \gamma = 0 \) is employed. This approach was found to yield much better approximations than simply taking the solution at the old time level even though it does not account for anisotropy.

## 3.3 Non-linear Solver

### 3.3.1 Newton Iteration

To formulate an algorithm for solving (3.14) we rewrite the equation as \( G(\xi) = 0 \), where the vector \( \xi \) contains the unknown grid values \( \phi_{i,j}^n \) and \( u_{i,j}^n \) at time \( n \Delta t \). A sequence of approximate solutions \( \{\xi\} \) can be obtained by Newton iteration,

\[
\xi_{i+1} = \xi_i - \lambda t \left( \frac{\partial G}{\partial \xi}(\xi_i) \right)^{-1} G(\xi_i) =: \xi_i + \lambda d_i,
\]  

(3.15)

given an initial guess \( \xi_0 \). For \( \lambda = 1 \) the classical Newton’s method is obtained, and by allowing other \( \lambda \) we get the damped Newton iteration. Newton’s method is attractive because it guarantees quadratic convergence in the vicinity of the root if the Jacobian is non-singular. However, if the initial guess is far from the root \( \{\xi\} \) may converge slowly, irregularly or not at all. In the present context this corresponds to the predictor giving a poor approximation of the solution at the next time level, which will occur more frequently as the time steps are increased.

In the line search framework damping factors \( \lambda \) are determined adaptively such that acceptable next iterates are found, thereby increasing the domain of convergence while retaining quadratic convergence in the vicinity of the root [13]. We begin by observing that solving the non-linear system can be transformed into the equivalent problem of finding zeros to

\[
g(\xi) = \frac{1}{2} \left\| G(\xi) \right\|_2^2,
\]

and modifications to Newton’s method can be derived by considering methods for determining local minima of \( g \).
3.3. Non-linear Solver

Noting that $d_i$ is a descent direction of $g$ at $\xi_i$, the residual norm is guaranteed to decrease for $\lambda_i$ sufficiently small. We formulate the condition

$$g(\xi_i + \lambda_i d_i) \leq g(\xi_i) + \alpha \lambda_i \nabla g(\xi)^T d_i, \quad \alpha \in (0, 1),$$

(3.16)

which can be interpreted as requiring the average rate of decrease in $g$ to be at least some fraction $\alpha$ of the initial rate of decrease. Condition (3.16) does not impose any constraints on the $\lambda_i$, and may result in $\lambda_i \ll 1$. We formulate the additional constraint that the rate of decrease of $g$ in the direction $d_i$ at the next iterate should be larger than some prescribed fraction of the rate of decrease at the current iterate,

$$\nabla g(\xi_{i+1})^T (\xi_{i+1} - \xi_i) \geq \beta \nabla g(\xi_i)^T (\xi_{i+1} - \xi_i), \quad \beta \in (\alpha, 1),$$

(3.17)

which can be formulated in terms of $\lambda_i$ and $d_i$ by use of (3.15). Conditions (3.16) and (3.17) will be referred to as the $\alpha$- and $\beta$-conditions.

It can be shown that for $\alpha < 1/2$, the $\alpha$- and $\beta$-conditions can be satisfied simultaneously. Furthermore, if the local convergence assumptions underlying Newton's method are met the sequence $\{\xi_i\}$ converges superlinearly to a minima of $g$ for $\lambda_i$ satisfying (3.16) and (3.17). For $i$ greater than some $i_0$, the convergence rate will be quadratic [13].

The backtracking modification is formulated from a minimization problem and may give convergence to a non-zero minima of $g$. This is rarely seen in simulation. It is sufficient that any of the $\xi_i$ lies in the domain of convergence of Newton's method to guarantee convergence to the root.

It remains to determine $\lambda_i$ such that the $\alpha$- and $\beta$-conditions are satisfied, which is done by backtracking. In each Newton iteration we begin by investigating whether (3.16) and (3.17) are satisfied for the full Newton step, $\lambda_i = 1$. If this is not the case, we note that some information on $g$ is known from this investigation. By fitting an interpolating polynomial in $\lambda_i$ to the known data we obtain an analytic expression from which to determine the next value of $\lambda_i$. After having tried the full Newton step this polynomial is of second degree, and in successive backtracking a third degree polynomial is used. Some heuristic limits on how much $\lambda_i$ is allowed to change in a single backtracking are used to handle the case when the interpolating polynomial is a poor model of $g(\xi_i + \lambda_i d_i)$. Full details are given in [13].

There are other ways of increasing the domain of convergence of Newton iteration. One alternative is the model-trust region approach, which is based on the observation that if the full Newton step is not an acceptable next iterate the underlying assumption on the behavior of $g$ is not satisfied for all $0 < \lambda \leq 1$. Instead of determining a damping factor and moving along the Newton direction, one tries to determine a model-trust region in which $g(\xi_i + \lambda_i d_i)$ can be trusted to behave like a quadratic in $\lambda_i$. In this region we then try to determine a search direction $s_i$ such that $g(\xi_i + s_i)$ is minimized. This approach has the advantage of taking into account the full dimensional behavior of $g$. It may however require the solution to more than one linear system in each Newton iteration. As will be discussed in Section 3.3.2 the linear systems are solved by iterative methods, and since a matrix factorization is not available the amount of work may increase significantly.

A reverse communication implementation of Newton's method has been written in C based on the modular algorithmic formulations provided in [13]. It supports classical Newton iteration, damped Newton iteration and the backtracking line search modification with or without enforcing the $\beta$-condition. The code runs on both conventional and parallel architectures.

Figure 3.1 shows the number of iterations required for three of the algorithms applied to the simple scalar equation $G(\xi) = \xi - \cos \xi = 0$, as the initial guess approaches the point where $G'(\xi) = 0$. For the worst case the classical Newton's method converges in 902
iterations, whereas the backtracking line search modification yields convergence in at most 6 iterations (or 5 if the $\beta$-condition is enforced).

For purposes of dendritic growth simulation the backtracking line search modification allows larger time steps for the implicit scheme. As the time step increases it becomes more difficult to find a good predictor giving an initial guess to the solution of (3.14) such that the Newton iteration converges. Thus there is an upper bound on the time step imposed by the predictor. The increased domain of convergence relaxes this bound and increases the range of time steps available.

So far we have not discussed how to determine when convergence has occurred. For practical purposes we are usually not interested in solving the non-linear system to full machine precision. Instead we wish to solve the system to such an accuracy that error contributions from the solver are small compared to discretization errors. In the simulations below the iteration is terminated when the norm of the residual or the norm of the Newton correction is smaller than some fixed value. In most simulations the maximum norm of the residual is used.

We now proceed by discussing how to compute Newton directions by use of Krylov subspace methods.

3.3.2 Krylov Subspace Projection Methods

By solving the non-linear system (3.14) obtained by discretization of the PDEs using the modified Newton iteration described in the previous section the numerical scheme has been reduced to solving linear systems, $Jd = f$. This operation is expensive compared to the other operations in the Newton iteration and expected to account for a large fraction of the total run time. It is therefore important to find efficient methods for solving the linear systems.
3.3. Non-linear Solver

If the grid is mapped into a vector using the natural ordering (3.9) the matrices arising from the discretization will be sparse and banded. The matrix $J \in \mathbb{R}^{2N_x N_y \times 2N_x N_y}$ has upper and lower bandwidths $N_x N_y$ and fewer than $16N_x N_y$ nonzero elements, as depicted in Figure 4.1. If the system were to be solved using direct methods one would typically choose an ordering of the elements different from (3.9), in order to reduce the bandwidth. Even in this case the storage requirements of a direct method becomes prohibitive, and instead Krylov subspace projection methods have been considered [40].

