A Fixed-Grid Numerical Method for Dendritic Solidification with Natural Convection

Thesis by

Patrick M. Lahey

In Partial Fulfillment of the Requirements
for the Degree of
Doctor of Philosophy

California Institute of Technology
Pasadena, California

1999

(Submitted December 18, 1998)

Acknowledgements

First and foremost, I would like to express my deepest love and appreciation to my future wife, Brunella, for her enduring love and support. My parents and siblings have been a source of constant encouragement and warmth. I do not know what I would have done without them. I would also like to gratefully acknowledge the patience, guidance and support provided by my advisor, Professor Daniel I. Meiron. Finally, I would like to thank the Fannie and John Hertz Foundation for their financial support during the initial stages of this project.

Abstract

The solidification of a material into an undercooled melt occurs quite frequently in material processing applications. The interface between the solid and liquid phases in such cases is inherently unstable. This instability can lead to the formation of dendritic growth patterns which may significantly impact the microstructure of the resulting solid. Because the microstructure of materials notably influences their macroscopic properties, there is significant interest in understanding and controlling the formation and evolution of dendrites.

For many years, material scientists have sought to develop a predictive theory that could relate the observed dimensions and characteristics of dendrites to the thermal and fluid dynamics conditions that prevailed during their formation. To date, no such general theory exists. The problem is a difficult one, both from an experimental and mathematical standpoint.

In this work, we develop an accurate numerical method capable of simulating dendritic solidification both with and without natural convection effects. The scheme explicitly tracks and parametrizes the interface between the liquid and solid phases using a series of independent marker particles. Due to the release of latent heat, the derivatives of the temperature of a growing dendrite are discontinuous across the interface. As a consequence, great care is required when discretizing the derivatives at nodes adjacent to the interface. We use a generalized version of LeVeque and Li's immersed interface method to accurately compute the spatial derivatives. We also develop an accurate one-step time marching scheme for problems with derivatives that jump discontinuously across a moving interface. The method is notable because it does not require that the same time discretization scheme be applied to every term in the governing equation.

Contents

List of Figures

List of Tables

Chapter 1 Introduction

1.1 Motivation

Although foreign to residents of warm weather climates, virtually everyone who lives in the "snow belt" is familiar with dendritic crystals. Frost patterns and snow flakes are among the most common examples in nature of materials that result from dendritic solidification. The appearance of a snow flake is clearly quite different from that of an ice cube even though both are made from exactly the same material. The difference between the two is due to the conditions that prevailed during their formation. This applies not just to ice, but to most materials. The physical, chemical and mechanical properties of a solid typically depend on the thermal and fluid dynamic conditions in the liquid during its solidification. The determination of the relationship between a material's properties and the conditions under which it was formed is not merely of academic interest. Most materials of technological interest have, at some point, solidified from a liquid state. The conditions during this solidification process can significantly influence the material properties and, hence, engineering utility of the resulting substance (see [34]). It is well known that materials resulting from the solidification of an undercooled liquid (i.e., a liquid below its equilibrium freezing temperature) can, and frequently do, have a dendritic microstructure. Many materials, including alloys and semi-conductors, are solidified under these conditions so there is significant interest in understanding and controlling the evolution of growing dendrites.

Dendritic growth patterns can occur during the solidification of both pure and multi-component liquids. In this work, we will limit our consideration to pure substances. Even for a single component system, insight into the root cause of the rich and complex observable phenomena has proven to be elusive. In Fig. 1.1 we have reproduced a photograph of a succinonitrile dendrite. Succinonitrile (SCN) is a commonly used material in dendritic solidification experiments. Its main advantage over other materials is that its material properties (i.e., density, thermal conductivity, etc.) are virtually identical in its liquid and solid states. As we shall see, this simplifies the mathematical model for the problem. From an experimental point of view, it is convenient to work with because its melting temperature is fairly close to room temperature (about 58^o C). Scientists and engineers wish to be able to predict characteristic properties of dendrites like the one in Fig. 1.1. The goal is to be able to predict growth rate, tip radius, side branching density and other characteristics as a function of the prevailing thermal and flow field conditions in the liquid. Although progress has been made (see [21]), a complete theory capable of reliably predicting these quantities has not yet been developed.

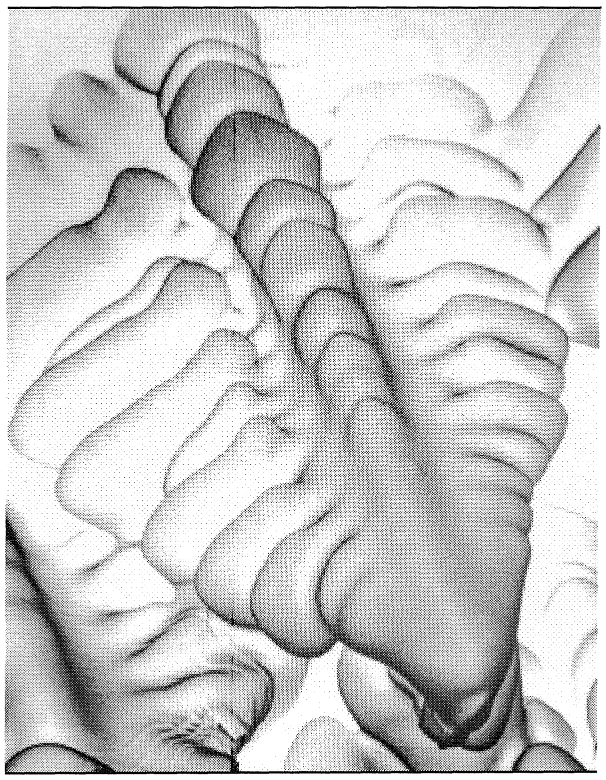

Figure 1.1: Succinonitrile dendrite

There have been many impediments to the development of a successful predictive theory for dendritic solidification. The necessary purity of the sample materials and exacting degree of thermal control, among other things, pose difficult challenges for the experimentalist. The inherently non-linear nature of the mathematical models has also proved troubling for mathematicians and physicists. Given these problems, it seems clear that numerical methods offer researchers a valuable additional tool which can complement experimental investigations. Numerical methods can provide information that is difficult to measure experimentally. In addition, it is possible to directly ascertain the impact of different physical aspects of the problem by simulating the system with or without that bit of physics.

1.2 Scope of this work

In this work, we have begun the process of developing a general tool for the investigation of dendritic solidification. We have constructed several front tracking/fixed (Eulerian) grid, staggered mesh schemes capable of simulating periodic problems with moving boundaries. In particular, we develop methods capable of simulating:

1. the flow of an incompressible fluid through an irregular domain with fixed or deforming boundaries
2. the dendritic solidification of a pure substance in the absence of convection and
3. the dendritic solidification of a pure substance in the presence of natural convection.

All of these methods are based upon a common discretization technique which is a generalization of the immersed interface method of LeVeque and Li (see [38]). While this work is exclusively focused on second order accurate solutions, the discretization approach that we develop is capable of generating arbitrarily accurate stencils. We check the accuracy of all our numerical schemes using exact and approximate solutions. In every case, second order accuracy is observed.

1.3 Outline of the remaining chapters

In Chapter 2, we discuss the standard mathematical models for dendritic solidification both with and without natural convection. Two exact one-dimensional solutions are presented that illustrate the behavior of the temperature field near the interface and provide valuable benchmarks for our numerical scheme. We also perform a linear stability analysis to determine the stability of a planar interface propagating at a constant speed (one of the exact solutions we provide). The analysis is performed both with and without natural convection effects. The resulting stability predictions indicate a dependence of the growth rate on the orientation of the system. That is, the growth rate can be different depending on whether the interface moves with or against the direction of gravity. This dependence has been noted by experimentalists (see [37]). These approximate solutions are valuable both for the insight they provide into the behavior of a solidifying system and as checks on our two-dimensional numerical method.

In Chapter 3, we introduce the discretization techniques that will be used throughout this work. Our method discretizes the system on fixed (Eulerian) finite difference grids. Special care is required at those stencils that span the interface (i.e., involve both liquid and solid nodes) because, as we will see in Chapter 2, the derivatives of the temperature are discontinuous across the interface. We develop a generalized version of the immersed interface method (see [38]) which allows the accurate discretization of the derivatives of a function even if its value and/or derivatives jump across the interface. We also derive a one-step time marching scheme that is capable of accurately evolving a function forward, even if its time derivative jumps, at a prescribed time, during the simulation.

In Chapter 4, we develop numerical methods in one and two dimensions for the dendritic solidification problem in the absence of convection. In one dimension, the domain is discretized by a uniformly spaced finite difference grid and the interface is tracked by an independent marker particle. We investigate a problem with non-symmetric (unequal) material properties to illustrate the theory developed in Chapter 3. In two dimensions, the domain is discretized by a uniformly spaced cell-centered finite difference grid and a set of independent marker particles are used to explicitly parametrize and track the interface. We restrict our attention to symmetric problems, in this case, which allows the discrete system to be solved efficiently. The scheme is validated against the exact and approximate solutions developed in Chapter 2.

In Chapter 5, we develop a scheme for the simulation of incompressible fluid flow in complex geometries. The domain is discretized by a fixed, uniformly spaced staggered mesh. A set of independent marker particles are used to explicitly parametrize and identify the position of the liquid-solid interface. A flow field is computed everywhere in the computational domain regardless of whether that region is liquid or solid. The method simulates the presence of a solid boundary by the application of a forcing at the interface such that the fluid satisfies the no-slip condition at that location. In this way, an independent flow is determined in both the liquid and solid regions of the computational domain. The flow in the liquid region is physically meaningful while the flow computed in the solid region is not. The advantage of this approach is that the system can be inverted using "fast solvers" (see [68]) which offsets the additional degrees of freedom introduced by the fictitious flow in the solid region. An important degeneracy in the system for the interfacial forcing is identified and eliminated and problems with mass conservation are also discussed. The scheme is validated using a combination of simple exact solutions and a convergence study.

In Chapter 6, we couple the techniques introduced in Chapters 4 and 5 and develop a scheme for the simulation of dendritic solidification in the presence of natural convection. The method employs a regularly spaced staggered mesh with the interface being tracked and parametrized by a series of independent marker particles. The scheme is validated against the approximate solutions developed in Chapter 2. We also demonstrate a case in which natural convection has a significant impact on the evolution of a growing dendrite.

In Chapter 7, we summarize the original contributions made in this work and discuss various avenues of future research that we feel would extend this work in useful ways.

Chapter 2 Dendritic solidification

2.1 Mathematical model

Solidification is the process of a liquid undergoing a change of phase into a solid. This typically occurs at an interface between a region that has already solidified and the remaining liquid. The solid grows by the accretion of liquid molecules effectively moving the interface. It is important to recognize that, unlike many moving boundary problems, the interface in solidification is not a material surface. The interface only moves through the acquisition or emancipation of material, hence motion is always accompanied by the release (solidification) or absorption (melting) of latent heat. The manner in which this latent is removed from the interface has a significant impact on the evolution of the geometry of the solid and liquid regions. When the liquid is above the melting temperature of the material, latent heat is removed through the solid and the interface evolves as a smooth regular surface that is stable to small perturbations of its geometry. When the liquid is undercooled below the thermodynamic melting temperature of the solid, however, we will show that the interface is unstable to small geometric perturbations and can evolve into a highly complicated shape. The case of a solid solidifying into an undercooled liquid is called dendritic solidification because of the branching tree like structures that often result (dendrite comes from the Greek word for tree).

First, we establish the equations governing the transfer of heat in each phase. In the solid region, heat transfer occurs solely by conduction. Thus the temperature of the solid, $T_S\left(\mathbf{x},t\right)$, satisfies

$$\rho_S c_S \frac{\partial T_S}{\partial t} = K_S \nabla^2 T_S, \tag{2.1}$$

where ρ_S is the density of the solid and K_S and c_S are the thermal conductivity and specific heat, respectively, of the solid. Note that we are assuming in this work that all material properties are constant in each phase. In the liquid region, heat can be transported either by conduction or through the physical motion of the material. Thus the temperature of the liquid, $T_L(\mathbf{x}, t)$, is governed by

$$\rho_L c_L \left(\frac{\partial T_L}{\partial t} + \mathbf{u} \cdot \nabla T_L \right) = K_L \nabla^2 T_L, \tag{2.2}$$

where $\mathbf{u}$ is the local velocity of the liquid and ρ_L, K_L and c_L are the liquid density, thermal conductivity and specific heat, respectively. In order to simplify the model, the fluid motion in the liquid phase is frequently neglected. Under this assumption, the temperature in the liquid phase satisfies

$$\rho_L c_L \frac{\partial T_L}{\partial t} = K_L \nabla^2 T_L, \tag{2.3}$$

which eliminates the need to consider the additional equations governing the motion of the fluid. While there are cases where the above is a valid approximation, a large body of evidence exists (see [37]) indicating that the effects of convection are crucial in some circumstances. Thus, it is necessary to introduce additional equations governing the fluid motion. In the absence of some external forcing such as the motion of a container wall or an imposed pressure gradient, the motion of the liquid is due solely to the effects of buoyancy. Buoyancy here refers to the small but important variation of liquid density in a nonuniform temperature field which can induce fluid motion. We will utilize the Boussinesq approximation (see [36]) which accounts for the effects of buoyancy through the addition of a temperature dependent forcing term to the incompressible Navier-Stokes equations. The Boussinesq equations apply only in the liquid and require that both incompressibility (governed by the continuity equation),

$$\nabla \cdot \mathbf{u} = \mathbf{0}, \tag{2.4}$$

and conservation of momentum,

$$\left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} \right) = -\nabla p + \nu \nabla^2 \mathbf{u} + \beta \left(T_L - T_{ref} \right) \mathbf{g}, \tag{2.5}$$

are satisfied. We must introduce several new variables to include fluid motion effects. These are the pressure, p, the fluid viscosity, ν, the coefficient of thermal expansion, β, which is evaluated at the reference temperature, T_{ref}, and the acceleration due to gravity, $\mathbf{g}$.

The above equations specify the physical processes occurring in the bulk liquid and solid regions. In addition to these, it is necessary to supply conditions that stipulate the behavior of the fluid velocity and temperature on the solidifying interface, $\mathbf{X} = (X, Y)$, between the two phases.

From the point of view of the liquid, the interface is the position of a solid boundary. Thus, the tangential velocity of a viscous fluid will satisfy

$$\mathbf{u} \cdot \boldsymbol{\tau} = 0. \tag{2.6}$$

The interface propagates into the fluid not by pushing the adjacent liquid away but by acquiring it. Thus, the normal velocity of the liquid is not equal to the normal velocity of the interface but, in the most general case, is related to it. If the density of the solid and liquid phases differ, then conservation of mass requires that there be a flow into or out of the interface

$$\mathbf{u} \cdot \mathbf{n} = \left(\frac{\rho_S - \rho_L}{\rho_L} \right) \left(\frac{\partial \mathbf{X}}{\partial t} \cdot \mathbf{n} \right). \tag{2.7}$$

In the simulations we perform that include convection, the symmetric model (material properties equal in each phase) is used so, for our purposes, the liquid satisfies

$$\mathbf{u} = \mathbf{0} \tag{2.8}$$

at the interface. Next, we examine the consequences of conservation of energy. The interface moves solely by the melting or solidification of material. Accompanying this phase change is the liberation of latent heat. Thus, the heat flux must be discontinuous across the interface in response to this release of energy. The exact condition is

$$(K_S \nabla T_S - K_L \nabla T_L) \cdot \mathbf{n} = \rho_S L \left(\frac{\partial \mathbf{X}}{\partial t} \cdot \mathbf{n} \right) \tag{2.9}$$

where L is the latent heat of fusion, $\mathbf{n}$ is the normal to the interface pointing out of the solid and $\mathbf{X}$ is the position of the interface. If the motion of the interface were known a priori, then the above system of equations would be sufficient to determine the temperature and fluid velocity at any point later in time. Unfortunately, the motion of the interface is not known and must be determined as part of the solution. Thus an additional equation is required to close the system and determine the motion of the interface. This last constraint arises from the consideration of the thermodynamics of the problem.