This class of methods has several attractive properties, and can be given special interpretation in the context of Newton iteration. The methods have in common that approximate solutions $d_m$ are found in subspaces $d_0 + K_m$ of dimension $m$, where $d_0$ is some initial guess,

$$K_m(J, r_0) = \text{span} \{r_0, Jr_0, J^2 r_0, \ldots, J^{m-1} r_0 \}$$

is the Krylov subspace, and $r_0 := f - Jd_0$ is the initial residual. The approximations satisfy a Petrov-Galerkin condition $f - Jd_m \perp L_m$, where $L_m$ is a subspace of dimension $m$ whose basis will depend on the considered method. We note that the basis of the Krylov subspace is computed by repeated matrix-vector multiplications, and in fact most of the work is needed to compute and orthogonalize this basis.

The choice of Krylov method is to large extent determined by the matrix $J$. Since the matrices considered here are non-symmetric GMRES [40, 42] has been chosen. For GMRES one takes $L_m = J K_m$ resulting in approximations such that the residual norm is minimized over all vectors in $d_0 + K_m$ [40]. Other possible choices include the Conjugate Gradient Squared method (CGS), the Biconjugate Gradient Stabilized method (Bi-CGSTAB) and the Quasi-Minimal Residual method (QMR). QMR requires the matrix transpose to form $L_m$ and has not been considered since some additional difficulties are involved for efficient parallel implementation. CGS and Bi-CGSTAB are associated with smaller storage requirements than GMRES, but need twice as many matrix-vector products per iteration. A thorough exposition of Krylov methods and related topics can be found in [40].

Although there is an upper bound on the number of iterations needed to find the solution in exact arithmetic Krylov methods are often used to find a solution to a given tolerance. Otherwise the work would most likely exceed that of a direct solver, and for GMRES the storage requirements increase linearly with the number of iterations since the entire Krylov basis has to be stored. The GMRES implementations used here have been taken from SPARSKIT [39] and its parallel counterpart P SPARSLIB [41] where termination criteria based on the relative residual are used. Convergence occurs when the norm of the residual is smaller than some prescribed fraction of the residual norm of the initial guess.

To limit storage requirements restarting is used. An upper limit on the dimension of the Krylov subspace is determined and if convergence has not occurred when this limit is reached, iteration is restarted with the current approximation as a new initial guess. With restarting convergence of GMRES is no longer guaranteed but the approach works well for most cases.

The convergence of Krylov methods is strongly dependent on the spectra of the matrix $J$. Often more tractable problems can be obtained by preconditioning the system, i.e. by solving the system $M_L^{-1} J M_R^{-1} d = M_L^{-1} f$, where the matrices $M_L$ and $M_R$ are the left and right preconditioners and the solution to the original system is given by $d = M_R^{-1} d$. By appropriately choosing preconditioners whose inverses can be efficiently evaluated the convergence rate can be greatly increased at a low computational cost. The preconditioning can be incorporated in the GMRES algorithm so that no explicit changes have to be made to the matrix or the right hand side.
The efficiency of the fully implicit discretization described in Section 3.2.2 is currently limited by the lack of a good preconditioner. Left Jacobi preconditioning is employed, for which \( M_L \) is the matrix containing the diagonal elements of \( J \).

In the context of Newton iteration there are a few additional points to be noted. The matrix appearing in the linear systems is the Jacobian \( J = \frac{\partial G}{\partial \xi} \), and Krylov methods require only the action of \( J \) on vectors. Thus we can use a finite difference approximation, e.g.

\[
\left( \frac{\partial G}{\partial \xi}(\xi_i) \right) \cdot v \approx \frac{G(\xi_i + \delta v) - G(\xi_i)}{\delta}, \quad 0 < \delta \ll 1,
\]

to avoid explicit representation of the Jacobian. In the current application the evaluation of the non-linear function is expensive compared to computing matrix-vector products and in most simulations an explicit expression for the Jacobian is computed despite the increase in storage requirements.

As previously noted in Section 3.3.1 the linear behavior of \( G(\xi) \) assumed in the formulation of Newton’s method may be a poor model far from the root. For this reason it is not always necessary to compute the exact Newton direction, and a “sufficiently accurate” approximation will do. It is possible to relate the accuracy needed in GMRES to the residual norm in the current Newton step, and formulate sufficient conditions for superlinear or quadratic convergence [7]. In [6] it is shown how non-linear Krylov subspace correction methods can be formulated, including a Newton-GMRES algorithm based on backtracking line search techniques.

3.4 A Fast Poisson Solver

For the scheme based on equation splitting it is possible to solve the linear systems more efficiently using a fast Poisson solver [33]. The technique is based on the observation that the bandwidth of the linear systems can be reduced by applying the discrete cosine transform (DCT) to the discretized PDE. The solver requires less storage than GMRES, and gives the (numerically) exact solution in a fixed number of floating point operations.

With the natural ordering defined by (3.9) the system (3.12) will have dimension \( N_x N_y \), and a banded structure as depicted in the lower right submatrix of Figure 4.1. The elements located on diagonals at distance \( N_y \) from the main diagonal come from the difference operator acting in the \( y \)-direction, and can be eliminated by applying the DCT to the discretized PDE in the \( y \)-direction [33] thus reducing the system to tridiagonal form. The tridiagonal system can be solved efficiently using a direct solver, and after an inverse DCT the solution to the original system is obtained.

The cosine transform is used to ensure that zero Neumann conditions hold at the boundaries. It is possible to apply the DCT in the \( x \)-direction as well which would result in a diagonal system. The amount of work needed to compute a DCT is larger than the amount of work needed to solve a tridiagonal system. Hence the second transform is not performed.

The DCT of a vector of length \( n \) is most efficiently computed using a complex Fast Fourier Transform (FFT) of length \( n/2 \), where the elements are obtained by permuting data in the original vector [33]. Optimized FFT routines are available in the public domain, and here we have chosen to work with the FFTW library [15]. The DCT is the inverse of itself, apart from a constant factor, so the programming effort needed to implement a fast Poisson solver given an FFT-routine is small. The storage requirements can be kept low by using in-place transforms, where the input vector to the FFT-routine is overwritten by its Fourier transform.
3.4. A Fast Poisson Solver

We note that the fast Poisson solver relies heavily on the fact that the matrix is a
discretization of the standard five point stencil on a uniform grid. The approach is not
applicable to the anisotropic Laplacian (3.7) due to non-linearities in $\phi$. It is also difficult
to efficiently apply this approach on non-uniform grids, even if they are structured.

A parallel version of the fast Poisson solver has been implemented but due to permuta-
tion of the data vectors a low parallel efficiency is obtained. It has not been used for any
simulations presented here.
Chapter 4

Parallel Implementation

Two different methods for temporal discretization of the semi-discrete system (3.10) were introduced in Section 3.2. The fully implicit approach lead to large non-linear systems of equations, and is more expensive than the scheme based on equation splitting in terms of both floating point operations per time step and storage requirements. However, numerical experiments indicate that a substantial gain in accuracy can be obtained, see Section 5.1, and for this reason one would prefer to use the fully implicit discretization for simulation.

A parallel implementation has been written in C for simulation in two dimensions based on the MPI message passing library [17]. The underlying idea is rather simple: by letting several computers work on the same problem in parallel the computational resources are increased which can be exploited to reduce simulation times and/or increase problem sizes. The main difficulty in parallel computing is to formulate algorithms such that all computers are efficiently used. This problem is addressed in Section 4.1, and the achieved computational performance investigated in Section 4.3. Some technical details of the implementation are described in Section 4.2. The chapter is to some extent based on previous work presented in [3].

4.1 Parallelization Issues

4.1.1 Distributed Parallel Computing

The parallel architectures currently available can be roughly divided into two categories: shared memory and distributed memory architectures. Here we shall consider distributed memory architectures, whose characteristic feature is that each process has its own local memory, and the memory of other processes can not be accessed directly. The word “process” could for example be interpreted as a single processor in a multiprocessor system, or a computer in a network.

In a distributed memory setting communication between processes is necessary to get access to non-local data. One of the critical issues when designing programs is to keep the ratio of communication to computation small. We distinguish between local communication involving two processes, and global communication involving all processes. One would like to avoid the latter if possible since it means that all processes must reach a certain stage in the computation before the execution in any process can proceed. At these so called synchronization points all processes have to wait for the slowest one. Thus it is important to have a good load balance so that an equal amount of work is performed on all processes,
which can be accomplished by choosing a proper distribution of data across the processes, as discussed in Section 4.1.2. The data distribution also affects the amount of communication needed.

It should be noted that the amount of communication also depends on the choice of algorithm. For this application the algorithms used on a serial computer can also be applied in a parallel environment with little modification.

### 4.1.2 Data Distribution

To determine a data distribution which balances computation and minimizes communication a few key computational kernels are identified in the non-linear solver from Section 3.3:

- evaluating (3.14) and its Jacobian
- computing matrix vector multiplications
- computing vector updates (“saxpy’s”)
- computing scalar products and norms.

The vector update is a strictly local operation which does not require communication. Computing scalar products requires one global reduction, regardless of how the data is distributed across the processes. The matrix appearing in the matrix vector multiplications is the Jacobian of (3.14), and thus the communication pattern of this operation is identical to that of evaluating the non-linear function. Therefore we choose a data distribution such that (3.14) can be efficiently evaluated.

The structure of the Jacobian is shown in Figure 4.1, assuming that the vectors $\Phi$ and $U$ containing the grid values in the semi-discrete system (3.10) are stored in natural order (3.9). For parallel implementation the natural ordering is advantageous since it makes it easy to identify data dependencies. We note that the only part of the GMRES algorithm which depends on the ordering is the evaluation of matrix-vector products. In particular, the number of floating point operations per iteration is constant.
The Jacobian consists of four block matrices, each having $N_x N_y$ rows and columns. From (3.14) one sees that the upper part of the matrix corresponds to the discretization of (2.20), and the lower to the discretization of (2.21). Furthermore, the left half of the matrix shows dependence on $\phi$, and the right half dependence on $u$. Thus the structure of the upper left and lower right blocks are determined by difference approximations (3.6) and (3.7) to spatial derivatives, and the remaining two blocks come from the coupling between the PDEs.

To evaluate the non-linear function (3.14) in a gridpoint $\phi$ must be known in the eight neighboring gridpoints and $u$ in the closest four gridpoints. This statement holds also for matrix vector multiplications with the Jacobian if the upper part of the vector is interpreted as $\phi$, and the lower as $u$. In order to minimize communication the grid should be distributed so that as few needed gridpoints as possible are stored on external processes. For purposes of load balancing the number of gridpoints should be evenly distributed among the processes.

The grid is distributed row-wise across the processes, as exemplified in Figure 4.2, such that the number of elements is roughly equal on all processes. For an $N_x \times N_y$ grid distributed over $p$ processes approximately $N_x N_y / p$ gridpoints are stored locally. To compute the action of the difference operator one row needs to be sent to and received from the nearest neighbors. The fraction of elements to send and receive is $\approx 2p / N_y$, which for practical purposes means a few percent.

It is possible to find data distributions which reduce the amount of communication while maintaining load balance, e.g. by graph partitioning [40]. They are however more grid-dependent and require additional technicalities in the implementation. The simple distribution considered here should give equal performance when the fraction of elements to send and receive is sufficiently small.

4.2 Implementation

The parallel implementation is based on the more expensive of the two time stepping schemes described in Section 3.2. The primary focus has been to write an efficient im-
The implementation to allow accurate simulation in times comparable to the "cheaper" algorithm. For this reason existing numerical libraries have been used to a large extent. The only numerical routine written specifically for this application is the backtracking line search modification of Newton's method. All calls to basic linear algebra routines are made to ESSL (IBM SP2) or BLAS (Sun workstations). The GMRES implementation is taken from the PSPARSLIB library [41].

Inter-process communication has been implemented using MPI [17]. Non-blocking routines are used wherever possible to overlap computation and communication. Many of the operations needed for GMRES and the Newton iteration are inherently global and will introduce synchronization points. These are unavoidable in the Newton iteration since all global reductions are used either to determine whether convergence has occurred or to update the approximate solution.

The GMRES implementation uses the classical Gram-Schmidt algorithm (CGS) [40] to orthogonalize the newly computed vector against the previously computed vectors in each inner iteration. This may result in poor parallel performance since two global reductions are used for each orthogonalization. It is possible to orthogonalize the entire Krylov basis at once using e.g. the $m$-step GMRES($m$) algorithm [27], thus reducing the number of synchronization points. Such algorithms are not supported in the current release of PSPARSLIB.

It should be noted that the Modified Gram-Schmidt algorithm (MGS) is often preferred over CGS in serial implementations because of better stability properties [4]. In a distributed environment $j+1$ synchronization points are needed in the $j$th inner iteration of GMRES, and the orthogonalization becomes prohibitively expensive. Experiments have been performed using an in house implementation of GMRES supporting both CGS and MGS, as well as iterative versions. For a number of different test cases the two orthogonalizations yielded convergence in the same number of iterations, and the CGS orthogonalization appears to work well for the systems arising from this application.

Table 4.1 shows the distribution of the total run time over the different routines for one particular simulation. Notably computing the analytic Jacobian of (3.14) accounts for almost half of the total time. Still using the analytic expression instead of difference approximations significantly reduces the run time for most simulations at the expense of larger storage requirements. A more detailed look at the routines for computing (3.14) and its Jacobian show that almost 75% of the time is spent inside the trigonometric functions $\sin$, $\cos$ and $\tan2$ from the C math library. Thus there is a large part of the code which can not be optimized further. Temporary storage of computed values is used to avoid excessive calls to the trigonometric functions.