In elementary thermodynamics, it is assumed that a liquid freezes at a constant temperature. This is an extremely good approximation in most situations. In the case of the solidification of an undercooled liquid (a liquid cooled below its equilibrium freezing temperature), however, the assumption of isothermal solidification leads to mathematically ill-posed models (see [35]). The situation can be rescued, however, by the inclusion of some additional physics. The Gibbs-Thomson condition,

$$T_S(\mathbf{x}, t) = T_L(\mathbf{x}, t) = T_M \left(1 - \frac{\kappa}{L} \gamma(\mathbf{n}) \right), \tag{2.10}$$

governs the solidification temperature at the interface (see [35]) where T_M is the thermodynamic melting temperature, γ is the surface tension, κ is the local interface curvature and L is the latent heat of fusion. Note that the surface tension can depend on the normal, $\mathbf{n}$, to the interface which enables crystalline anisotropy effects to be modeled. We will introduce a specific functional form for the surface tension in Chapter 4. It is possible to include additional terms to the above that model, for instance, the finite rate at which liquid molecules attach to the solid (kinetic effects) or the effect of discontinuous heat capacity (see [29]) but this is sufficient for our purposes.

In practice it is useful to re-scale the above equations so that we can work in terms of dimensionless quantities. The scalings that we use are discussed in Appendix A.

In terms of the dimensionless temperature, θ, the system governing solidification in the absence of convection is

$$\frac{\partial \theta_S}{\partial t} = H_S \nabla^2 \theta_S, \tag{2.11}$$

in the solid,

$$\frac{\partial \theta_L}{\partial t} = H_L \nabla^2 \theta_L, \tag{2.12}$$

in the liquid. On the interface the temperature satisfies

$$\theta_S = \theta_L = -\varepsilon_c(\mathbf{n})\,\kappa, \tag{2.13}$$

and

$$(h_S \nabla \theta_S - h_L \nabla \theta_L) \cdot \mathbf{n} = \left(\frac{\partial \mathbf{X}}{\partial t} \cdot \mathbf{n} \right). \tag{2.14}$$

We will discuss the functional form for the dimensionless capillary parameter, ε_c, later in this chapter. The dimensionless system governing solidification with natural convection is

$$\frac{\partial \theta_S}{\partial t} = H_S \nabla^2 \theta_S, \tag{2.15}$$

in the solid,

$$\boldsymbol{\nabla} \cdot \mathbf{u} = 0, \tag{2.16}$$

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \boldsymbol{\nabla} \mathbf{u} = -\nabla p + D \nabla^2 \mathbf{u} + \mathbf{B} \theta_L, \tag{2.17}$$

and

$$\frac{\partial \theta_L}{\partial t} + \mathbf{u} \cdot \boldsymbol{\nabla} \theta_L = H_L \nabla^2 \theta_L, \tag{2.18}$$

in the liquid. On the interface the fluid velocity satisfies

$$\mathbf{u} = \mathbf{0}, \tag{2.19}$$

and the temperature satisifies

$$\theta_S = \theta_L = -\varepsilon_c(\mathbf{n})\,\kappa, \tag{2.20}$$

and

$$(h_S \nabla \theta_S - h_L \nabla \theta_L) \cdot \mathbf{n} = \left(\frac{\partial \mathbf{X}}{\partial t} \cdot \mathbf{n} \right). \tag{2.21}$$

The direction of $\mathbf{B} = (B_x, B_y)$ determines the orientation of the system with respect to gravity. The direction of gravity determines the direction of the buoyant forcing in the fluid which, as we shall see, can have a major impact on the evolution of a solidifying system.

The above equations are highly non-linear due to the motion of the solidification interface. Because of this there are very few exact solutions available. Indeed, this is the main motivation for the development of numerical methods capable of simulating solidification problems. Developing numerical methods for complicated systems of equations is a difficult business, however. It is important to have checks on the results that the algorithms generate to ensure that more than just "pretty pictures" are being produced. Several exact solutions are discussed below that we have found to be useful as benchmarks for our numerical method.

2.2 Analytic solutions

We seek one-dimensional (flat interface) solutions to the dendritic solidification system without convection. That is, we wish to determine the location of the interface, $y = Y(t)$, such that the temperature in the solid, $y \leq Y(t)$, satisfies

$$\frac{\partial \theta_S}{\partial t} = H_S \frac{\partial^2 \theta_S}{\partial y^2}, \tag{2.22}$$

and the temperature in the liquid, $y > Y(t)$, satisfies

$$\frac{\partial \theta_L}{\partial t} = H_L \frac{\partial^2 \theta_L}{\partial y^2}. \tag{2.23}$$

At the interface, $y = Y(t)$, the temperature must satisfy

$$\theta_S = \theta_L = 0 \tag{2.24}$$

and

$$h_S \frac{\partial \theta_S}{\partial y} - h_L \frac{\partial \theta_L}{\partial y} = \frac{\partial Y}{\partial t}. \tag{2.25}$$

As is the case with most moving boundary problems, there are very few exact solutions known for the above system. We present two solutions here that describe the motion of a flat advancing interface into an undercooled liquid. They are both examples of dendritic solidification which are useful for validating one-dimensional simulations. Because these problems are physically unstable, however, fully two-dimensional simulations can have trouble maintaining the flat interface solution as time advances. Later we will use perturbation techniques to construct approximate solutions that are better suited for such comparisons.

2.2.1 The sub-critically undercooled case

First we consider the evolution of a solid region at a uniform temperature of $\theta_S = \theta_{-\infty} \geq 0$ (clearly $\theta_{-\infty} = 0$ is the most physically relevant choice) and a liquid region undercooled at uniform temperature of $\theta_L = \theta_{+\infty} \leq 0$ that are brought into contact at $y = 0$. Seeking a similarity solution in terms of

$$\eta = \frac{y}{2\sqrt{t}}, \tag{2.26}$$

the system can be reduced to a single non-linear algebraic equation for a,

$$\frac{h_S \theta_{-\infty} \exp\left(-\frac{a^2}{H_S}\right)}{\sqrt{H_S}\left(1 + \operatorname{erf}\left(\frac{a}{\sqrt{H_S}}\right)\right)} + \frac{h_L \theta_{+\infty} \exp\left(-\frac{a^2}{H_L}\right)}{\sqrt{H_L}\left(1 - \operatorname{erf}\left(\frac{a}{\sqrt{H_L}}\right)\right)} + \sqrt{\pi} a = 0, \tag{2.27}$$

where a is related to the interface position by

$$Y(t) = 2a\sqrt{t}. \tag{2.28}$$

The temperature in the solid, $y \leq Y$, is given by

$$\theta_S = \theta_{-\infty} - \theta_{-\infty} \left[\frac{1 + \operatorname{erf}\left(\dfrac{y}{2\sqrt{tH_S}} \right)}{1 + \operatorname{erf}\left(\dfrac{a}{\sqrt{H_S}} \right)} \right] \tag{2.29}$$

and the temperature in the liquid, $y \geq Y$, by

$$\theta_L = \theta_{+\infty} - \theta_{+\infty} \left[\frac{1 - \operatorname{erf}\left(\dfrac{y}{2\sqrt{tH_L}} \right)}{1 - \operatorname{erf}\left(\dfrac{a}{\sqrt{H_L}} \right)} \right], \tag{2.30}$$

where erf() denotes the error function (see [1]). A typical temperature profile generated by the above solution is plotted in Fig. 2.1. Note the discontinuity in the derivative of the temperature across the interface at $\theta = 0$. This is a representative temperature profile for a solidifying system.

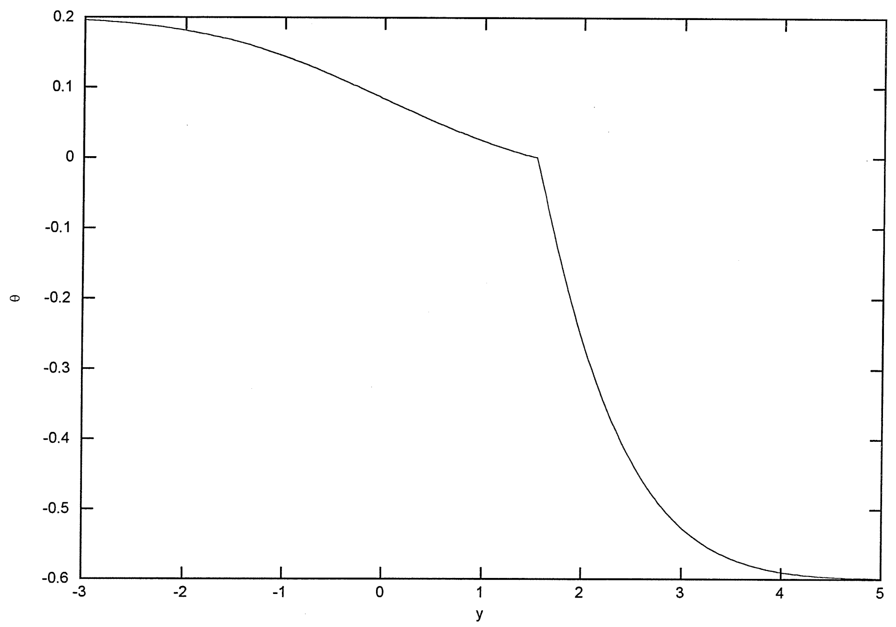

Figure 2.1: Temperature profile at $t = 1$ with $T_\infty = -0.6$ and $T_{-\infty} = 0.2$

It is not possible to find a solution of the above form for all possible initial temperature distributions. Consider, for example, the simple case in which the solid is initially at the melting temperature. Rearranging the above equation for a and substituting in $b = \frac{a}{\sqrt{H_L}}$ yields

$$-\frac{h_L}{H_L}\theta_{+\infty} = \sqrt{\pi} b e^{b^2} \left[1 - \operatorname{erf}(b)\right] \tag{2.31}$$

which has a bounded monotonically increasing right-hand side. Letting $b \to \infty$ we find that the liquid undercooling must satisfy

$$-\theta_{+\infty} < \frac{H_L}{h_L} = \Delta_c \tag{2.32}$$

for there to be a solution of the above form. We will refer to Δ_c as the critical undercooling. The solution above is only achievable for undercoolings that are smaller than this amount. Physically this means that for sufficiently large undercooling there is no buildup of latent heat ahead of the interface as it propagates into the liquid. Thus, solutions with under-coolings greater than this must have a different functional dependence. As we see below, this is indeed the case.

2.2.2 The critically undercooled case

When the liquid is sufficiently undercooled, it is possible for the latent heat released during solidification to be dissipated without building up near the interface. This can be demonstrated by seeking "traveling wave" solutions,

$$\theta_S = \theta_S(y - Vt), \tag{2.33}$$

$$\theta_L = \theta_L(y - Vt), \tag{2.34}$$

where the interface, $Y(t) = Vt$, travels at constant velocity. Seeking solutions of the above form, we find that the solid must be uniformly at the melting temperature

$$\theta_S = 0 \tag{2.35}$$

and that the liquid temperature must decay exponentially ahead of the interface

$$\theta_L = \Delta_c \left[\exp\left(-\frac{V\,(y - Vt)}{H_L} \right) - 1 \right]. \tag{2.36}$$

This solution is valid for all velocities $V \geq 0$ but has a unique undercooling of

$$\theta_\infty = -\Delta_c. \tag{2.37}$$

It is possible to find solutions with undercooling below this by including molecular attachment effects in the Gibbs-Thomson relation (see [50]), but this will be sufficient for our purposes. A plot of the temperature profile with $V = 1/2$ is shown in Fig. 2.2.

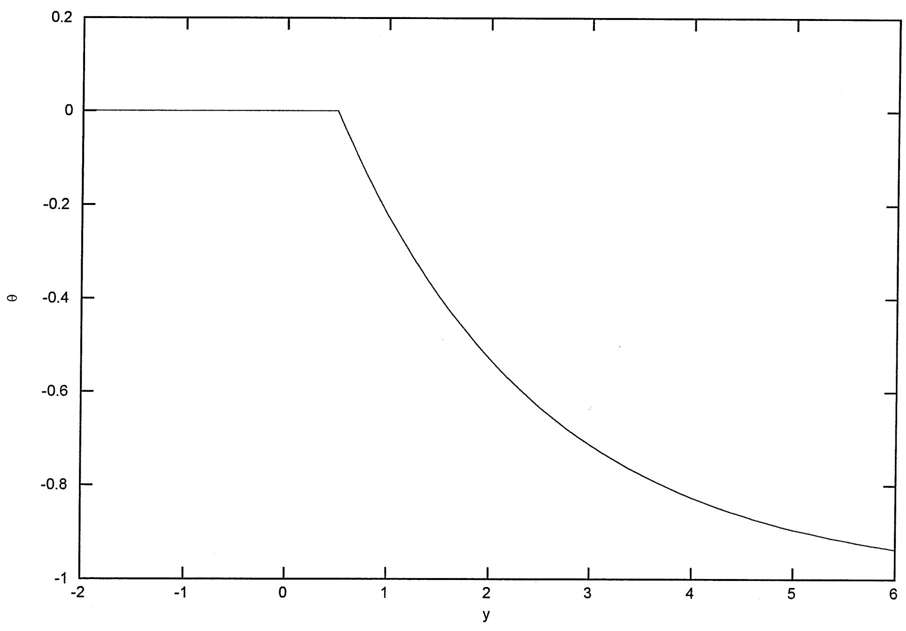

Figure 2.2: Temperature distribution at $t = 1$ for $V = 1/2$ and all material properties set to unity

Both of these solutions provide valuable checks for our numerics. We will use them both in Chapter 4.

2.3 Stability of a flat interface

We are interested in examining the stability of a solidifying planar interface, $y = Y(t)$, that is subject to a small morphological (shape) perturbation. This problem has been studied extensively in the literature both with and without convection (see [20] and the references therein).We will restrict our attention to the case in which the material properties are identical in each phase (the symmetric model), so that

$$H_L = H_S = H, \tag{2.38}$$

$$h_L = h_S = h. \tag{2.39}$$

The dimensionless governing equations for this problem are (2.15)-(2.21). We will only consider problems for which the interface grows against (moves up) or with (moves down) gravity so

$$B_x = 0. \tag{2.40}$$

Infinitely far from the interface we assume that the temperature in both the liquid and the solid decay to constants

$$\lim_{y\to-\infty} \theta_S(x,y,t) = \theta_{-\infty}, \tag{2.41}$$

$$\lim_{y\to+\infty} \theta_L(x,y,t) = \theta_{+\infty}, \tag{2.42}$$

and that there is no motion in the liquid

$$\lim_{y\to+\infty} u(x,y,t) = 0, \tag{2.43}$$

$$\lim_{y\to+\infty} v(x,y,t) = 0. \tag{2.44}$$

The quantities in our system that explicitly depend on the geometry of the interface, parametrically given by $(X(q,t), Y(q,t))$, are

$$n_x = \sin(\phi) = -\frac{\partial Y}{\partial q}\left[\left(\frac{\partial X}{\partial q}\right)^2 + \left(\frac{\partial Y}{\partial q}\right)^2\right]^{-\frac{1}{2}}, \tag{2.45}$$

$$n_y = \cos(\phi) = \frac{\partial X}{\partial q}\left[\left(\frac{\partial X}{\partial q}\right)^2 + \left(\frac{\partial Y}{\partial q}\right)^2\right]^{-\frac{1}{2}}, \tag{2.46}$$

$$\kappa = \left[\frac{\partial^2 X}{\partial q^2}\frac{\partial Y}{\partial q} - \frac{\partial^2 Y}{\partial q^2}\frac{\partial X}{\partial q}\right]\left[\left(\frac{\partial X}{\partial q}\right)^2 + \left(\frac{\partial Y}{\partial q}\right)^2\right]^{-\frac{3}{2}}, \tag{2.47}$$

$$\varepsilon_c(\phi) = \delta_0\left[1 + \delta_1 \sin^2\left(\frac{k_A\phi}{2}\right)\right]. \tag{2.48}$$

The above form for the capillary parameter is equivalent to the conventionally assumed form,

$$\varepsilon_c(\phi) = \bar{\varepsilon}_c\left[1 - A_c\cos(k_A\phi)\right], \tag{2.49}$$

if we let

$$\delta_0 = \bar{\varepsilon}_c(1 - A_c) \tag{2.50}$$

and

$$\delta_1 = \frac{2A_c}{1 - A_c} \tag{2.51}$$

where the constant $\bar{\varepsilon}_c$ is directly related to the material properties of the substance and the values of the constants A_c and k_A are selected to model the anisotropy of the material.