<table>
<thead>
<tr>
<th>Routine</th>
<th>Fraction of run time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analytic Jacobian of (3.14)</td>
<td>47.7 %</td>
</tr>
<tr>
<td>Evaluating (3.14)</td>
<td>23.8 %</td>
</tr>
<tr>
<td>ESSL</td>
<td>10.9 %</td>
</tr>
<tr>
<td>matrix vector multiplications</td>
<td>5.7 %</td>
</tr>
<tr>
<td>Preconditioning</td>
<td>1.4 %</td>
</tr>
<tr>
<td>Communication</td>
<td>1.1 %</td>
</tr>
</tbody>
</table>

Table 4.1. Distribution of total run time over communication and computational routines for a simulation on a 400 × 400 grid running on 4 IBM SP2 processors. Results were obtained by $gprof$. The time used for call counts was excluded before fractions were computed.
### 4.3. Computational Performance

<table>
<thead>
<tr>
<th>Purpose of allocated storage</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Solution at different time levels</td>
<td>$2(k + 1) N_x N_y / p$</td>
</tr>
<tr>
<td>Action of difference operator</td>
<td>$12 N_x$</td>
</tr>
<tr>
<td>Newton solver</td>
<td>$10 N_x N_y / p$</td>
</tr>
<tr>
<td>Krylov solver (PSPARSLIB [41])</td>
<td>$(10 + 4m) N_x N_y / p$</td>
</tr>
<tr>
<td>Jacobian of (3.14)</td>
<td>$16 N_x N_y / p$</td>
</tr>
<tr>
<td>Preconditioner (Jacobi)</td>
<td>$2N_x N_y / p$</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>$\approx (40 + 2k + 4m) N_x N_y / p$</td>
</tr>
</tbody>
</table>

Table 4.2. Approximate storage requirements per process measured in `sizeof(double)`. Simulation on $N_x \times N_y$-grid using a BDF($k$)-method and Krylov subspaces of dimension $m$ running on $p$ processes.

Both GMRES and the Newton solver are based on reverse communication protocols. The internal work in these routines make up the major part of the time used by ESSL. The 9% not accounted for by Table 4.1 are overhead, initialization and unidentified routines. This fraction agrees well with the measured speedup, see Table 4.3.

The approximate memory usage of the program is shown in Table 4.2. The reciprocals of the matrix diagonal are stored for the Jacobi preconditioner to avoid excessive divisions. Simpler versions of Newton iteration require less storage. One could also decrease storage requirements by choosing another Krylov method. Typically the dimensions of the Krylov subspaces are small, say 5–20, and the gain in storage is not sufficient to justify the increase in computational complexity.

An effort has been made to write modular code and the numerical methods (apart from the discretization) can be replaced with a relative small amount of work. The generalization to similar physical problems, such as solidification of binary alloys [49], is also straightforward. Both numerical and physical parameters used for simulation are read from an input file for flexibility.

### 4.3. Computational Performance

The scalability of the implementation has been investigated on two different architectures, the IBM SP2 *Strindberg* located at the Center for Parallel Computers (PDC), Stockholm, and networks of Sun Ultra 5 workstations. The (approximate) computational performance lies in the range of 80–150 MFlops on a single thin IBM SP 2 node, depending on grid size and data. Peak performances of up to 230 MFlops have been obtained.

Tables 4.3 and 4.4 show speedup on the different architectures for a few different grid sizes. Speedup is formally defined as the execution time of the fastest serial algorithm on one processor divided by the execution time of the parallel algorithm [18] and is a measure of the number of processors which can be efficiently used to solve a given problem. The results shown were obtained by comparison to a modified version of the parallel code, in which all the communication has been removed. The timings were measured over 100 time steps using `MPI_TIMe`. Realistic parameters were used and initial data was chosen such that the fastest transients had died out. The results should be representative for a typical simulation.

The speedup for the larger grids is more than optimal. For these simulations the numerical parameters used for the one processor job are different from the ones used on several processors due to RAM memory limitations. In most cases the dimensions of the Krylov
<table>
<thead>
<tr>
<th>Nodes</th>
<th>200 × 200</th>
<th>400 × 400</th>
<th>600 × 600</th>
<th>800 × 800</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>1.92</td>
<td>1.89</td>
<td>4.27</td>
<td>4.73</td>
</tr>
<tr>
<td>4</td>
<td>3.56</td>
<td>3.63</td>
<td>8.14</td>
<td>9.15</td>
</tr>
<tr>
<td>8</td>
<td>6.56</td>
<td>6.34</td>
<td>15.30</td>
<td>17.22</td>
</tr>
<tr>
<td>16</td>
<td>11.25</td>
<td>11.38</td>
<td>26.24</td>
<td>31.45</td>
</tr>
<tr>
<td>32</td>
<td>12.89</td>
<td>17.36</td>
<td>44.10</td>
<td>51.37</td>
</tr>
</tbody>
</table>

Table 4.3. Speedup on an IBM SP2 for different grid sizes

<table>
<thead>
<tr>
<th>Computers</th>
<th>200 × 200</th>
<th>400 × 400</th>
<th>600 × 600</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>1.91</td>
<td>2.61</td>
<td>4.09</td>
</tr>
<tr>
<td>4</td>
<td>3.83</td>
<td>5.17</td>
<td>31.03</td>
</tr>
<tr>
<td>8</td>
<td>6.14</td>
<td>10.00</td>
<td>60.60</td>
</tr>
<tr>
<td>16</td>
<td>7.34</td>
<td>16.49</td>
<td>93.78</td>
</tr>
</tbody>
</table>

Table 4.4. Speedup on clusters of Sun workstations (Ultra 3) for different grid sizes

subspaces have been decreased (compare Table 4.2). For the 600 × 600-grid on one Sun workstation, finite difference approximations were used to evaluate the action of the Jacobian.

The results are encouraging even for the smaller grids. All simulations on eight processors show an efficiency, defined as the ratio of speedup to the number of processors, exceeding 75%. On the IBM SP2 the efficiency is more than 70% for all grids on 16 processors.

Another scalability metric is the scaled speedup [18], which is a measure of how long a given parallel program would have taken to run on a hypothetical serial processor with the same amount of RAM memory as the parallel machine. The motivation for this metric is that with a larger machine one is able to solve larger problems, which is not reflected in the fixed-size speedup. By increasing the problem size linearly with the number of processors memory and surface-to-volume effects remain constant and usually the efficiency is higher than for the fixed-size case which is limited by Amdahl's law [18].

Since the considered implementation is based on iterative methods it is not obvious how to increase the problem size. Parameters such as grid resolution, time step and tolerances all affect execution time and performance. Furthermore, the convergence is to large extent dependent on data. For these reasons two different limits have been investigated: increased size of physical domain with all numerical parameters kept fixed, and decreased grid spacing with the physical problem kept fixed. In both cases the problem size was increased such that approximately 400² and 200² gridpoints reside on each processor on the IBM SP2 and Sun workstations respectively. The results are shown in Table 4.5.