For this stability analysis, we set

$$q = x, \tag{2.52}$$

so that the shape of interface is reduced to a simple function of x,

$$X = x, \tag{2.53}$$

$$Y = Y_0(t) + \varepsilon Y_1(x,t), \tag{2.54}$$

where $\varepsilon \ll 1$. We seek a solution of the form

$$u = \varepsilon u_1(x,y,t), \tag{2.55}$$

$$v = \varepsilon v_1(x,y,t), \tag{2.56}$$

$$p = p_0(x,y,t) + \varepsilon p_1(x,y,t)\,, \tag{2.57}$$

$$\theta_L = \theta_{L0}(x,y,t) + \varepsilon \theta_{L1}(x,y,t)\,, \tag{2.58}$$

$$\theta_S = \theta_{S0}(x,y,t) + \varepsilon \theta_{S1}(x,y,t)\,. \tag{2.59}$$

Neglecting all $O(\varepsilon)$ terms, the system, to leading order, is

$$0 = -\frac{\partial p_0}{\partial x}, \tag{2.60}$$

$$0 = -\frac{\partial p_0}{\partial y} + B_y \theta_{L0}, \tag{2.61}$$

$$\frac{\partial \theta_{L0}}{\partial t} = H\nabla^2 \theta_{L0}, \tag{2.62}$$

$$\frac{\partial \theta_{S0}}{\partial t} = H\nabla^2 \theta_{S0}. \tag{2.63}$$

The boundary conditions at infinity on the temperature are unchanged,

$$\lim_{y\to-\infty} \theta_{S0}(x,y,t) = \theta_{-\infty}, \tag{2.64}$$

$$\lim_{y\to+\infty} \theta_{L0}(x,y,t) = \theta_{+\infty}. \tag{2.65}$$

The fluid velocity is assumed $O(\varepsilon)$ so it does not appear in the $O(1)$ system. No conditions are imposed on the fluid at this stage. The interface boundary conditions, again to leading order, are

$$\theta_{S0}(x, Y_0(t), t) = 0, \tag{2.66}$$

$$\theta_{L0}(x, Y_0(t), t) = 0, \tag{2.67}$$

$$\frac{\partial \theta_{S0}}{\partial y}(x, Y_0(t), t) - \frac{\partial \theta_{L0}}{\partial y}(x, Y_0(t), t) = \frac{1}{h}\frac{dY_0(t)}{dt}. \tag{2.68}$$

Note that there is no reference to either fluid velocity or the interface perturbation in the above. It describes nothing more than a propagating flat interface problem. So, before we solve for the leading order quantities, it is useful to derive the $O(\varepsilon)$ system to see how the leading order solutions will influence the solution at the next stage.

The system governing the $O(\varepsilon)$ terms is

$$\frac{\partial u_1}{\partial x} + \frac{\partial v_1}{\partial y} = 0, \tag{2.69}$$

$$\frac{\partial u_1}{\partial t} = -\frac{\partial p_1}{\partial x} + D\nabla^2 u_1, \tag{2.70}$$

$$\frac{\partial v_1}{\partial t} = -\frac{\partial p_1}{\partial y} + D\nabla^2 v_1 + B_y \theta_{L1}, \tag{2.71}$$

$$\frac{\partial \theta_{L1}}{\partial t} + u_1 \frac{\partial \theta_{L0}}{\partial x} + v_1 \frac{\partial \theta_{L0}}{\partial y} = H\nabla^2 \theta_{L1}, \tag{2.72}$$

$$\frac{\partial \theta_{S1}}{\partial t} = H\nabla^2 \theta_{S1}. \tag{2.73}$$

It is convenient to represent the flow field in terms of a stream function

$$u_1 = +\frac{\partial \psi}{\partial y}, \tag{2.74}$$

$$v_1 = -\frac{\partial \psi}{\partial x}, \tag{2.75}$$

which allows the continuity equation to be eliminated and the momentum equations to be combined. The final $O(\varepsilon)$ system is

$$\frac{\partial}{\partial t}\left(\nabla^2 \psi\right) = D\nabla^4 \psi - B_y \frac{\partial \theta_{L1}}{\partial x}, \tag{2.76}$$

$$\frac{\partial \theta_{L1}}{\partial t} + \left(\frac{\partial \psi}{\partial y}\right) \frac{\partial \theta_{L0}}{\partial x} - \left(\frac{\partial \psi}{\partial x}\right) \frac{\partial \theta_{L0}}{\partial y} = H\nabla^2 \theta_{L1}, \tag{2.77}$$

$$\frac{\partial \theta_{S1}}{\partial t} = H\nabla^2 \theta_{S1}. \tag{2.78}$$

The boundary conditions at infinity on the stream function are

$$\lim_{y \to +\infty} \psi(x, y, t) = 0, \tag{2.79}$$

$$\lim_{y \to +\infty} \frac{\partial \psi}{\partial y}(x, y, t) = 0, \tag{2.80}$$

and on the temperature perturbations

$$\lim_{y\to-\infty}\theta_{S1}(x,y,t) = 0, \tag{2.81}$$

$$\lim_{y\to+\infty}\theta_{L1}(x,y,t) = 0. \tag{2.82}$$

On the interface, the stream function must satisfy no-slip conditions, which for this problem implies

$$\psi(x,Y_0(t),t)=0, \tag{2.83}$$

$$\frac{\partial\psi}{\partial y}(x,Y_0(t),t)=0. \tag{2.84}$$

The boundary conditions on the temperature perturbations at the interface are given by

$$\theta_{S1}(x,Y_0(t),t)+Y_1(x,t)\frac{\partial\theta_{S0}}{\partial y}(x,Y_0(t),t)=\delta_0\frac{\partial^2 Y_1}{\partial x^2}, \tag{2.85}$$

$$\theta_{L1}(x,Y_0(t),t)+Y_1(x,t)\frac{\partial\theta_{L0}}{\partial y}(x,Y_0(t),t)=\delta_0\frac{\partial^2 Y_1}{\partial x^2}, \tag{2.86}$$

and

$$\begin{aligned}&\frac{\partial\theta_{S1}}{\partial y}(x,Y_0(t),t)-\frac{\partial\theta_{L1}}{\partial y}(x,Y_0(t),t)= \\ &\frac{1}{h}\frac{\partial Y_1}{\partial t}+Y_1(x,t)\left[\frac{\partial^2\theta_{S0}}{\partial y^2}(x,Y_0(t),t)-\frac{\partial^2\theta_{L0}}{\partial y^2}(x,Y_0(t),t)\right].\end{aligned} \tag{2.87}$$

It is worthwhile to examine (2.85) and (2.86) more closely. These equations result from the condition that the temperature at the interface is the corrected melting temperature. There is a contribution from the leading order solution because of the perturbed shape of the interface. It is interesting to note, however, that the discontinuity of the derivative of the leading order solution translates into a discontinuity in the values of the temperature perturbations. Subtracting (2.85) from (2.86) and using (2.68), we find

$$\theta_{L1}(x,Y_0(t),t)-\theta_{S1}(x,Y_0(t),t)=\frac{1}{h}Y_1(t)\frac{dY_0(t)}{dt}. \tag{2.88}$$

This interesting consequence of linear perturbation theory is rarely mentioned in the literature.

Now we examine the stability of a propagating wave solution of the above systems. In this case a solution to the $O(1)$ system is given by

$$Y_0 = Vt, \tag{2.89}$$

$$\theta_{S0} = 0, \tag{2.90}$$

$$\theta_{L0} = \frac{H}{h}\left[e^{-V(y-Vt)/H} - 1\right], \tag{2.91}$$

$$p_0 = \frac{HB_y}{h}\left\{\frac{H}{V}\left[1 - e^{-V(y-Vt)/H}\right] - (y - Vt)\right\}. \tag{2.92}$$

For the $O(\varepsilon)$ system, we seek a solution of the form

$$\begin{aligned} Y_1 &= e^{\sigma t}\cos(ax), & (2.93)\\ \psi &= e^{\sigma t}\sin(ax) f(y - Vt), & (2.94)\\ \theta_{S1} &= e^{\sigma t}\cos(ax) g_S(y - Vt), & (2.95)\\ \theta_{L1} &= e^{\sigma t}\cos(ax) g_L(y - Vt), & (2.96) \end{aligned}$$

which, after some manipulation, yields the following equations

$$Df'''' + Vf''' - \left(\sigma + 2Da^2\right) f'' - a^2Vf' + \left(\sigma a^2 + Da^4\right) f + B_y a g_L = 0, \tag{2.97}$$

$$Hg_L'' + Vg_L' - \left(\sigma + Ha^2\right) g_L + a\left(\frac{\partial \theta_{L0}}{\partial y}\right) f = 0, \tag{2.98}$$

$$Hg_S'' + Vg_S' - \left(\sigma + Ha^2\right) g_S = 0 \tag{2.99}$$

with boundary conditions

$$f(0) = f'(0) = f(\infty) = f'(\infty) = 0 \tag{2.100}$$

$$g_L(0) = -\delta_0 a^2 + \frac{V}{h} \tag{2.101}$$

$$g_L(\infty) = 0 \tag{2.102}$$

$$g_S(0) = -\delta_0 a^2 \tag{2.103}$$

$$g_S(-\infty) = 0 \tag{2.104}$$

$$w(\sigma) = g_S'(0) - g_L'(0) - \frac{V^2}{Hh} - \frac{\sigma}{h} = 0. \tag{2.105}$$

The linear stability of individual modes is determined by whether solutions to the above have positive or negative values of σ. An individual mode is linearly stable if $\sigma < 0$ and it is linearly unstable if $\sigma > 0$.

When the effects of buoyancy induced fluid motion are neglected (i.e., we let $B_y = 0$), Mullins and Sekerka (see [51]) demonstrated that it is possible to significantly simplify the above system:

$$f = 0, \tag{2.106}$$

$$g_L = A_L e^{-q_L(y-Vt)}, \tag{2.107}$$

$$g_S = A_S e^{q_S(y-Vt)}, \tag{2.108}$$

$$A_L = -\delta_0 a^2 + \frac{V}{h}, \tag{2.109}$$

$$A_S = -\delta_0 a^2, \tag{2.110}$$

where

$$q_L = \frac{V + \sqrt{V^2 + 4H\sigma + 4H^2a^2}}{2H}, \tag{2.111}$$

$$q_s = \frac{-V + \sqrt{V^2 + 4H\sigma + 4H^2a^2}}{2H}, \tag{2.112}$$

and the growth rate is determined by the solution of the non-linear algebraic equation

$$w(\sigma) = A_s q_s + A_L q_L - \frac{V^2}{hH} - \frac{\sigma}{h} = 0. \tag{2.113}$$

Solutions of (2.113) are $O(-a^2)$ for large a (i.e., high wave number perturbations are stable) and there is usually a range of wave numbers, $0 < a \leq a_c$ for which $\sigma > 0$. A typical solution of (2.113) is shown in Fig. 2.3.

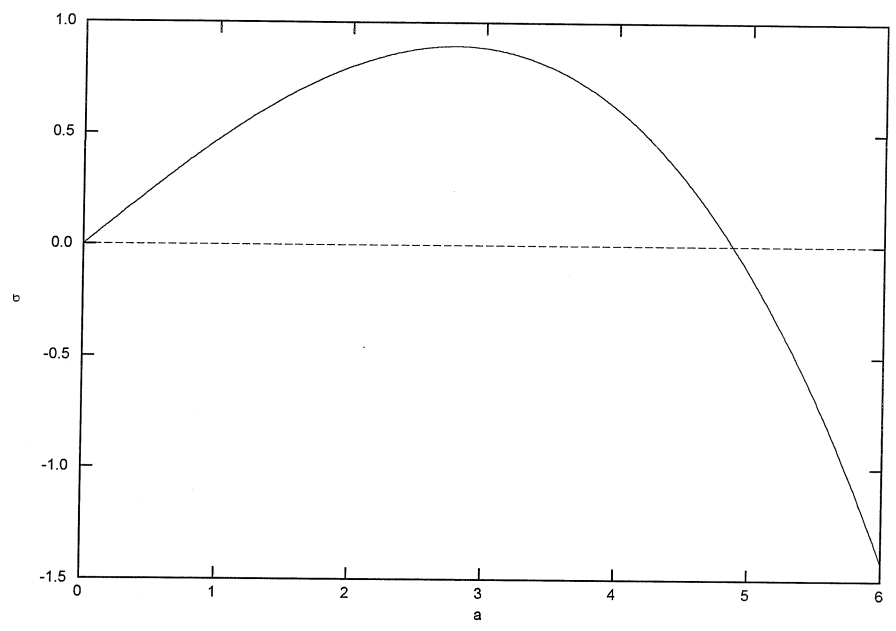

Figure 2.3: Growth rate (σ) versus wave number (a) for the $V = 1/2$ case

When fluid motion is allowed (i.e., $B_y \neq 0$), it is no longer possible to reduce (2.97)-(2.99) to algebraic equations and the system must be solved numerically.

If we consider σ known and ignore (2.105), then (2.97)-(2.99) is straightforward to solve. We discretize g_S on a truncated domain, $-L \leq y \leq 0$, using centered finite differences and take $y = -L$ to be $y = -\infty$. We also solve the coupled system for f and g_L on a truncated domain, $0 \leq y \leq L$, using centered finite differences and take $y = L$ to be $y = \infty$. This yields a value of $w(\sigma)$ which will most likely violate (2.105), so we must iterate on σ until the entire system is satisfied. We have found that the secant method is efficient for determining the growth rate, σ, given values for a and B_y.

In Fig. 2.4 we have plotted the dependence of σ on B_y for $a = 1$, $V = 1/2$ and unit material properties. The basic trend in Fig. 2.4 is that the growth rate is increased when the dendrite grows parallel with gravity and decreased when the dendrite grows anti-parallel to $\mathbf{g}$. This is in agreement with experimental observation (see [21]). In Figures 2.5-2.8 we have plotted the spatial variation for all the field variables for

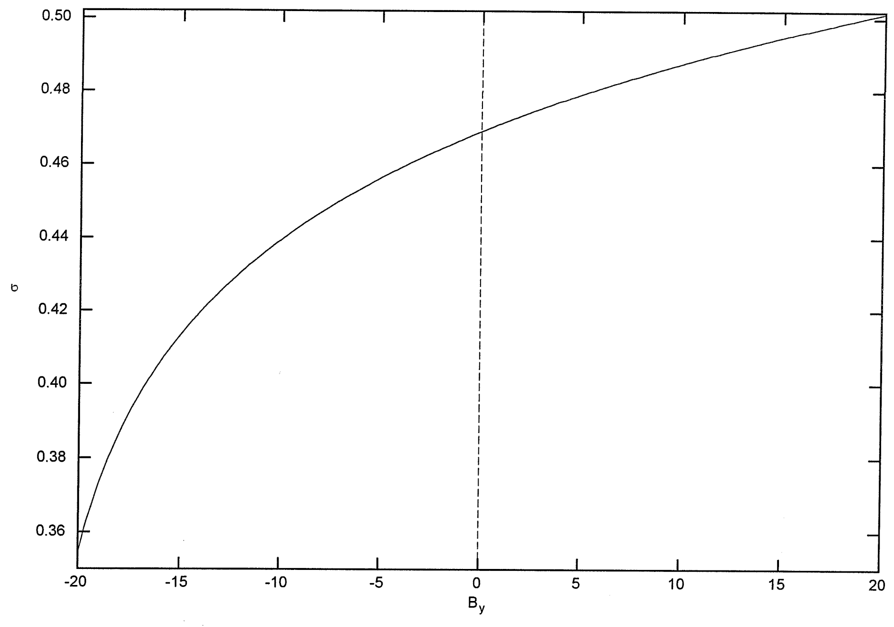

Figure 2.4: Dependence of σ on B_y for $a = 1$

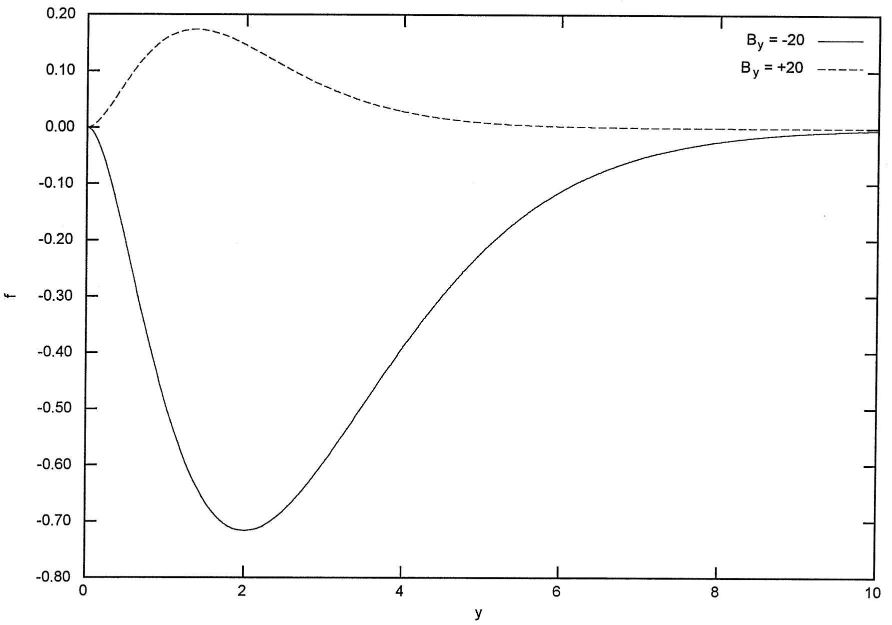

Figure 2.5: Spatial variation of f for $B_y = -20$ and $B_y = +20$

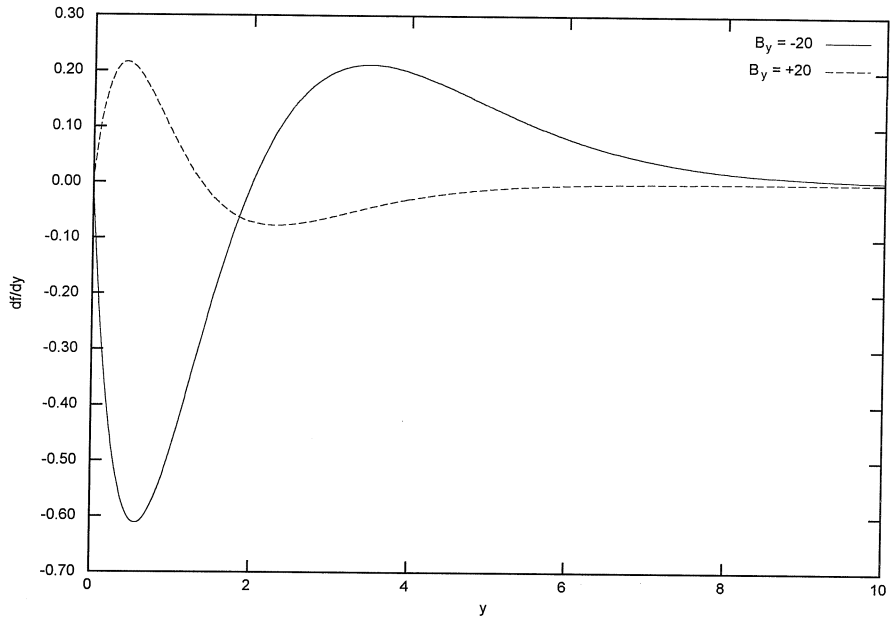

Figure 2.6: Spatial variation of f' for $B_y = -20$ and $B_y = +20$

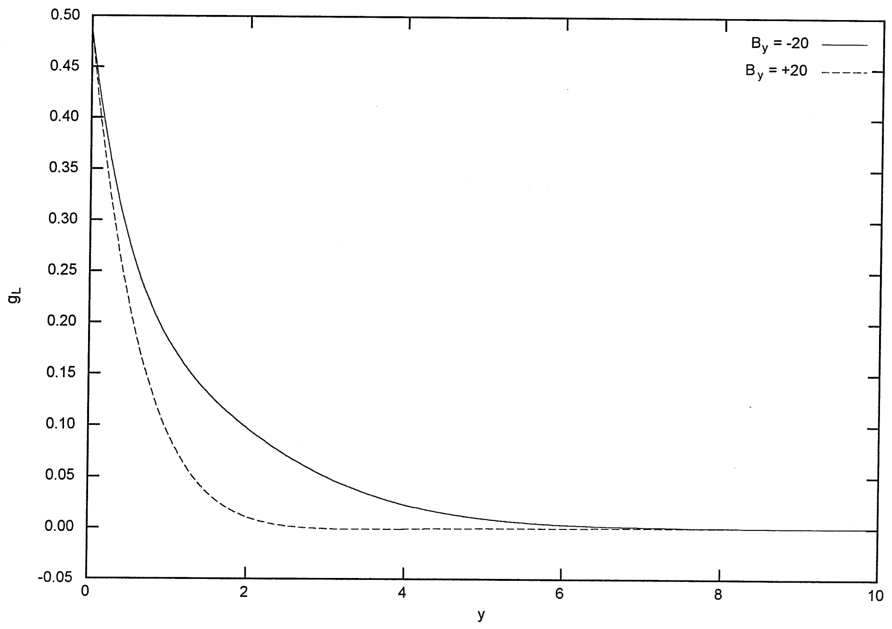

Figure 2.7: Spatial variation of g_L for $B_y = -20$ and $B_y = +20$

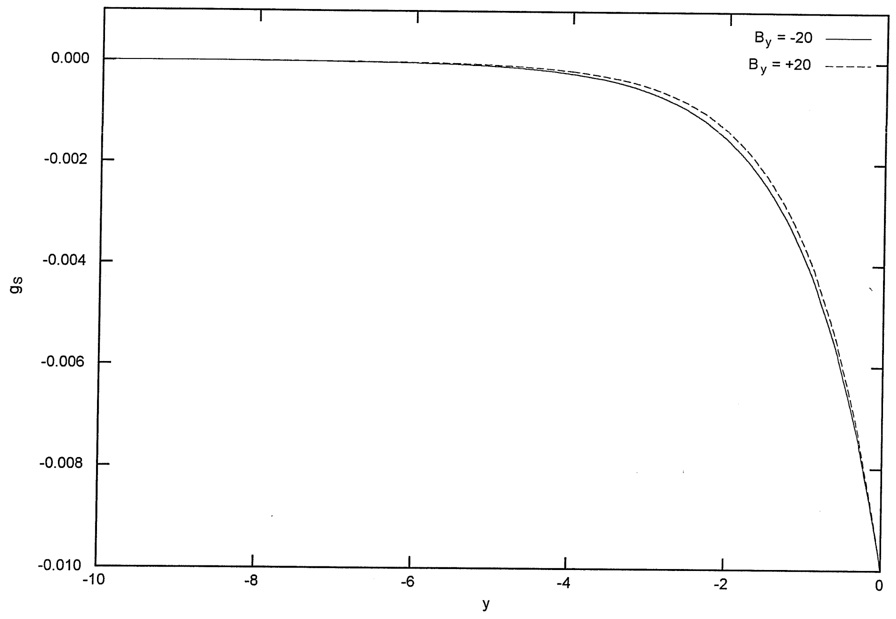

Figure 2.8: Spatial variation of g_S for $B_y = -20$ and $B_y = +20$

$a = 1$, $V = 1/2$ and $B_y \pm 20$. It is not a coincidence that the growth rate deviates more from its pure diffusion value and that the flow is stronger when $B_y < 0$. We will discuss this in Chapter 6.

Chapter 3 Discretization of non-smooth functions

3.1 Introduction

The discretization of the diffusion equation in dendritic solidification problems is complicated by the discontinuities in the temperature derivatives across the interface. There are many different approaches that researchers use to solve problems featuring non-smooth solutions across an interface. These include boundary integral/element methods, finite element methods, domain mapping methods and fixed grid approaches. We will discuss all of these techniques later in Chapters 4, 5 and 6. In this chapter we focus our attention on the development of a fixed grid approach that can accurately discretize non-smooth problems.

Our goal here is to develop accurate finite difference stencils that can be used to discretize the derivatives of a function with jumps in its value and/or derivatives across an interface (i.e., a point in one dimension or curve in two dimensions). We assume that the interface is completely embedded (i.e., contained) in the computational domain. In many problems, including dendritic solidification, the value of the solution is prescribed on the embedded interface and simple "cut" finite difference stencils can be employed (see [28]). One difficulty with this approach is that the regularity of the computational mesh is broken by the modified stencils adjacent to the interface. This eliminates the use of so-called "fast solvers" (see [68]) and leads to discrete systems that can be difficult and expensive to invert. A further difficulty is that cut stencils are difficult to apply to problems that satisfy non-Dirchlet boundary conditions on the interface. As a consequence, researchers have attempted to develop alternate approaches. The most significant of these, for our purposes, is the immersed interface method (see [38], [43] and [78]). The immersed interface method was originally developed to discretize elliptic problems (see [38], [40], [43] and [27]). It has since been extended to solve acoustic problems in heterogeneous media (see [81]), one-dimensional parabolic problems with moving boundaries (see [41] and [42]) and incompressible Stokes flow with moving boundaries (see [39]). In this work, we show how a modified version of the method may be used to simulate dendritic solidification problems with and without convection and to compute incompressible flow through irregular regions.

The immersed interface method employs carefully constructed finite difference stencils which allow the accurate calculation of the derivatives of non-smooth functions. A regular finite difference grid and standard second order accurate difference formulas are used at nodes whose stencils do not span the interface (i.e., stencils that use nodes which lie on only one side of the interface). At the nodes whose regular stencils do span the interface, custom difference formulas are determined. The resulting stencils span the interface but they are derived in such a way that accurate estimates of the derivatives are determined. The derivation of these difference formulas, as presented in [38], is quite involved and requires, among other things, a coordinate transformation if we are solving a partial differential equation. The basic idea is to use knowledge of the jumps in the function value and its derivatives at a single point on the interface near the node to make the truncation error of the derived stencil sufficiently small (see [38] for details). The resulting stencils allow the accurate discretization of problems with non-smooth solutions on regular finite difference grids.

In this chapter we will generalize the immersed interface method. Our derivation, while still somewhat involved, does not require any coordinate transforms be applied when partial differential equations are being discretized. Further, our stencils typically use multiple points on the interface in their derivation which yields stronger coupling between the behavior of the function at the interface and it derivatives at the grid nodes. Working in both one and two dimensions, we will focus on the derivation of second order accurate stencils. The general formulas we develop, however, apply to stencils of any order. Finally, we will present a one-step time marching scheme which is capable of accurately evolving functions that are subject to discontinuities in their time derivative. The scheme is notable in that it allows the individual terms in the governing equation to be discretized differently which can be convenient for non-linear problems.

3.2 Spatial derivatives in one dimension

In this section we will derive finite difference stencils to accurately calculate the derivatives of a function, $u(x)$, whose value and/or derivatives jumps discontinuously at a single point, $x = X$, called the interface. We assume that the function is sufficiently smooth away from the interface that standard finite difference stencils yield accurate approximations to the derivatives whenever all the nodes in the stencil lie to left or right of X. Suppose, however, that we wish to calculate the derivative of u at $x = x_i$ and $x = x_{i+1}$ where $x_{i+1} - x_i = \Delta x$ and these nodes are separated by the interface, $x_i < X < x_{i+1}$. We shall demonstrate shortly that naively applying the standard first order accurate finite difference formulas,

$$u_x(i) = \frac{u(i+1) - u(i)}{\Delta x} + O(\Delta x), \tag{3.1}$$

$$u_x(i+1) = \frac{u(i+1) - u(i)}{\Delta x} + O(\Delta x), \tag{3.2}$$

can yield an approximation to the derivative with an error as large as $O(\Delta x^{-1})$. Unfortunately, it is impossible to improve the above approximations without more detailed knowledge of how the function behaves at the interface. In our formulation we require that jump conditions, which specify a relationship between the derivatives of the function on each side of the interface, be known.

Suppose for example we know that u satisfies the following jump conditions:

$$[u] = u^+ - u^-, \tag{3.3}$$

$$[u_x] = u_x^+ - u_x^-, \tag{3.4}$$

where

$$\begin{aligned} u^{+} &= \lim_{\varepsilon\to 0^{+}} u\left(X+\varepsilon\right), && (3.5)\\ u^{-} &= \lim_{\varepsilon\to 0^{+}} u\left(X-\varepsilon\right), && (3.6)\\ u_x^{+} &= \lim_{\varepsilon\to 0^{+}} \frac{du\left(X+\varepsilon\right)}{dx}, && (3.7)\\ u_x^{-} &= \lim_{\varepsilon\to 0^{+}} \frac{du\left(X-\varepsilon\right)}{dx}, && (3.8)\end{aligned}$$

and the values of $[u]$ and $[u_x]$ are known. Before we proceed it is important to emphasize that the above jump conditions simply state that the jumps or discontinuities of the function value and first derivative are known. The actual value of the function or its first derivative on either side of the interface can not be deduced from this information. Only the difference between the function values or derivatives is specified. This knowledge is sufficient, however, to construct accurate finite difference stencils as we now demonstrate.