The scaled speedup is closer to the optimal than the fixed-size speedup as expected. When the size of the physical domain is increased there is only a slight decrease in efficiency due to the global reductions. The results are not as good when the grid is refined since a larger number of iterations are required for GMRES to converge and thus more global reductions are needed. It is possible that a better efficiency can be obtained by changing the time step as well as the grid spacing.
### 4.3. Computational Performance

<table>
<thead>
<tr>
<th>Nodes</th>
<th>IBM SP2</th>
<th></th>
<th>Sun workstations</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Case 1</td>
<td>Case 2</td>
<td>Case 1</td>
<td>Case 2</td>
</tr>
<tr>
<td>2</td>
<td>1.99</td>
<td>1.92</td>
<td>1.92</td>
<td>1.67</td>
</tr>
<tr>
<td>4</td>
<td>3.97</td>
<td>3.63</td>
<td>3.72</td>
<td>3.31</td>
</tr>
<tr>
<td>8</td>
<td>7.94</td>
<td>7.60</td>
<td>6.78</td>
<td>6.14</td>
</tr>
<tr>
<td>16</td>
<td>15.70</td>
<td>12.90</td>
<td>12.71</td>
<td>10.04</td>
</tr>
</tbody>
</table>

*Table 4.5.* Scaled speedup for increased size of the physical domain with numerical parameters kept fixed (case 1), and decreased grid spacing with all other parameters kept fixed (case 2).
Chapter 5

Numerical Results

The fully implicit numerical scheme described in Section 3.2.2 is more expensive per time step than the one based on equation splitting and we would like to investigate whether one actually benefits from using it. The main issue is whether the less restrictive stability condition leads to reduced simulation times given a certain accuracy requirement. In Section 5.1 the asymptotic order of convergence for the different approaches is investigated for decreasing time steps with all other parameters kept fixed. The execution time is compared to achieved accuracy for simulations in two space dimensions in Section 5.1.1.

We performed a formal asymptotic analysis of the phase field model in Section 2.3 and discussed its interpretation in the sharp and thin interface limits. An experimental comparison of the two different asymptotic limits has previously been published by Karma and Rappel in [25], where interfacial velocities and tip shapes are compared to analytic solutions and results from boundary integral computations. In Section 5.2 we investigate errors in the dimensionless Gibbs-Thomson condition for decreasing \( \epsilon \) in two space dimensions. An example is also given where the use of the thin interface correction allows simulation with larger \( \epsilon \) without introducing unphysical behaviour of the solutions.

A simulation previously presented by Braun and Murray in [5] is reproduced in Section 5.3.1 to verify the parallel implementation. Effects of grid anisotropy and random perturbation are briefly examined in Section 5.3.2, and some two dimensional computations are presented.

5.1 A Comparison of Time Stepping Schemes

We investigate the order of accuracy in time by considering the semi-discrete system (3.10) in the spherically symmetric case. Dimensionless parameters are chosen to agree with pure nickel for a certain undercooling, \( \alpha = 400 \), \( m = 0.05 \) and \( \Delta = 0.5 \) [51]. It is necessary to have \( \epsilon \lesssim 0.005 \) to avoid unphysical oscillations in the phase field variable for this particular set of parameters, which is discussed in more detail in Section 5.2 below. We choose \( \epsilon = 0.0025 \), which is the smallest value considered in [51]. The results presented here have been obtained in the sharp interface limit of the asymptotics. The errors and asymptotic convergence rates depicted in Figures 5.1 and 5.2 are almost identical when the thin interface correction is employed.

Computations are performed on the domain \([0, 2]\) using the second order accurate discretization discussed in Section 3.1.1, with \( \Delta x = \epsilon/2 \). The phase field equations are solved
Figure 5.1. Approximate relative errors in computed solutions measured in Euclidean norm.

Figure 5.2. Order of convergence estimated by (5.2) using Euclidean norm.
5.1. A Comparison of Time Stepping Schemes

in time until \( T = 0.13 \) starting from a spherical solid of radius \( r_0 = 0.5 \) and uniform undercooling (3.1). Initial data is needed on more than one time level for BDF-methods of order two or higher. We first solve the phase field equations for \( 0 \leq t \leq 0.03 \) using a BDF(3)-method with half the smallest step size used in the convergence experiments. By saving the computed solutions at appropriate time levels, simulation can then be initiated from \( t = 0.03 \) with accurate initial data on all time levels. At this time the fastest initial transients have died out.

When computations are initiated at \( t = 0.03 \) the solid-liquid interface is located at \( r \approx 0.72 \). At the final time \( t = 0.13 \) the location is \( r \approx 1.02 \), so the interface is moving with an average dimensionless speed 3. The grid size and integration time have been chosen such that the temperature at the outer boundary \( r = 2 \) remains close to \(-1\) throughout the simulation.

Time stepping for the approach based on equation splitting uses the forward Euler method for the phase field as in (3.11), and the Crank-Nicholson scheme to update temperature. Due to stability smaller time steps have been chosen than for the implicit scheme. The one dimensional case is particularly simple since the linear systems are tridiagonal, and can be solved directly without using a fast Poisson solver. The number of floating point operations per time step is proportional to the number of grid points.

A number of additional parameters related to the non-linear solver must be set. These parameters have an impact on the convergence of the iterative methods, and hence the total execution time of the program. To ensure that errors from the solver are small compared to errors from discretization we set tolerances close to the limits imposed by IEEE floating point arithmetics. The non-linear systems are solved by the backtracking line search modification of Newton’s method without enforcing the directional derivative condition. Iteration is interrupted when the discrete maximum norm of the residual (3.14) is smaller than \( 10^{-12} \).

We iterate over Krylov subspaces of dimension 20 using an implementation of FGMRES taken from SPARSKIT [39] to determine Newton directions. Left Jacobi preconditioning is employed and convergence is said to have occurred when the Euclidean vector norm of the residual is smaller than \( 10^{-14} \).

Newton’s method typically converges in 3-5 iterations. With the possible exception of the first one or two iterations convergence is quadratic. The backtracking line search modification significantly reduces the number of iterations for the larger time steps considered.

Figure 5.1 shows approximate relative errors in the computed solutions. The values were obtained by comparison to the solution expected to be the most accurate, that is, the one computed with the smallest time step and highest order of accuracy. Attempts to compare results from phase field simulations to sharp interface solutions computed using a numerical method from [43] have been made. As the asymptotic order of convergence of the sharp interface scheme was found to be 1/2, and a more accurate scheme would necessitate solving a large full system of non-linear equations this approach was not pursued further.

We expect the norm of the error in computed solutions to behave as

\[
e c \Delta t^p + O(\Delta t^{p+1})
\]

for a fixed time as \( \Delta t \downarrow 0 \) since \( c \) and the grid spacing are kept fixed. By determining the constants \( c \) and \( p \), errors from temporal discretization of the semi-discrete system can be related to errors from spatial discretization.

The order of convergence is estimated by

\[
p = \log_2 \left( \frac{\| \Phi_{2\Delta t} : U_{2\Delta t}^N - \Phi_{\Delta t} : U_{\Delta t}^N \|}{\| \Phi_{\Delta t} : U_{\Delta t}^N - \Phi_{\Delta t/2} : U_{\Delta t/2}^N \|} \right).
\]
Table 5.1. Order of convergence and error constant estimated by (5.2) and (5.3) for the finest
time step. Errors are measured in Euclidean norm of the solution vector.