Finite difference stencils are typically derived using Taylor expansions. For instance, to calculate $u_x\left(i\right)$, expand $u\left(i+1\right)$ in a Taylor series about x_i,

$$u\left(i+1\right) = u\left(i\right) + \left(\Delta x\right) u_x\left(i\right) + O\left(\Delta x^2\right), \tag{3.9}$$

and solve for $u_x\left(i\right)$,

$$u_x\left(i\right) = \frac{u\left(i+1\right)-u\left(i\right)}{\Delta x} + O\left(\Delta x\right). \tag{3.10}$$

The above derivation fails, however, if the interface lies between x_{i+1} and x_i because it assumes u is continuously differentiable everywhere in this interval. This is why standard finite difference stencils yield poor results when used to differentiate non-smooth functions. The derivations can be modified, however, to account for the presence of the interface. We have assumed that u is continuously differentiable everywhere to the left (where x_i is located) and right (where x_{i+1} is located) of the interface. So, even though it is not possible to expand $u\left(i+1\right)$ about $x = x_i$, it is possible to expand it about any point to the right of the interface. The most convenient choice is a point arbitrarily close to the interface but still on the right-hand side, $x = X^{+}$, which yields

$$u\left(i+1\right) = u^{+} + \left(x_{i+1} - X\right) u_x^{+} + O\left(\Delta x^2\right). \tag{3.11}$$

We can then express $u\left(i+1\right)$ in terms of quantities on the other side of the interface by substituting (3.3) and (3.4) into (3.11),

$$u\left(i+1\right) = u^{-} + \left(x_{i+1} - X\right) u_x^{-} + [u] + \left(x_{i+1} - X\right)[u_x] + O\left(\Delta x^2\right). \tag{3.12}$$

Noting that u is continuously differentiable between x_i and X^{-}, we can use the expansions

$$\begin{aligned} u^{-} &= u\left(i\right) + \left(X - x_i\right) u_x\left(i\right) + O\left(\Delta x^2\right), & (3.13)\\ u_x^{-} &= u_x\left(i\right) + O\left(\Delta x\right), & (3.14) \end{aligned}$$

to obtain

$$u\left(i+1\right) = u\left(i\right) + \left(\Delta x\right) u_x\left(i\right) + [u] + \left(x_{i+1} - X\right)[u_x] + O\left(\Delta x^2\right). \tag{3.15}$$

This is nothing more than the standard Taylor series expansion plus an additive correction term which is a function of known quantities ($[u]$, $[u_x]$ and X). Solving the above for the derivative, we find

$$u_x\left(i\right) = \frac{u\left(i+1\right) - u\left(i\right)}{\Delta x} - \frac{[u] + \left(x_{i+1} - X\right)[u_x]}{\Delta x} + O\left(\Delta x\right), \tag{3.16}$$

which is merely the standard finite difference stencil augmented by an additive correction. Using an analogous expansion of $u\left(i\right)$ about $x = X^{-}$, we find

$$u_x\left(i+1\right) = \frac{u\left(i+1\right) - u\left(i\right)}{\Delta x} - \frac{[u] + \left(x_i - X\right)[u_x]}{\Delta x} + O\left(\Delta x\right). \tag{3.17}$$

We can verify that this is correct by testing it on

$$u = \begin{cases} a + bx & x < X \\ \alpha + \beta x & x \geq X \end{cases}, \tag{3.18}$$

where, in this case, the known jumps are

$$[u] = \alpha - a + (\beta - b) X, \tag{3.19}$$

$$[u_x] = \beta - b. \tag{3.20}$$

A first order accurate stencil for the first derivative should yield the exact value. Assuming $x_i < X \leq x_{i+1}$, the standard finite difference stencil, (3.10), yields either

$$\frac{u(i+1) - u(i)}{\Delta x} = b + \frac{[u] + (x_{i+1} - X)[u_x]}{\Delta x}, \tag{3.21}$$

or

$$\frac{u(i+1) - u(i)}{\Delta x} = \beta + \frac{[u] + (x_i - X)[u_x]}{\Delta x}, \tag{3.22}$$

after some manipulation, which verifies both that (3.16) and (3.17) are correct and that naive use of the standard stencil can yield an error as large as $O(\Delta x^{-1})$ for this problem.

We found above that it was possible to calculate the derivative of a non-smooth function using a simple additive correction to the standard finite difference formulas. Unfortunately, this is not always the case. Suppose, for example, we wanted to calculate the derivative of a function that satisfied jump conditions of the form

$$[u] = u^+ - u^-, \tag{3.23}$$

$$[hu_x] = h^+ u_x^+ - h^- u_x^-, \tag{3.24}$$

where the values of $[u]$, $[hu_x]$, h^+ and h^- are known. Jump conditions of this form arise for instance in dendritic solidification when the material properties in the solid and liquid phases are different. As before, we wish to compute the derivative of u at

x_i and x_{i+1} where the interface lies between these nodes, $x_i < X \leq x_{i+1}$. Following exactly the same procedure we used above, we expand $u(i+1)$ about $x = X^+$,

$$u(i+1) = u^+ + (x_{i+1} - X) u_x^+ + O\left(\Delta x^2\right). \tag{3.25}$$

Using the jump conditions to express this in terms of quantities evaluated on the other side of the interface, we obtain

$$u(i+1) = u^- + \frac{h^-}{h^+}(x_{i+1} - X) u_x^- + [u] + \frac{(x_{i+1} - X)}{h^+}[hu_x] + O\left(\Delta x^2\right). \tag{3.26}$$

Expanding the derivatives at the interface about x_i yields

$$\begin{aligned} u^- &= u(i) + (X - x_i) u_x(i) + O\left(\Delta x^2\right), && (3.27)\\ u_x^- &= u_x(i) + O(\Delta x). && (3.28) \end{aligned}$$

Substituting these into (3.26) and collecting terms, we find

$$\begin{aligned} u(i+1) &= u(i) + \left[\frac{h^+(X - x_i) + h^-(x_{i+1} - X)}{h^+}\right] u_x(i) \\ &\quad + [u] + \frac{(x_{i+1} - X)}{h^+}[hu_x] + O\left(\Delta x^2\right), \end{aligned} \tag{3.29}$$

which, unlike the previous case, is quite different from the standard Taylor expansion. Solving this for the derivative, we find

$$u_x(i) = h^+\left[\frac{u(i+1) - u(i)}{h^+(X - x_i) + h^-(x_{i+1} - X)}\right] - \left(\frac{h^+[u] + (x_{i+1} - X)[hu_x]}{h^+(X - x_i) + h^-(x_{i+1} - X)}\right). \tag{3.30}$$

Expanding $u(i)$ about $x = X^-$ and following an analogous procedure, we find

$$u_x(i+1) = h^-\left[\frac{u(i+1) - u(i)}{h^+(X - x_i) + h^-(x_{i+1} - X)}\right] - \left(\frac{h^-[u] + (x_i - X)[hu_x]}{h^+(X - x_i) + h^-(x_{i+1} - X)}\right). \tag{3.31}$$

Note that, in this case, the stencil that must be used to calculate the first derivative is not simply an additive correction of the standard stencil, but a stencil specific to the exact geometry of the interface and the jump conditions we started with. It is quite easy to verify that the above stencils exactly compute the derivative of a piecewise linear polynomial satisfying (3.23) and (3.24) so we will not bother to present the calculation here.

The lesson to take away from the above two examples is that it is possible to accurately compute the derivative of a piecewise smooth function even when differencing across the interface. We now develop general formulas for stencils that can accurately compute the derivatives of non-smooth functions. As we found above, to construct such stencils, it is necessary to have information about how the function behaves across the interface. We assume that jump conditions of the form

$$\frac{d^p u^+}{dx^p} = \sum_{k=0}^{p} D_k^{p-} \frac{d^k u^-}{dx^k} + J_p^-, \tag{3.32}$$

and

$$\frac{d^p u^-}{dx^p} = \sum_{k=0}^{p} D_k^{p+} \frac{d^k u^+}{dx^k} + J_p^+, \tag{3.33}$$

are known, where we require

$$D_k^{p+} = D_k^{p-} = 0 \quad \text{if } k > p. \tag{3.34}$$

To give a concrete example, the jump conditions of the above form for the second derivative would be

$$u_{xx}^+ = D_2^{2-} u_{xx}^- + D_1^{2-} u_x^- + D_0^{2-} u^- + J_2^-, \tag{3.35}$$

and

$$u_{xx}^- = D_2^{2+} u_{xx}^+ + D_1^{2+} u_x^+ + D_0^{2+} u^+ + J_2^+, \tag{3.36}$$

where the values of the coefficients D_2^{2-}, D_1^{2-},..., D_1^{2+}, D_0^{2+} are known as are the values of the "jumps" J_2^- and J_2^+. Note that it would be incorrect to identify J_2^- or J_2^+ with $[u_{xx}]$ in the general case as other derivatives are also present. While this notation may seem somewhat awkward, it is necessary to allow the generality we wish. We now develop general stencil formulas of arbitrary order for the calculation of the derivatives of a piecewise smooth function whose values and/or derivatives may be discontinuous at the interface.

Suppose we wish to calculate the derivatives of a function u that is sufficiently smooth to the left and right of an interface located at $x = X$. We assume that jump conditions of the form (3.32) and (3.33) are known for u. The stencils we construct will use $N+1$ nodes located at $x = x_\sigma$, $0 \leq \sigma \leq N$, and approximate the derivatives at the location $x = x_0$. We will denote the function evaluated at the $x = x_\sigma$ node by $u(\sigma)$. As a demonstration of the general procedure, we begin by calculating the stencil coefficients for a stencil that is not intersected by the interface (i.e., all of its nodes lie to the left or right of X).

To determine an $N+1$ node stencil that approximates the derivatives of a smooth function at $x = x_0$, we expand the function at each node about $x = x_0$

$$u(\sigma) = \sum_{n=0}^{N} \left[\frac{(x_\sigma - x_0)^n}{n!} \right] \frac{d^n u(0)}{dx^n} + O\left((x_\sigma - x_0)^{N+1} \right). \tag{3.37}$$

Put in more general terms, we have

$$u(\sigma) - C_\sigma = \sum_{n=0}^{N} \omega_{n\sigma} \frac{d^n u(0)}{dx^n} + O\left((x_\sigma - x_0)^{N+1} \right), \tag{3.38}$$

where

$$\omega_{n\sigma} = \frac{(x_\sigma - x_0)^n}{n!}, \tag{3.39}$$

and

$$C_\sigma = 0. \tag{3.40}$$

We wish to determine stencils to approximate the m^{th} derivative $(0 \leq m \leq N)$ of the form

$$\frac{d^m u(0)}{dx^m} = \sum_{\sigma=0}^{N} \alpha_{\sigma m} \left(u(\sigma) - C_\sigma \right). \tag{3.41}$$

Neglecting the error term and substituting (3.38) into (3.41) yields

$$\frac{d^m u(0)}{dx^m} = \sum_{\sigma=0}^{N}\sum_{n=0}^{N} \omega_{n\sigma}\alpha_{\sigma m}\frac{d^n u(0)}{dx^n}, \tag{3.42}$$

or

$$\sum_{n=0}^{N}\left(\sum_{\sigma=0}^{N}\omega_{n\sigma}\alpha_{\sigma m} - \delta_{nm}\right)\frac{d^n u(0)}{dx^n} = 0, \tag{3.43}$$

where the Kronecker delta is defined by

$$\delta_{nm} = \begin{cases} 1 & n = m \\ 0 & n \neq m \end{cases}. \tag{3.44}$$

Noting that the value of the derivatives are arbitrary, this implies

$$\sum_{\sigma=0}^{N}\omega_{n\sigma}\alpha_{\sigma m} - \delta_{nm} = 0, \tag{3.45}$$

which, when written in matrix form, is

$$\boldsymbol{\omega\alpha} = \mathbf{I}. \tag{3.46}$$

Thus the stencil coefficients are determined by

$$\boldsymbol{\alpha} = \boldsymbol{\omega}^{-1}. \tag{3.47}$$

For a concrete example, consider computing the 3 node stencil ($N = 2$) for a smooth function associated with the nodes located at $x_0 = x$, $x_1 = x - \Delta x$ and $x_2 = x + \Delta x$. In this case, we find

$$\boldsymbol{\omega} = \begin{bmatrix} 1 & 1 & 1 \\ 0 & -\Delta x & \Delta x \\ 0 & \dfrac{\Delta x^2}{2} & \dfrac{\Delta x^2}{2} \end{bmatrix}, \tag{3.48}$$

which corresponds to the stencil coefficients

$$\boldsymbol{\alpha} = \begin{bmatrix} 1 & 0 & \frac{-2}{\Delta x^2} \\ 0 & \frac{-1}{2\Delta x} & \frac{1}{\Delta x^2} \\ 0 & \frac{1}{2\Delta x} & \frac{1}{\Delta x^2} \end{bmatrix}, \tag{3.49}$$

or, using (3.41), the standard finite difference formulas

$$\frac{du(0)}{dx} = \frac{u(2) - u(1)}{2\Delta x}, \tag{3.50}$$

$$\frac{d^2u(0)}{dx^2} = \frac{u(2) - 2u(0) + u(1)}{\Delta x^2}. \tag{3.51}$$

Suppose we now need to accurately differentiate a function at a node whose stencil spans the interface (i.e., has nodes that lie on both sides). Moving back to the more general setting, we know that we can use a Taylor series expansion to express $u(\sigma)$ in terms of $u(0)$ and its derivatives if the nodes x_0 and x_σ are not separated by the interface (i.e., both lie on the same side of the interface). From (3.38), this implies that

$$\omega_{n\sigma} = \frac{(x_\sigma - x_0)^n}{n!}, \tag{3.52}$$

and

$$C_\sigma = 0 \tag{3.53}$$

for all σ such that x_0 and x_σ are on the same side of the interface. We still wish to express nodes separated from x_0 in terms of $u(0)$ and its derivatives, but this can not be done directly. Following the approach we used at the beginning of this chapter, we start by expressing $u(\sigma)$ in terms of values and derivatives evaluated on the same side of the interface as x_σ. That is, if $x_\sigma > X$, then we use

$$u(\sigma) = \sum_{p=0}^{N} \left[\frac{(x_\sigma - X)^p}{p!}\right] \frac{d^p u^+}{dx^p} + O\left((x_\sigma - X)^{N+1}\right) \tag{3.54}$$

and if $x_\sigma < X$ we use

$$u(\sigma) = \sum_{p=0}^{N} \left[\frac{(x_\sigma - X)^p}{p!}\right] \frac{d^p u^-}{dx^p} + O\left((x_\sigma - X)^{N+1}\right). \tag{3.55}$$

To simplify the presentation we will assume $x_\sigma > X$ from now on. Once the process is understood, the extension to the $x_\sigma < X$ case is trivial. We can express $u(\sigma)$ in terms of quantities evaluated on the other side of the interface by using the jump conditions. Substituting (3.32) into (3.54) we find

$$\begin{aligned} u(\sigma) &= \sum_{p=0}^{N} \sum_{k=0}^{p} D_k^{p-} \left[\frac{(x_\sigma - X)^p}{p!}\right] \frac{d^k u^-}{dx^k} + \sum_{p=0}^{N} \left[\frac{(x_\sigma - X)^p}{p!}\right] J_p^- \\ &\quad + O\left((x_\sigma - X)^{N+1}\right). \end{aligned} \tag{3.56}$$

The derivatives at the interface can be approximated using a Taylor series expansion about $x = x_0$,

$$\frac{d^k u^-}{dx^k} = \sum_{n=k}^{N} \left[\frac{(X - x_0)^{n-k}}{(n-k)!}\right] \frac{d^n u(0)}{dx^n} + O\left((X - x_0)^{N+1-k}\right).$$

Substituting this into (3.56) yields

$$\begin{aligned} u(\sigma) &= \sum_{p=0}^{N} \sum_{k=0}^{p} \sum_{n=k}^{N} D_k^{p-} \left[\frac{(x_\sigma - X)^p}{p!}\right] \left[\frac{(X - x_0)^{n-k}}{(n-k)!}\right] \frac{d^n u(0)}{dx^n} \\ &\quad + \sum_{p=0}^{N} \left[\frac{(x_\sigma - X)^p}{p!}\right] J_p^- + O(E), \end{aligned} \tag{3.57}$$

where the error is given by

$$E = \sum_{p=0}^{N} \sum_{k=0}^{p} D_k^{p-} \left[\frac{(x_\sigma - X)^p}{p!}\right] [X - x_0]^{N+1-k}. \tag{3.58}$$