<table>
<thead>
<tr>
<th>Scheme</th>
<th>$c$</th>
<th>$p$</th>
</tr>
</thead>
<tbody>
<tr>
<td>BDF(1)</td>
<td>$3.4 \cdot 10^2$</td>
<td>1.01</td>
</tr>
<tr>
<td>BDF(2)</td>
<td>$8.4 \cdot 10^4$</td>
<td>1.99</td>
</tr>
<tr>
<td>BDF(3)</td>
<td>$9.8 \cdot 10^6$</td>
<td>3.07</td>
</tr>
<tr>
<td>Equation splitting</td>
<td>$1.2 \cdot 10^2$</td>
<td>1.00</td>
</tr>
</tbody>
</table>

where the superscript denotes the time level at which the solutions are compared, and
the subscript denotes the time step used (c.f. Richardson extrapolation). $[\Phi; U]$ should
be interpreted as a column vector containing the grid values of \( \phi \) and \( u \). The estimate is
relevant for sufficiently small time steps when the leading order error term dominates over
higher order terms in (5.1). Equation (5.2) is attractive because it estimates \( p \) by comparing
computed solutions, and does not require the actual error.

Estimated convergence rates are shown in Figure 5.2. The order of the fully implicit
discretizations approach the order of the underlying BDF-methods as \( \Delta t \) is decreased.
The scheme based on equation splitting is first order accurate but seems to converge to a slightly
different solution, see Figure 5.1. These results were obtained for specific dimensionless pa-
rameters in the phase field model, but the asymptotic convergence rates have been observed
for other choices as well. Almost identical curves are obtained when other norms, such as
the maximum norm, are used in (5.2).

The error constant \( c \) in (5.1) can be approximated by

\[
    c = \frac{\| [\Phi^N_{2\Delta t} \; U^N_{2\Delta t}] - [\Phi^N_{\Delta t} \; U^N_{\Delta t}] \|}{\| 2p - 1 \|^{1/\Delta} t^p},
\]

where \( p \) is the limit of (5.2) as \( \Delta t \downarrow 0 \). From Figure 5.2 we see that \( p \) is close to its limit
only for the smaller time steps, and hence relevant estimates of \( c \) can not be obtained for
the larger \( \Delta t \).

Approximated values of \( c \) and \( p \) for the finest time step are given in Table 5.1. The
differences compared to the next finest time step are below 5\% for all schemes. For the
BDF-methods the error constant grows by a factor of approximately 100 as the order is
increased. Since the largest time step considered is \( 1.25 \cdot 10^{-3} \) one benefits from using a
more accurate BDF-method, which is also seen from Figure 5.1.

The number of floating point operations needed for the scheme based on equation splitting
is inversely proportional to the time step. With all output from the program removed
slightly more than a minute is needed for the largest time step. It is more difficult to predict
the execution time for the implicit scheme since it depends on the convergence of iterative
methods. Initially the run time decreases with the time step since the predictor yields a
more accurate initial guess for Newton’s method. More non-linear systems must be solved
as the time step is decreased, and after a certain point the execution time increases. The
second order accurate BDF-scheme requires approximately 90 minutes for the time step
\( \Delta t = \Delta x/8 \). The execution time can be reduced to 27 minutes with an increase of 1\% in
the error (compared to the reference solution) by changing the tolerances to \( 10^{-6} \) and \( 10^{-8} \)
in the non-linear solver and FGMRES respectively. Increasing the tolerances by a factor
1000 more results in a 38\% increase of the error, and a total run time of 13 minutes. The
run times can probably be reduced further by employing a more efficient preconditioner.
5.1. A Comparison of Time Stepping Schemes

![Figure 5.3](image_url)

**Figure 5.3.** Phase field and temperature for a case where the scheme based on equation splitting fails to capture the correct propagation speed. The curves show the solution at times 0.05, 0.1, ..., 0.25, moving from left to right.

The scheme based on equation splitting may fail to capture the correct propagation speed of the interface. Such behavior is more frequently observed as the propagation speed increases. Figure 5.3 shows results from a simulation performed with \( \alpha = 400 \), \( m = 0.05 \), \( \Delta = 1.0 \), and \( \epsilon = \Delta x = 0.005 \). Growth is initiated from a spherical seed of radius 0.5 and uniform undercooling. Time steps have been chosen smaller than or equal to \( 5 \cdot 10^{-5} \), which is within the stability domains of the considered schemes. The errors from the splitting approach are much larger for this case than for the one discussed above.

We note two things. There is a significant difference between the computed solutions for the two different time steps when the splitting approach is used, which can be explained by the first order accuracy. Secondly, the deviation in interface position increases with time which indicates that the interfacial velocity is not captured correctly.

The BDF(2)-discretization is less sensitive to the choice of time step, and reducing \( \Delta t \) by a factor 10 does not alter the plotted solution. To obtain visual agreement between the two schemes a \( \Delta t \) of the order \( 10^{-5} \) must be used in the scheme based on splitting. At this point the execution time is a few times larger than for the implicit approach.

### 5.1.1 Execution Time in Two Space Dimensions

The spherically symmetric case is sufficient for investigating the asymptotic order of convergence of the time stepping schemes, but we are interested in performing simulations in two and three space dimensions. The linear systems arising from the splitting approach will no longer be tridiagonal and can not be solved using direct methods. For purposes of simulation it is therefore more relevant to compare execution times to achieved accuracy in two space dimensions.

We choose the same physical parameters as above: \( \alpha = 400 \), \( m = 0.05 \) and \( \Delta = 0.5 \). In addition to these we set the anisotropy parameters in (2.24) to \( \gamma = 0.015 \), \( k = 4 \) and \( \theta_0 = 0 \).
Simulations are performed on a 201 × 201-point discretization of the domain [0, 1] × [0, 1], with \( \Delta x = \Delta y = \epsilon = 0.005 \). The grid is coarser and \( \epsilon \) larger than for the one dimensional simulations to allow computation on a single computer within a couple of hours.

To obtain initial data the phase field equations are solved for \( 0 \leq t \leq 0.015 \) using a BDF(3)-method with time step \( \Delta t = 7.8125 \cdot 10^{-5} \), starting from a phase field given by (5.5) with \( r_0 = 0.05 \), and \( u(x, y) = -\phi(x, y) \).

In Figure 5.4 the achieved accuracy compared to a reference solution is plotted versus the execution time. The smallest time step used is given in the plot, and each successive data point corresponds to an increase in \( \Delta t \) by a factor two. It is possible to use larger time steps for the BDF(2)- and BDF(3)-schemes but large execution times are needed due to poor convergence in the GMRES iteration.

Figure 5.4 suggests that the fully implicit scheme is more efficient if a high accuracy is desired. We note that by using the parallel implementation described in Chapter 4 the run times can be reduced further for the fully implicit scheme, see Tables 4.3 and 4.4.

5.2 Sharp vs. Thin Interface Asymptotics

To investigate implications of the thin interface correction on computational efficiency we first consider the one dimensional phase field model, with dimensionless parameters given by \( \alpha = 200, \; m = 0.05, \; \Delta = 0.5 \). In the thin interface limit the value of \( m \) is corrected according to (2.52). For simulations the grid spacing is set to \( \Delta x = 0.005 \) and BDF(2) is employed for time stepping with \( \Delta t = 10^{-4} \).