This can be bounded easily, assuming $D_k^{p-} = O(1)$, by noting

$$D_k^{p-} \left[\frac{(x_\sigma - X)^p}{p!}\right] [X - x_0]^{N+1-k} = O\left(D_k^{p-} (x_\sigma - x_0)^{N+1+p-k}\right), \tag{3.59}$$

which, from (3.34), implies

$$E = O\left((x_\sigma - x_0)^{N+1}\right). \tag{3.60}$$

Although we managed, in (3.57), to express $u(\sigma)$ in terms of $u(0)$ and its derivatives, it is not yet in a form from which we can easily extract the value of $\omega_{n\sigma}$. To achieve this, we use (3.34) to extend the sum over k to run from 0 to N which allows the sums over k and n to be easily exchanged yielding

$$\begin{aligned} u(\sigma) &= \sum_{n=0}^{N}\left\{\sum_{p=0}^{N}\sum_{k=0}^{n} D_k^{p-}\left[\frac{(x_\sigma - X)^p}{p!}\right]\left[\frac{(X-x_0)^{n-k}}{(n-k)!}\right]\right\}\frac{d^n u(0)}{dx^n} \\ &\quad + \sum_{p=0}^{N}\left[\frac{(x_\sigma - X)^p}{p!}\right] J_p^- + O\left((x_\sigma - x_0)^{N+1}\right). \end{aligned} \tag{3.61}$$

The number of terms in the inner sum can be reduced slightly by using (3.34) again, which allow us to write

$$\begin{aligned} u(\sigma) &= \sum_{n=0}^{N}\left\{\sum_{k=0}^{n}\sum_{p=k}^{N} D_k^{p-}\left[\frac{(x_\sigma - X)^p}{p!}\right]\left[\frac{(X-x_0)^{n-k}}{(n-k)!}\right]\right\}\frac{d^n u(0)}{dx^n} \\ &\quad + \sum_{p=0}^{N}\left[\frac{(x_\sigma - X)^p}{p!}\right] J_p^- + O\left((x_\sigma - x_0)^{N+1}\right). \end{aligned} \tag{3.62}$$

Comparing (3.62) and (3.38) we find

$$\omega_{n\sigma} = \sum_{k=0}^{n}\sum_{p=k}^{N} D_k^{p-}\left[\frac{(x_\sigma - X)^p}{p!}\right]\left[\frac{(X-x_0)^{n-k}}{(n-k)!}\right], \tag{3.63}$$

and

$$C_\sigma = \sum_{p=0}^{N}\left[\frac{(x_\sigma - X)^p}{p!}\right] J_p^-. \tag{3.64}$$

If the above analysis is repeated for the case when $x_\sigma < X$, we obtain identical results save for the replacement of all "-" signs with "+" signs. The general expression for the expansion coefficients, $\omega_{n\sigma}$, and additive corrections, C_σ, is given by

$$\omega_{n\sigma} = \begin{cases} \sum_{k=0}^{n}\sum_{p=k}^{N} D_k^{p+}\left[\frac{(x_\sigma - X)^p}{p!}\right]\left[\frac{(X - x_0)^{n-k}}{(n-k)!}\right] & \text{if } x_0 \geq X \text{ and } x_\sigma < X \\ \sum_{k=0}^{n}\sum_{p=k}^{N} D_k^{p-}\left[\frac{(x_\sigma - X)^p}{p!}\right]\left[\frac{(X - x_0)^{n-k}}{(n-k)!}\right] & \text{if } x_0 < X \text{ and } x_\sigma \geq X \\ \sum_{n=0}^{N}\left[\frac{(x_\sigma - x_0)^n}{n!}\right] & \text{otherwise} \end{cases} \tag{3.65}$$

and

$$C_\sigma = \begin{cases} \sum_{n=0}^{N}\left[\frac{(x_\sigma - X)^n}{n!}\right] J_n^+ & \text{if } x_0 \geq X \text{ and } x_\sigma < X \\ \sum_{n=0}^{N}\left[\frac{(x_\sigma - X)^n}{n!}\right] J_n^- & \text{if } x_0 < X \text{ and } x_\sigma \geq X \\ 0 & \text{otherwise} \end{cases} . \tag{3.66}$$

The general procedure required to discretize the derivatives at x_0 is

1. Using (3.65), set the columns of $\boldsymbol{\omega}$ associated with each stencil node.
2. Using (3.66), set the values of the additive corrections, C_σ, associated with each stencil node.
3. Determine the stencil coefficients using $\boldsymbol{\alpha} = \boldsymbol{\omega}^{-1}$.
4. Compute the value of the m^{th} derivative at x_0 using

$$\frac{d^m u(0)}{dx^m} = \sum_{\sigma=0}^{N} \alpha_{\sigma m}\left(u(\sigma) - C_\sigma\right). \tag{3.67}$$

If the nodal values are unknown and the above is to be incorporated into a system to determine their values, the additive correction, $\sum_{\sigma=0}^{N} \alpha_{\sigma m} C_\sigma$, is incorporated into the system forcing. We will see some examples of this later.

Once the expansion coefficients $\omega_{n\sigma}$ and additive corrections C_σ have been determined, it is possible to compute the stencil coefficients $\alpha_{m\sigma}$ such that the derivatives of the function can be approximated:

$$\frac{d^m u(0)}{dx^m} = \sum_{\sigma=0}^{N} \alpha_{\sigma m} \left(u(\sigma) - C_\sigma\right) + O(F), \tag{3.68}$$

with error F. The order of the error can be heuristically established as follows. Note from (3.65) that $\omega_{n\sigma} = O(\Delta x^n)$, where Δx is average distance between adjacent nodes. This implies that we can re-scale the expansion coefficients using a diagonal matrix,

$$\begin{bmatrix} 1 & & & & & \\ & \Delta x & & & & \\ & & \Delta x^2 & & & \\ & & & \cdot & & \\ & & & & \cdot & \\ & & & & & \Delta x^N \end{bmatrix} \hat{\boldsymbol{\omega}} \boldsymbol{\alpha} = \mathbf{I}, \tag{3.69}$$

so that all the entries of $\hat{\omega}_{n\sigma}$ are $O(1)$. Solving for the stencil coefficients we find

$$\boldsymbol{\alpha} = \hat{\boldsymbol{\omega}}^{-1} \begin{bmatrix} 1 & & & & & \\ & \Delta x^{-1} & & & & \\ & & \Delta x^{-2} & & & \\ & & & \cdot & & \\ & & & & \cdot & \\ & & & & & \Delta x^{-N} \end{bmatrix}. \tag{3.70}$$

Since the entries of $\hat{\boldsymbol{\omega}}^{-1}$ are assumed to be $O(1)$, this implies that $\alpha_{\sigma m} = O(\Delta x^{-m})$, which simply says that the coefficients of the stencil used to calculate the m^{th} derivative will be $O(\Delta x^{-m})$. Noting that the expansions for $u(\sigma) - C_\sigma$ include derivatives out to order N, the first neglected derivative will contribute to the error term. Since the coefficient of this term is $O\left(\Delta x^{N+1}\right)$, we have

$$\frac{d^m u(0)}{dx^m} = \sum_{\sigma=0}^{N} \alpha_{\sigma m} \left(u(\sigma) - C_\sigma\right) + O\left(\Delta x^{N+1-m}\right). \tag{3.71}$$

Thus, if we have a three point stencil ($N = 2$), the stencil for the first derivative will be second order accurate while the stencil for the second derivative, in general, will only be first order accurate.

It is interesting to examine (3.65) and (3.66) for the simple case

$$D_k^{p-} = D_k^{p+} = \delta_{pk}, \tag{3.72}$$

which corresponds to jump conditions

$$\frac{d^p u^+}{dx^p} = \frac{d^p u^-}{dx^p} + J_p^- \tag{3.73}$$

and

$$\frac{d^p u^-}{dx^p} = \frac{d^p u^+}{dx^p} + J_p^+ . \tag{3.74}$$

For this special case, the expansion coefficients for nodes separated from x_0 by the interface are given by

$$\omega_{n\sigma} = \sum_{k=0}^{n} \left[\frac{(x_\sigma - X)^k}{k!} \right] \left[\frac{(X - x_0)^{n-k}}{(n-k)!} \right] = \frac{(x_\sigma - x_0)^n}{n!} \tag{3.75}$$

while the additive correction terms are given by

$$C_\sigma = \sum_{n=0}^{N} \left[\frac{(x_\sigma - X)^n}{n!} \right] J_n^-, \tag{3.76}$$

or

$$C_\sigma = \sum_{n=0}^{N} \left[\frac{(x_\sigma - X)^n}{n!} \right] J_n^+ . \tag{3.77}$$

Jump conditions of the form (3.73) and (3.74) are called symmetric jump conditions because they give exactly the same information. In the symmetric case, only one jump condition per derivative is required, and we write this as

$$\frac{d^p u^+}{dx^p} = \frac{d^p u^-}{dx^p} + \left[\frac{d^p u}{dx^p} \right], \tag{3.78}$$

where the value of the jump of the derivative, $\left[\frac{d^p u}{dx^p}\right] = J_p^- = -J_p^+$, is assumed to be known. For the symmetric case the expansion coefficients are given by

$$\omega_{n\sigma} = \left[\frac{(x_\sigma - x_0)^n}{n!}\right], \tag{3.79}$$

regardless of the position of the nodes relative to the interface and the additive corrections are given by

$$C_\sigma = \begin{cases} -\sum_{n=0}^{N}\left[\frac{(x_\sigma - X)^n}{n!}\right]\left[\frac{d^n u}{dx^n}\right] & \text{if } x_0 \geq X \text{ and } x_\sigma < X \\ +\sum_{n=0}^{N}\left[\frac{(x_\sigma - X)^n}{n!}\right]\left[\frac{d^n u}{dx^n}\right] & \text{if } x_0 < X \text{ and } x_\sigma \geq X \\ 0 & \text{otherwise} \end{cases} . \tag{3.80}$$

It is clear, for non-symmetric jump conditions, that it is necessary to explicitly calculate $\omega_{\sigma n}$, C_σ and $\alpha_{\sigma m}$ in order to accurately approximate the derivatives near the interface. For symmetric jump conditions, however, it is only necessary to compute C_σ. Recall that the expansion coefficients are simply the standard Taylor expansion coefficients, thus

$$u(\sigma) - C_\sigma = \sum_{n=0}^{N}\left[\frac{(x_\sigma - x_0)^n}{n!}\right]\frac{d^n u(0)}{dx^n}. \tag{3.81}$$

From the theory of finite differences and (3.71), we know that when a Taylor series of the form (3.81) is substituted into

$$Q = \frac{d^m u(0)}{dx^m} - \sum_{\sigma=0}^{N}\alpha_{\sigma m}\sum_{n=0}^{N}\left[\frac{(x_\sigma - x_0)^n}{n!}\right]\frac{d^n u(0)}{dx^n} \tag{3.82}$$

and the stencil coefficients, $\alpha_{\sigma m}$, are those associated with a r^{th} order method derived for a smooth function, the value of Q will be $O(\Delta x^r)$ as long as enough terms are included in the expansion (i.e., $N+1-m \geq r$). Thus, in the symmetric jump condition case, there is no need to compute custom stencil coefficients. We can simply use standard stencil coefficients in (3.71) leaving only the values of the additive corrections

C_σ to be determined. If we use the stencil coefficients associated with an r^{th} order scheme, the derivatives at the stencils spanning the interface will remain r^{th} order accurate provided we calculate enough terms in C_σ (i.e., we satisfy $N+1-m \geq r$). In general, the order of the approximation of the m^{th} derivative by an r^{th} order scheme will be the smaller of r and $N+1-m$.

Examples

Having established that any function that satisfies symmetric jump relations can be accurately differentiated using slightly modified standard finite difference stencils, we now give some simple applications of this technique. Consider finding the Green's function for the steady state heat equation,

$$\frac{d^2G}{dx^2} = \delta\left(x - X\right), \tag{3.83}$$

on the domain $0 \leq x \leq 1$. The jump relations governing G are implicitly given above in the statement of the problem. Because of the nature of the singularity, we know that G must be continuous across the interface at $x = X$,

$$G^+ = G^-. \tag{3.84}$$

Integrating the equation across in the interface establishes the jump in the first derivative,

$$\frac{dG^+}{dx} = \frac{dG^-}{dx} + 1, \tag{3.85}$$

and evaluating the equation on both sides of the interface fixes the jump in the second derivative,

$$\frac{d^2G^+}{dx^2} = \frac{d^2G^-}{dx^2}. \tag{3.86}$$

The jumps of the higher derivatives are obtained by differentiating (3.83) and evaluating on both sides of the interface. For this problem

$$\frac{d^n G^+}{dx^n} = \frac{d^2 G^-}{dx^2} \text{ for } n \geq 2. \tag{3.87}$$

Therefore, the jumps at $x = X$ are given by

$$\left[\frac{d^n G}{dx^n}\right] = \begin{cases} 0 & n \neq 1 \\ 1 & n = 1 \end{cases}. \tag{3.88}$$

The jump conditions calculated above are symmetric. Thus, we can use standard finite difference stencils to discretize the derivatives of G. From the definition of the additive correction, (3.80), and (3.88) it is clear that the corrections can be calculated to arbitrary order using only a one term sum (i.e., the term proportional to the jump in the first derivative). Discretizing the problem using (3.41) with standard second order finite difference stencil coefficients and collecting all known quantities on the right-hand side leads to the system

$$\begin{aligned} G(0) &= 0 & (3.89) \\ \frac{G(i+1) - 2G(i) + G(i-1)}{\Delta x^2} &= f(i), \, 0 < i < M & (3.90) \\ G(M) &= 0 & (3.91) \end{aligned}$$

where $\Delta x = \frac{i}{M}$, $x_i = i\Delta x$, $G(i) = G(x_i)$ and, if the interface falls between x_k and x_{k+1},

$$f(i) = \begin{cases} \dfrac{x_{k+1} - X}{\Delta x^2} & i = k \\ \dfrac{X - x_k}{\Delta x^2} & i = k+1 \\ 0 & \text{otherwise} \end{cases} \tag{3.92}$$

M	2^{nd} order error	Ratio	4^{th} order error	Ratio
10	0.01200		0.01170	
20	0.00600	2.0000	0.00593	1.9730
40	0.00300	2.0000	0.00298	1.9900
80	0.00150	2.0000	0.00149	2.0000

Table 3.1: Resolution study of the one-dimensional steady heat equation using an uncorrected second and fourth order accurate stencil

Note that the forcing in this system is due entirely to the additive correction terms that modified the stencils. Solving this system yields the exact solution

$$G(x; X) = \begin{cases} -x(1-X) & \text{for } 0 \le x \le X \\ -X(1-x) & \text{for } X < x \le 1 \end{cases} \tag{3.93}$$

to within machine precision.

For our next example, we again consider solving the steady state heat equation. This time we suppose that the system

$$\frac{d^2u}{dx^2} = f \tag{3.94}$$

is subjected to non-uniform heating

$$f = \begin{cases} 0 & \text{for } 0 \le x \le X \\ 1 & \text{for } X < x \le 1 \end{cases} \tag{3.95}$$

with a fixed temperature specified at both ends of the domain

$$u(0) = 0 \tag{3.96}$$

$$u(1) = 0 \tag{3.97}$$

In this case, it is possible to naively discretize the problem using standard, uncorrected, finite differences. The naive solution will only be first order accurate, however, regardless of the formal accuracy of the stencils used ($X = 0.6$ is used to calculate the results in Table 3.1).