Figure 5.5 shows results from computation with \( \epsilon = 0.01 \). For the sharp interface limit an unphysical oscillatory intermediate state appears in the region where the phase field changes value, which is not seen in the thin interface limit. Furthermore, by examining the
5.2. Sharp vs. Thin Interface Asymptotics

Figure 5.5. Simulation resulting in an unphysical intermediate state of the phase field in the sharp interface interpretation of the model because of a too large value of \( \epsilon \).

Time dependent behavior we find that the width of the oscillatory transition region increases with time. This phenomena is analyzed in [16] and it is shown to be a property of the phase field model independent of discretization and numerical methods. An upper bound is given on the size of \( \epsilon \alpha \) such that no unphysical intermediate states appear. The upper bound depends on \( m \) and becomes less restrictive when the thin interface correction is employed.

The correct qualitative behavior is obtained when \( \epsilon \) is decreased or \( m \) is modified according to the thin interface correction (2.52). For this problem \( \epsilon \) must be reduced to approximately 0.007 without the thin interface correction. If the ratio \( \epsilon / \Delta x \) is kept fixed this corresponds to a 40% larger grid.

We note that the agreement between the phase field model and the Stefan problem depends on the size of \( \epsilon \), and also that the unphysical intermediate states observed here can arise in the thin interface interpretation but for larger values of \( \epsilon \).

From the formal asymptotics we expect the thin interface interpretation to give better agreement with the interface conditions as it is more accurate in terms of \( \epsilon \). We now investigate this conjecture in two space dimensions to avoid the enforced curvature of the spherical symmetry assumption. The Gibbs-Thomson condition is examined at the point where the interface intersects the x-axis which greatly simplifies the extraction of the necessary data from the computed solutions. Because of difficulties in accurately estimating the temperature gradients to the left and right of the interface the jump condition is not considered.

We need to know interfacial velocity, curvature and temperature, which are determined as follows. In each time step a cubic spline interpolant with zero Neumann conditions prescribed at the boundaries is fitted to the phase field along the x-axis. The position of the interface is determined by using Newton’s method to find the point where the spline interpolant of the phase field is equal to 1/2. Convergence is typically achieved in two or three iterations to a tolerance of \( 10^{-15} \). At this point the curvature is approximated by a
second order finite difference approximation to

\[ K = \frac{\phi y y}{\phi x}, \]

where boundary conditions have been used to eliminate some terms appearing in the expression for the curvature in a general point along the interface. The temperature is determined by evaluating a spline interpolant at the interface. We fit a cubic spline interpolant to the interface position as a function of time and evaluate its derivative to get an approximate velocity. This is done in the post-processing stage when the computation has been completed.

By inserting the interfacial quantities extracted from the computation in the dimensionless anisotropic Gibbs-Thomson condition [51],

\[ u = -\frac{1}{6\sqrt{2}\Delta}\left( \frac{\nu}{m\eta^2(\theta)} + (\eta(\theta) + \eta''(\theta))K \right), \tag{5.4} \]

we get an estimate of how well it is satisfied. Such a comparison is shown in Figure 5.6, for the case \( \alpha = 400 \), \( m = 0.05 \) and \( \Delta = 0.5 \), and a few different values of \( \epsilon \). Fourfold anisotropy of magnitude 0.015 is used with initial data chosen as for the example discussed in Section 5.3.1 below. The phase field equations are solved on a 601 \times 601-point discretization of [1.5, 1.5] using the BDF(2)-scheme with time step \( 5 \cdot 10^{-5} \).

Figure 5.6 indicates that the thin interface asymptotics not only increase the range of \( \epsilon \) available for simulation, but also gives better agreement with the Gibbs-Thomson condition. With the thin interface correction a twice as large \( \epsilon \) can be used without an increase in the error. The curves plotted in Figure 5.6 show results after the fastest transients have died out and the dendrite tip has approximately constant curvature and propagation speed. This state is not reached in the computation with \( \epsilon = 0.01 \). We note however that \( \epsilon = 0.01 \) leads to unphysical intermediate states without the thin interface correction.

One can use (5.4) and information extracted from the computations to determine the “effective value” of \( m \) instead of the error in the Gibbs-Thomson condition. Such a comparison shows that with the thin interface correction the effective value of \( m \) resulting from simulation lies closer to the desired value 0.05.

5.3 Computed Dendrites

5.3.1 A Reproduced Example

In [5] Braun and Murray present adaptive computations based on the phase field model introduced in Chapter 2 with the sharp interface interpretation of the asymptotics. To verify the parallel implementation and illustrate how the increased computational resources can be exploited one of these computations have been reproduced.

The physical parameters used for the simulation are \( \alpha = 400 \), \( m = 0.035 \) and \( \Delta = 0.25 \). Four-fold anisotropy is prescribed by setting \( k = 4 \) and \( \gamma = 0.015 \) in (2.23). Following [5] the reference direction \( \theta_0 = 0 \) is chosen so that the preferred growth directions lie along the \( x \)- and \( y \)-axes. The initial phase field is given by

\[ \phi(x, y, 0) = \frac{1}{2} \left[ \tanh \left( \frac{\sqrt{x^2 + y^2} - r_0(x, y)}{2\sqrt{2}\epsilon} \right) + 1 \right], \]

where

\[ r_0(x, y) = 0.05 \left( 1 + 0.5 \cos \left( \tan^{-1} \left( \frac{y}{x} \right) \right) \right), \tag{5.5} \]
5.3. Computed Dendrites

and the initial temperature is taken to be \( u(x, y, 0) = -\phi(x, y, 0) \). \( \rho_0(x, y) \) expresses the approximate location of the solid-liquid interface along the straight line passing through the origin and \((x, y)\).

Computations are carried out on a \( 601 \times 601 \)-point discretization of \([0, 3] \times [0, 3]\) with constant time step \( \Delta t = 0.001 \) using the BDF(2)-scheme. The time step chosen here is smaller than motivated by stability and the domain of convergence of Newton's method. An empirical choice was made in an attempt to reduce the wall clock time needed for the entire simulation.

The backtracking line search modification of Newton's method with an analytic expression for the Jacobian is used to solve the non-linear systems. Iteration is interrupted when the largest element of the residual has modulus smaller than \(10^{-6}\). The GMRES implementation from PSPARSLIB [41] discussed in Section 4.2 is employed for solving the linear systems with left Jacobi preconditioning. To reduce storage requirements we use Krylov subspaces of dimension ten. Solutions are determined such that the relative residual is smaller than \(10^{-6}\). Convergence in the Newton iteration is achieved in 3–6 iterations after the fastest initial transients have died out.

Braun and Murray employed a finite difference scheme based on local uniform grid refinement in space and an adaptive BDF(2)-method to advance the solution in time. The dendrite shape in Figure 5.7 computed using the parallel implementation from Chapter 4 agrees well with [5, Fig. 1].

This particular test case has been presented in a previous work [3] with larger error tolerances in the non-linear solver, and a termination criteria based on the Euclidean norm of the residual. The results are almost identical to the ones computed here. A comparison shows that the largest difference in the computed phase fields corresponds to a shift in the interface location of approximately 0.005 at the final time. Figure 5.7 therefore appears to be identical to the corresponding figure in [3].

\[
\text{Figure 5.6. Difference between computed interfacial temperature, and temperature predicted by Gibbs-Thomson condition (5.4). The solid lines are computed with the thin interface correction to } m \text{, and the dotted lines without the correction.}
\]
5.3.2 Effects of anisotropy

In addition to the anisotropy modeled by (2.23) there will be a certain grid-induced anisotropy coming from the truncation error in the difference approximation (3.7). Effects of this contribution are numerically investigated by simulating the solidification of pure nickel using different reference directions $\theta_0$ in (3.8). We use ourfold anisotropy of magnitude $\gamma = 0.015$ and set remaining parameters to $\alpha = 400$, $\Delta = 0.5$ and $m = 0.05$. Computations are performed on a 1024 x 1024 grid with grid spacing $\Delta x = \Delta y = \epsilon = 0.005$ using the scheme based on equation splitting with time step $10^{-4}$. The initial seed is circular of radius 0.05 and the temperature distribution given by $u(x, y) = -\phi(x, y)$.

Figure 5.8 shows results obtained with $\theta_0 = 0$ and $\theta_0 = 45^\circ$, where the curves from the latter case have been rotated $45^\circ$ to simplify comparison. The actual computation was performed in one quarter of the domain shown, and has been reflected around the $x$- and $y$-axes. The agreement between the two solutions along the principal growth direction is within 3%. In the direction where the slowest growth is expected there is a significant difference and a different number of “fingers” are observed for the two cases.

The solidification of pure nickel has also been simulated with a random perturbation of 1% amplitude added to the non-linear function $F(\Phi, U)$ of (3.10) to account for impurities in the material and thermal noise [26, 51]. Figures 5.9 - 5.11 show the results in the absence of anisotropy ($\gamma = 0$), with fourfold anisotropy ($k = 4$) and with sixfold anisotropy ($k = 6$) respectively. The solutions have been reflected around the $x$- and $y$-axes. Hence the apparent symmetry in the crystal shapes.

Apart from noting that the actual number of symmetry axis correspond to the value of $k$ chosen for the simulation, we see that the propagation speed of the interface increases with $k$. This effect is accounted for by the second derivative of $\eta(\theta)$ appearing in the Gibbs-Thomson condition (5.4).
5.3. Computed Dendrites

![Figure 5.8](image)

**Figure 5.8.** Simulation of dendritic growth with different reference directions for the anisotropy. The dashed line corresponds to \( \theta_0 = 0 \), and the solid line to \( \theta_0 = 45^\circ \). The location of the crystal melt interface is shown for times \( t = 0, 0.125, \ldots, 0.5 \).

From physics one would expect Figure 5.10 to be the relevant computation since it agrees with the BCC-lattice observed in solid nickel. By comparison to Figure 5.8 we see that the random perturbation leads to the appearance of sidearms which are not observed in the absence of perturbation. Also, only one finger appears in the least preferred growth direction which suggests that the grid anisotropy is sufficiently small to have a negligible qualitative impact on the solution.

A simulation with fourfold anisotropy and reference direction \( \theta_0 = 45^\circ \) is shown in Figure 5.12. The shape of the crystal is different from Figure 5.10 due to random perturbation but the position of the solid-liquid interface along the principal growth direction agrees within 2%. This should be compared to the difference in position along the \( x \)- and \( y \)-axes in Figure 5.10 which is strictly due to the random perturbation and approximately equal to 1.5%. The experiments indicate that the effects of grid anisotropy are less important in the presence of random perturbation.

Finally, an example of solidification starting from two initial seeds is shown in Figure 5.13. Computations are carried out on a \( 1536 \times 1536 \)-grid with the reference direction of the fourfold anisotropy chosen to be \( \theta_0 = 45^\circ \), and the remaining parameters chosen as above. The initial solids are circular with radii 0.05, and located at \( (x, y) = \pm(0.2, 0.2) \).
Figure 5.9. Solidification of pure nickel in the absence of anisotropy. The solid lines show the location of the solid-liquid interface at time levels $t = 0, 0.2, 0.4, 0.6$. The background color represents the heat distribution at $t = 0.6$.

Figure 5.10. Solidification of pure nickel with four-fold anisotropy. The solid lines show the location of the solid-liquid interface at time levels $t = 0, 0.2, 0.4, 0.6$. The background color represents the heat distribution at $t = 0.6$. 
5.3. Computed Dendrites

Figure 5.11. Solidification of pure nickel with six-fold anisotropy. The solid lines show the location of the solid-liquid interface at time levels $t = 0, 0.2, 0.4, 0.6$. The background color represents the heat distribution at $t = 0.6$.

Figure 5.12. Solidification of pure nickel with four-fold anisotropy and reference direction $\theta_0 = 45^\circ$. The solid lines show the location of the solid-liquid interface at time levels $t = 0, 0.2, 0.4, 0.6$. The background color represents the heat distribution at $t = 0.6$. The plot has been rotated $45^\circ$ to simplify comparison with Figure 5.10. The white regions are not represented on the computational grid.
Figure 5.13. Solidification of pure nickel starting from two initial seeds. The solid lines show the location of the solid-liquid interfaces at time levels $t = 0, 0.125, \ldots, 0.5$. The background color represents the heat distribution at $t = 0.5$. 
Chapter 6

Conclusions

We have seen how a phase field model can be used to simulate dendritic solidification of a pure material from its undercooled melt without explicitly tracking the solid-liquid interfaces. Based on [2, 14, 22, 23] it is described in Section 2.3 how a more accurate asymptotic analysis can be exploited to obtain better agreement between the phase field model and the corresponding Stefan problem. This is experimentally validated in Section 5.2, where it is shown that employing the thin interface correction leads to an increased accuracy in the Gibbs-Thomson condition. Furthermore, the correction allows simulation with a larger width of the diffuse interfaces without introducing unphysical oscillatory intermediate states.

In [2] it is shown how the thin interface asymptotics can be extended to a phase field model with different diffusivities in solid and liquid. The analysis yields some additional constraints on $p(\phi)$ and $g(\phi)$ which must be satisfied if the errors in the interface conditions are to be of size $\mathcal{O}(\epsilon^2)$. It would be interesting to apply the thin interface asymptotics also to other physical problems.

A semi-explicit first order accurate time stepping scheme based on equation splitting used by many researchers, see e.g. [25, 51], is compared to a more accurate fully implicit scheme. Asymptotic convergence rates of orders 1–3 are verified in Section 5.1 for the implicit scheme, but up to sixth order accuracy has been observed in simulations not presented here. Experiments described in Section 5.1.1 indicate that the fully implicit scheme, which is more expensive per time step, is more efficient than the semi-explicit scheme when one is interested in quantitative computations.

The efficiency of the fully implicit scheme is currently limited by the lack of a good preconditioner. Some preliminary experiments have been performed in Matlab using a preconditioner based on an Incomplete LU (ILU) factorization [40]. The number of floating point operations needed to solve the system can be reduced by as much as 99% for time steps of the same size as the grid spacing. It is however difficult to implement an ILU Factorization preconditioner efficiently on parallel architectures, even though there are parallel variants. We plan to investigate preconditioners based on domain decomposition and sparse approximate inverses [40].

The parallel implementation described in Chapter 4 makes it possible to perform large simulations in reasonable time, as illustrated in Section 5.3.1 where results of adaptive computations are reproduced. It will also be a useful tool for computing reference solutions when formulating an adaptive algorithm in the future.
Bibliography


Bibliography