The accuracy of the solution computed using uncorrected stencils is limited to first order (see Table 3.1) because the exact solution has a discontinuous second derivative that is not accounted for. This problem satisfies symmetric jump relations with jumps

$$\left[\frac{d^n u}{dx^n}\right] = \begin{cases} 0 & n \neq 2 \\ 1 & n = 2 \end{cases} \tag{3.98}$$

which can be determined from the problem statement. Recalling our discussion of the accuracy of corrected stencils, we found that if the additive correction included contributions from the jumps in derivatives through order N, that the discrete approximation to the m^{th} derivative calculated using an r^{th} order finite difference stencil would have an accuracy given by the minimum of r and $N+1-m$. In this case, the jumps in the value and first derivative of u are zero so including no additive correction term is equivalent to including one with $N=1$. Thus, the failure to include corrections leads to an $O(1)$ error in the discretization of the second derivative for this problem. The larger error at the stencils spanning the interface, in turn, pollutes the solution on the entire grid resulting in an accuracy $O(\Delta x)$ times the error incurred at the interface spanning stencils (see [38]), or, in this case, a first order accurate method. This explains why, in Table 3.1, the computed solution is only first order accurate, independent of the formal accuracy of the finite difference stencils.

Now we discretize this problem using second order accurate finite differences and apply the additive corrections outlined above to obtain

$$\begin{aligned} u(0) &= 0 && (3.99) \\ \frac{u(i+1) - 2u(i) + u(i+1)}{\Delta x^2} &= f(i), \; 0 < i < M && (3.100) \\ u(M) &= 0 && (3.101) \end{aligned}$$

where $\Delta x = \frac{i}{M}$, $x_i = i\Delta x$, $u(i) = u(x_i)$ and, if the interface falls between x_k and

x_{k+1},

$$f\left(i\right)=\begin{cases} 0 & i<k \\ \dfrac{\left(x_{k+1}-X\right)^2}{2\Delta x^2} & i=k \\ 1-\dfrac{\left(x_k-X\right)^2}{2\Delta x^2} & i=k+1 \\ 1 & i>k+1 \end{cases} . \tag{3.102}$$

Solving this system again yields the exact solution

$$u\left(x\right)=\begin{cases} \dfrac{-x\left(X-1\right)^2}{2} & \text{for } 0\leq x\leq X \\ \dfrac{-\left(1-x\right)\left(x-X^2\right)}{2} & \text{for } X<x\leq 1 \end{cases} \tag{3.103}$$

to within machine precision. We will demonstrate the application of the theory to a non-symmetric problem in Chapter 4.

3.3 Spatial derivatives in two dimension

In this section we develop a method for accurately differentiating non-smooth functions in two dimensions. As was the case in one dimension, we assume that the function under consideration is smooth and completely differentiable everywhere except at a small subset of points. In the one-dimensional case, the function was allowed to have discontinuities in its values and/or derivatives at a single point in the domain. In the two-dimensional case, we allow the function to have discontinuities in its values and/or derivatives across a smooth curve (interface). The interface must divide the domain into two distinct regions (i.e., inside and outside). This can be accomplished using either a simple closed curve or a curve of infinite extent. A point in the domain, (x, y), is said to be an outside, or "+", point if the normal to the curve, (n_x, n_y), at the closest point on the curve, (X, Y), is directed towards (x, y), and hence satisfies

$$\left(x-X\right)n_x+\left(y-Y\right)n_y\geq 0. \tag{3.104}$$

Similarly, a point (x, y) is said to be an inside, or "-", point if the normal is directed away from the point and, therefore, satisfies

$$(x - X)\, n_x + (y - Y)\, n_y < 0. \tag{3.105}$$

While it is clearly more complicated to identify which points are on the "+" and "-" sides of the interface in two dimensions, the above is a natural generalization of the one dimension test (i.e., a "+" point satisfies $y - Y \geq 0$ and a "-" point satisfies $y - Y < 0$).

We assume that a parametric representation for the curve, $(X(s), Y(s))$, in terms of arc length s is known. The unit tangent of such a curve is given by

$$\begin{aligned} \tau_x &= \frac{dX}{ds}, & (3.106) \\ \tau_y &= \frac{dY}{ds}, & (3.107) \end{aligned}$$

and we take the normal to be

$$\begin{aligned} n_x &= -\frac{dY}{ds}, & (3.108) \\ n_y &= +\frac{dX}{ds}. & (3.109) \end{aligned}$$

The jump conditions in two dimensions, which must be known, take the form

$$\frac{\partial^n u^+(s)}{\partial x^k \partial y^{n-k}} = \sum_{p=0}^{n} \sum_{r=0}^{p} D_{pr}^{nk-}(s) \frac{\partial^p u^-(s)}{\partial x^r \partial y^{p-r}} + J_{nk}^-(s), \tag{3.110}$$

and

$$\frac{\partial^n u^-(s)}{\partial x^k \partial y^{n-k}} = \sum_{p=0}^{n} \sum_{r=0}^{p} D_{pr}^{nk+}(s) \frac{\partial^p u^+(s)}{\partial x^r \partial y^{p-r}} + J_{nk}^+(s), \tag{3.111}$$

where

$$\begin{aligned} \frac{\partial^n u^+(s)}{\partial x^k \partial y^{n-k}} &= \lim_{\varepsilon \to 0+} \frac{\partial^n u^+(X(s) + \varepsilon n_x, Y(s) + \varepsilon n_y)}{\partial x^k \partial y^{n-k}}, & (3.112) \\ \frac{\partial^n u^-(s)}{\partial x^k \partial y^{n-k}} &= \lim_{\varepsilon \to 0+} \frac{\partial^n u^-(X(s) - \varepsilon n_x, Y(s) - \varepsilon n_y)}{\partial x^k \partial y^{n-k}}, & (3.113) \end{aligned}$$

and

$$D_{cd}^{ab+} = D_{cd}^{ab-} = 0 \text{ if } c > a. \tag{3.114}$$

As was the case in one dimension, we will demonstrate how to construct a general expansion of the form

$$u(\sigma) - C_\sigma = \sum_{n=0}^{N} \sum_{k=0}^{n} \omega_{nk\sigma} \frac{\partial^n u(0)}{\partial x^k \partial y^{n-k}} + O\left(\Delta x^{N+1}\right), \tag{3.115}$$

which represents $u(x_\sigma, y_\sigma)$ in terms of $u(x_0, y_0)$ and it derivatives. Using (3.115) we will derive stencils for the derivatives at (x_0, y_0) that remain accurate even when it is adjacent to the interface. First we will demonstrate the approach by using it to determine a stencil for the differentiation of a perfectly smooth function.

The construction of stencils in two dimensions is significantly more involved than it was in the simple one-dimensional case. We will work through the derivation of the stencils for smooth functions to demonstrate how to deal with the cumbersome multi-indexed objects that need to be calculated. Even in the general case of a function which may jump across the interface, if a line segment connecting the node at (x_σ, y_σ) and the node at (x_0, y_0) does not intersect the interface, we can use a simple Taylor series expansion for $u(\sigma)$,

$$u(\sigma) = \sum_{n=0}^{N} \sum_{k=0}^{n} \left[\frac{(x_\sigma - x_0)^k}{k!}\right] \left[\frac{(y_\sigma - y_0)^{n-k}}{(n-k)!}\right] \frac{\partial^n u(0)}{\partial x^k \partial y^{n-k}}. \tag{3.116}$$

The line segment connecting (x_0, y_0) and (x_σ, y_σ) is called the stencil leg associated with node σ. Comparing (3.115) and (3.116), we see that the expansion coefficients for a node, σ, whose stencil leg is not intersected by the interface, are

$$\omega_{nk\sigma} = \left[\frac{(x_\sigma - x_0)^k}{k!}\right] \left[\frac{(y_\sigma - y_0)^{n-k}}{(n-k)!}\right], \tag{3.117}$$

with the additive corrections terms,

$$C_\sigma = 0. \tag{3.118}$$

We will seek an $M+1$ node stencil with nodes (x_0, y_0), (x_1, y_1),...,(x_M, y_M) and with stencil coefficients defined by

$$\sum_{\sigma=0}^{M} \alpha_{\sigma m p} \left[u(\sigma) - C_\sigma \right] = \frac{\partial^m u(0)}{\partial x^p \partial y^{m-p}}. \tag{3.119}$$

Substituting (3.115) into (3.119) we find

$$\sum_{\sigma=0}^{M} \sum_{n=0}^{N} \sum_{k=0}^{n} \alpha_{\sigma m p} \omega_{nk\sigma} \frac{\partial^n u(0)}{\partial x^k \partial y^{n-k}} = \frac{\partial^m u(0)}{\partial x^p \partial y^{m-p}}, \tag{3.120}$$

which, using the Kronecker delta, can be rewritten as

$$\sum_{n=0}^{N} \sum_{k=0}^{n} \left(\sum_{\sigma=0}^{M} \omega_{nk\sigma} \alpha_{\sigma m p} - \delta_{nm} \delta_{kp} \right) \frac{\partial^n u(0)}{\partial x^k \partial y^{n-k}} = 0. \tag{3.121}$$

Noting that the values of the derivatives are arbitrary, we have

$$\sum_{\sigma=0}^{M} \omega_{nk\sigma} \alpha_{\sigma m p} = \delta_{nm} \delta_{kp}. \tag{3.122}$$

This is a perfectly valid set of equations, but the use of three indexed objects is cumbersome. It is possible to represent the equations using two index objects by parametrizing the sums over (n, k) and (m, p). We do this by defining new expansion coefficients,

$$\omega_{\lambda\sigma} = \omega_{n(\lambda)k(\lambda)\sigma}, \tag{3.123}$$

and stencil coefficients,

$$\alpha_{\sigma a} = \alpha_{\sigma m(a) p(a)}. \tag{3.124}$$

There are $N_\lambda = \frac{1}{2}(N+2)(N+1)$ terms in the sum over (n, k) so λ ranges from 0 to $N_\lambda - 1$. The values of $n(\lambda)$ and $k(\lambda)$ range over the values they would take on in the full double sum,

$$n(\lambda) = \{0, 1, 1, 2, 2, 2, 3, 3, 3, 3, 4, ...\}, \tag{3.125}$$

$$k(\lambda) = \{0,0,1,0,1,2,0,1,2,3,0,...\}. \tag{3.126}$$

The indices (m, p) specify the derivative that the stencil coefficients are discretizing, so they are parametrized exactly like (n, k), with a ranging from 0 to $N_\lambda - 1$ and

$$m(a) = \{0,1,1,2,2,2,3,3,3,3,4,...\}, \tag{3.127}$$

$$p(a) = \{0,0,1,0,1,2,0,1,2,3,0,...\}. \tag{3.128}$$

Using this notation in (3.122), we find

$$\sum_{\sigma=0}^{M} \omega_{\lambda\sigma}\alpha_{\sigma a} = \delta_{n(\lambda)m(a)}\delta_{k(\lambda)p(\alpha)} = \delta_{\lambda a}, \tag{3.129}$$

or, in matrix notation,

$$\boldsymbol{\omega\alpha} = \mathbf{I}. \tag{3.130}$$

The matrix $\boldsymbol{\omega}$ has N_λ rows and $M+1$ columns, one associated with each node in the proposed stencil. In order for the above to be an invertible system, therefore, we must specify N_λ nodes $(M = N_\lambda - 1)$ in our stencil. The position of the nodes is specified relative to the location of the node where the derivatives are evaluated, (x_0, y_0). Although it is not necessary for the nodes to be regularly spaced, we will assume that the computational nodes are located on a uniform grid with mesh spacing $\Delta y = \Delta x$. To discretize all the derivatives up to second order $(N = 2)$, we need a stencil consisting of six nodes. One such stencil is given by

$$\frac{x_\sigma - x_0}{\Delta x} = \{0,1,-1,0,0,-1\}, \tag{3.131}$$

$$\frac{y_\sigma - y_0}{\Delta x} = \{0,0,0,1,-1,-1\}. \tag{3.132}$$

This stencil is not unique. There are many choices that produce a non-singular $\boldsymbol{\omega}$. The above stencil corresponds to the standard five point cross with the additional inclusion of the lower left diagonal node to bring the total up to six. The coefficient matrix associated with these node positions is

$$\boldsymbol{\omega} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & \Delta x & -\Delta x & -\Delta x \\ 0 & \Delta x & -\Delta x & 0 & 0 & -\Delta x \\ 0 & 0 & 0 & \frac{1}{2}\Delta x^2 & \frac{1}{2}\Delta x^2 & \frac{1}{2}\Delta x^2 \\ 0 & 0 & 0 & 0 & 0 & \Delta x^2 \\ 0 & \frac{1}{2}\Delta x^2 & \frac{1}{2}\Delta x^2 & 0 & 0 & \frac{1}{2}\Delta x^2 \end{bmatrix}. \tag{3.133}$$

The stencil coefficients that correspond to these nodes are

$$\boldsymbol{\alpha} = \boldsymbol{\omega}^{-1} = \begin{bmatrix} 1 & 0 & 0 & \frac{-2}{\Delta x^2} & \frac{1}{\Delta x^2} & \frac{-2}{\Delta x^2} \\ 0 & 0 & \frac{1}{2\Delta x} & 0 & 0 & \frac{1}{\Delta x^2} \\ 0 & 0 & \frac{-1}{2\Delta x} & 0 & \frac{-1}{\Delta x^2} & \frac{1}{\Delta x^2} \\ 0 & \frac{1}{2\Delta x} & 0 & \frac{1}{\Delta x^2} & 0 & 0 \\ 0 & \frac{-1}{2\Delta x} & 0 & \frac{1}{\Delta x^2} & \frac{-1}{\Delta x^2} & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{\Delta x^2} & 0 \end{bmatrix}, \tag{3.134}$$

or

$$u_x(0) = \frac{u(1) - u(2)}{2\Delta x}, \tag{3.135}$$

$$u_y(0) = \frac{u(3) - u(4)}{2\Delta x}, \tag{3.136}$$

$$u_{xx}(0) = \frac{u(1) - 2u(0) + u(2)}{\Delta x^2}, \tag{3.137}$$

$$u_{xy}(0) = \frac{u(0) - u(2) - u(4) + u(5)}{\Delta x^2}, \tag{3.138}$$

$$u_{yy}(0) = \frac{u(3) - 2u(0) + u(4)}{\Delta x^2}. \tag{3.139}$$

Note that all of the above correspond to the standard second order finite difference stencils except for the formula for the cross derivative, u_{xy}. Unlike the other stencils, the formula for the cross derivative is only first order accurate. In general, however, this is the best we can expect if we only use expansions out to the second order derivatives. It is possible to obtain higher order formulas by specifying a larger stencil (10 nodes for third order accuracy, 15 nodes for fourth order accuracy, etc.).

Now we consider the derivation of the expansion coefficients and additive corrections in the general case. The first thing we need to do is establish some notation. We denote all the points in the domain that are outside the interface, as defined above, by Γ^+. Similarly, we denote all the points inside the interface by Γ^-. Thus, if $\mathbf{x} \in \Gamma^+$ then $\mathbf{x}$ is a point in the domain that is outside the interface, and if $\mathbf{x} \in \Gamma^-$ then it is inside the interface. Using this notation, the stencil leg connecting $u(0)$ and $u(\sigma)$ is assumed to not intersect the interface if $\mathbf{x}_0 \in \Gamma^+$ and $\mathbf{x}_\sigma \in \Gamma^+$ or $\mathbf{x}_0 \in \Gamma^-$ and $\mathbf{x}_\sigma \in \Gamma^-$ (i.e., both nodes lie on the same side of the interface). As we mentioned above, when the stencil leg associated with node σ is not intersected by the interface, the expansion coefficients are the standard Taylor series coefficients

$$\omega_{nk\sigma} = \left[\frac{(x_\sigma - x_0)^k}{k!}\right]\left[\frac{(y_\sigma - y_0)^{n-k}}{(n-k)!}\right], \tag{3.140}$$

and the additive corrections terms are not needed,

$$C_\sigma = 0. \tag{3.141}$$

In the general case, however, the above expansion will not be valid because it assumes the function is well behaved everywhere between the two nodes. We can construct the expansion coefficients, however, by following a straightforward generalization of the approach we took in one dimension. Suppose that we are constructing a stencil to calculate the derivatives at $\mathbf{x}_0 \in \Gamma^-$. We can use (3.140) and (3.141) for the stencil nodes that obey $\mathbf{x}_\sigma \in \Gamma^-$. For the nodes that lie on the opposite side of the interface ($\mathbf{x}_\sigma \in \Gamma^+$), however, the closest we can get to the center node using a Taylor series is to express $u(\sigma)$ in terms of the derivatives at the interface,

$$u(\sigma) = \sum_{a=0}^{N}\sum_{b=0}^{a}\left[\frac{(x_\sigma - X)^b}{b!}\right]\left[\frac{(y_\sigma - Y)^{a-b}}{(a-b)!}\right]\frac{\partial^a u^+}{\partial x^b \partial y^{a-b}}. \tag{3.142}$$

The point on the interface (X, Y) is taken to be the intersection between the stencil leg and the interface but, strictly speaking, this is not required. We can then use the jump condition, reproduced here with convenient indices,

$$\frac{\partial^a u^+}{\partial x^b \partial y^{a-b}} = \sum_{c=0}^{a} \sum_{d=0}^{c} D_{cd}^{ab-} \frac{\partial^c u^-}{\partial x^d \partial y^{c-d}} + J_{ab}^-, \tag{3.143}$$

to convert (3.142) into an expression relating $u(\sigma)$ to quantities evaluated on the opposite side of the interface,

$$\begin{aligned} u(\sigma) \;=\; & \sum_{a=0}^{N} \sum_{b=0}^{a} \sum_{c=0}^{a} \sum_{d=0}^{c} D_{cd}^{ab-} \left[\frac{(x_\sigma - X)^b}{b!} \right] \left[\frac{(y_\sigma - Y)^{a-b}}{(a-b)!} \right] \frac{\partial^c u^-}{\partial x^d \partial y^{c-d}} \\ & + \sum_{a=0}^{N} \sum_{b=0}^{a} \left[\frac{(x_\sigma - X)^b}{b!} \right] \left[\frac{(y_\sigma - Y)^{a-b}}{(a-b)!} \right] J_{ab}^-. \end{aligned} \tag{3.144}$$

Now we need to find a way to express $\frac{\partial^c u^-}{\partial x^d \partial y^{c-d}}$ in terms of $u(0)$ and its derivatives. Recalling that $\mathbf{x}_0 \in \Gamma^-$, we can approximate u at any sufficiently close $\mathbf{x} \in \Gamma^-$ using a standard Taylor series

$$u(x, y) = \sum_{n=0}^{N} \sum_{k=0}^{n} \left[\frac{(x - x_0)^k}{k!} \right] \left[\frac{(y - y_0)^{n-k}}{(n-k)!} \right] \frac{\partial^n u(0)}{\partial x^k \partial y^{n-k}}. \tag{3.145}$$

Differentiating (3.145) yields a general expression for the derivatives of $u(x, y)$,

$$\frac{\partial^c u(x, y)}{\partial x^d \partial y^{c-d}} = \sum_{n=c}^{N} \sum_{k=d}^{n+d-c} \left[\frac{(x - x_0)^{k-d}}{(k-d)!} \right] \left[\frac{(y - y_0)^{n+d-c-k}}{(n+d-c-k)!} \right] \frac{\partial^n u(0)}{\partial x^k \partial y^{n-k}}. \tag{3.146}$$

Evaluating (3.146) at (X, Y) and substituting the result into (3.144), we find

$$\begin{aligned} u(\sigma) \;=\; & \sum_{a=0}^{N} \sum_{b=0}^{a} \sum_{c=0}^{a} \sum_{d=0}^{c} \sum_{n=c}^{N} \sum_{k=d}^{n+d-c} \left\{ D_{cd}^{ab-} \left[\frac{(x_\sigma - X)^b}{b!} \right] \left[\frac{(y_\sigma - Y)^{a-b}}{(a-b)!} \right] \right. \\ & \left. \left[\frac{(X - x_0)^{k-d}}{(k-d)!} \right] \left[\frac{(Y - y_0)^{n+d-c-k}}{(n+d-c-k)!} \right] \frac{\partial^n u(0)}{\partial x^k \partial y^{n-k}} \right\} \\ & + \sum_{a=0}^{N} \sum_{b=0}^{a} \left[\frac{(x_\sigma - X)^b}{b!} \right] \left[\frac{(y_\sigma - Y)^{a-b}}{(a-b)!} \right] J_{ab}^-. \end{aligned} \tag{3.147}$$

The values of $\omega_{nk\sigma}$ are not clear in (3.147) because the sums over "n" and "k" are performed last instead of first. It is possible to rearrange the order of the summations if we use (3.114) and extend the sum over "c" to $0 \leq c \leq N$. This accomplished, we can easily exchange summations until the outer most sums are over "n" and "k" and (3.147) becomes (3.115) with

$$\begin{aligned}\omega_{nk\sigma} &= \sum_{a=0}^{N}\sum_{b=0}^{a}\sum_{d=0}^{k}\sum_{c=d}^{n+d-k}\left\{D_{cd}^{ab-}\left[\frac{(x_\sigma - X)^b}{b!}\right]\left[\frac{(y_\sigma - Y)^{a-b}}{(a-b)!}\right]\right.\\ &\quad \left.\left[\frac{(X-x_0)^{k-d}}{(k-d)!}\right]\left[\frac{(Y-y_0)^{n+d-c-k}}{(n+d-c-k)!}\right]\right\} \qquad (3.148)\end{aligned}$$

and

$$C_\sigma = \sum_{a=0}^{N}\sum_{b=0}^{a}\left[\frac{(x_\sigma - X)^b}{b!}\right]\left[\frac{(y_\sigma - Y)^{a-b}}{(a-b)!}\right]J_{ab}^{-}. \qquad (3.149)$$

These are the general expressions for the expansion coefficients and additive corrections for a stencil centered at $\mathbf{x}_0 \in \Gamma^-$ with a stencil leg intersected by the interface. The expressions for the $\mathbf{x}_0 \in \Gamma^+$ case are

$$\begin{aligned}\omega_{nk\sigma} &= \sum_{a=0}^{N}\sum_{b=0}^{a}\sum_{d=0}^{k}\sum_{c=d}^{n+d-k}\left\{D_{cd}^{ab+}\left[\frac{(x_\sigma - X)^b}{b!}\right]\left[\frac{(y_\sigma - Y)^{a-b}}{(a-b)!}\right]\right.\\ &\quad \left.\left[\frac{(X-x_0)^{k-d}}{(k-d)!}\right]\left[\frac{(Y-y_0)^{n+d-c-k}}{(n+d-c-k)!}\right]\right\} \qquad (3.150)\end{aligned}$$

and

$$C_\sigma = \sum_{a=0}^{N}\sum_{b=0}^{a}\left[\frac{(x_\sigma - X)^b}{b!}\right]\left[\frac{(y_\sigma - Y)^{a-b}}{(a-b)!}\right]J_{ab}^{+}, \qquad (3.151)$$

which are, of course, identical to the above except that D_{cd}^{ab-} is replaced by D_{cd}^{ab+} and J_{ab}^{-} is replaced by J_{ab}^{+}.

It is interesting to examine the special case of symmetric jump conditions. In two dimensions, symmetric jump conditions take the form

$$\frac{\partial^n u^+}{\partial x^k \partial y^{n-k}} = \frac{\partial^n u^-}{\partial x^k \partial y^{n-k}} + \left[\frac{\partial^n u}{\partial x^k \partial y^{n-k}}\right], \qquad (3.152)$$

and

$$\frac{\partial^n u^-}{\partial x^k \partial y^{n-k}} = \frac{\partial^n u^+}{\partial x^k \partial y^{n-k}} - \left[\frac{\partial^n u}{\partial x^k \partial y^{n-k}}\right]. \tag{3.153}$$

Clearly only one jump condition per derivative is required in the symmetric case. Comparing (3.110) with (3.152) and (3.111) with (3.153), we see that symmetric jump conditions corresponds to

$$D^{ab-}_{cd} = \delta_{ac}\delta_{bd}, \tag{3.154}$$

$$J^-_{nk} = \left[\frac{\partial^n u}{\partial x^k \partial y^{n-k}}\right], \tag{3.155}$$

and

$$D^{ab+}_{cd} = \delta_{ac}\delta_{bd}, \tag{3.156}$$

$$J^+_{nk} = -\left[\frac{\partial^n u}{\partial x^k \partial y^{n-k}}\right]. \tag{3.157}$$

When (3.154) is substituted into (3.148), it yields

$$\omega_{nk\sigma} = \sum_{d=0}^{k} \sum_{c=d}^{n+d-k} \left[\frac{(x_\sigma - X)^d}{d!}\right] \left[\frac{(y_\sigma - Y)^{c-d}}{(c-d)!}\right] \left[\frac{(X - x_0)^{k-d}}{(k-d)!}\right] \left[\frac{(Y - y_0)^{n+d-c-k}}{(n+d-c-k)!}\right] \tag{3.158}$$

Substituting $c = c' + d$ into (3.158) and reordering terms simplifies (3.158) to

$$\omega_{nk\sigma} = \sum_{d=0}^{k} \left[\frac{(x_\sigma - X)^d}{d!}\right] \left[\frac{(X - x_0)^{k-d}}{(k-d)!}\right] \sum_{c'=0}^{n-k} \left[\frac{(y_\sigma - Y)^{c'}}{(c')!}\right] \left[\frac{(Y - y_0)^{n-k-c'}}{(n-k-c')!}\right] \tag{3.159}$$

The sums in (3.159) can be calculated explicitly using the binomial formula which yields

$$\sum_{d=0}^{k} \left[\frac{(x_\sigma - X)^d}{d!}\right] \left[\frac{(X - x_0)^{k-d}}{(k-d)!}\right] = \frac{(x_\sigma - x_0)^k}{k!} \tag{3.160}$$

and

$$\sum_{c'=0}^{n-k} \left[\frac{(y_\sigma - Y)^{c'}}{(c')!}\right] \left[\frac{(Y - y_0)^{n-k-c'}}{(n-k-c')!}\right] = \frac{(y_\sigma - y_0)^{n-k}}{(n-k)!} \tag{3.161}$$

and we find

$$\omega_{nk\sigma} = \left[\frac{(x_\sigma - x_0)^k}{k!}\right] \left[\frac{(y_\sigma - y_0)^{n-k}}{(n-k)!}\right], \tag{3.162}$$

which are the standard Taylor series coefficients. Obviously, we find the same result if (3.156) is substituted into (3.150).

Thus non-smooth functions that satisfy symmetric jump conditions have expansions that differ from a Taylor series by only an additive correction. As we demonstrated in one dimension, this implies that standard finite difference stencil coefficients can be used to approximate the derivatives of these function. In particular, if

$$\frac{\partial^n u(x_0, y_0)}{\partial x^k \partial y^{n-k}} \approx \sum_{\sigma=0}^{M} \alpha_{\sigma n k} \left[u(x_\sigma, y_\sigma)\right] \tag{3.163}$$

is a r^{th} order approximation to the derivative, a smooth function then, for $N = n+r-1$

$$\frac{\partial^n u(x_0, y_0)}{\partial x^k \partial y^{n-k}} \approx \sum_{\sigma=0}^{M} \alpha_{\sigma n k} \left[u(x_\sigma, y_\sigma) - C_\sigma\right], \tag{3.164}$$

will be a r^{th} order accurate approximation to the derivative of $u(x, y)$ even when it has jumps between some of the nodes. The supporting argument is so similar to the one-dimensional case that we will not bother to present it here.

Calculating accurate derivatives in two dimensions is more involved than in one dimension, but the general procedure is similar. To discretize the derivatives at $\mathbf{x}_0$, we

1. Determine all the stencil leg intersections and the exact location of the points of intersection $(X(s_I), Y(s_I))$ (where s_I is the arc length associated with the intersection point).

2. Using (3.140), (3.148) or (3.150), set the columns of $\boldsymbol{\omega}$ associated with each stencil node.

3. Using (3.141), (3.149) or (3.151), set the values of the additive corrections, C_σ, associated with each stencil node.

4. Determine the stencil coefficients using $\boldsymbol{\alpha} = \boldsymbol{\omega}^{-1}$.

5. Compute the value of the desired derivative at x_0 using

$$\frac{\partial^n u(0)}{\partial^k x \partial y^{n-k}} = \sum_{\sigma=0}^{M} \alpha_{\sigma n k} \left[u(\sigma) - C_\sigma \right]. \tag{3.165}$$

If the nodal values are unknown and the above is to be incorporated into a system to determine their values, the additive correction, $\sum_{\sigma=0}^{M} \alpha_{\sigma m} C_\sigma$, is incorporated into the system forcing.

Examples

In this section we derive the jump conditions for the Poisson problem given by

$$\nabla^2 u = f. \tag{3.166}$$

In addition to the above, we assume that the jump in the value of the solution, $[u](s)$,

$$[u] = u^+ - u^-, \tag{3.167}$$

and the jump in the "flux," $[hu_n](s)$,

$$[hu_n](s) = [hu_n] = h^+ \frac{\partial u^+}{\partial n} - h^- \frac{\partial u^-}{\partial n}, \tag{3.168}$$

are known everywhere along the interface as are h^+ and h^- which are assumed to be constants. We will begin by deriving the jump conditions on the derivatives for the special case of symmetric material properties ($h = h^+ = h^-$). For this case, we can take $h = 1$ without loss of generality.

Symmetric jump conditions Consider the symmetric Poisson problem given by

$$\nabla^2 u = f \tag{3.169}$$

in each region, where $f(x,y)$ is a known function, as are the jumps in the solution value, $[u](s)$,

$$[u] = u^{+} - u^{-}, \tag{3.170}$$

and jump in normal derivative, $[u_n](s)$,

$$[u_n] = u_n^{+} - u_n^{-}, \tag{3.171}$$

along the interface. Our goal is to calculate the jumps in the spatial derivatives, $[u_x]$, $[u_y]$, etc.

To determine the jump in the x and y derivatives separately, we must extract this information from the known jump conditions (3.170) and (3.171). First, we note that the tangent to the interface is given by ($s =$ arc length)

$$\tau_x = \frac{\partial X}{\partial s}, \tag{3.172}$$

$$\tau_y = \frac{\partial Y}{\partial s}, \tag{3.173}$$

and define the normal to be

$$n_x = -\tau_y = -\frac{\partial Y}{\partial s}, \tag{3.174}$$

$$n_y = +\tau_x = +\frac{\partial X}{\partial s}. \tag{3.175}$$

So, from the definition of a normal derivative, we see that (3.170) implies

$$\left(n_x u_x^{+} + n_y u_y^{+}\right) - \left(n_x u_x^{-} + n_y u_y^{-}\right) = [u_n]. \tag{3.176}$$

Similarly, the derivative of (3.171) with respect to arc length implies

$$\left(\tau_x u_x^{+} + \tau_y u_y^{+}\right) - \left(\tau_x u_x^{-} + \tau_y u_y^{-}\right) = \frac{\partial [u]}{\partial s} = [u]_s. \tag{3.177}$$

Combining these together and using the relationship between the normal and tangent vectors yields

$$\begin{bmatrix} \tau_x & \tau_y \\ n_x & n_y \end{bmatrix} \begin{bmatrix} u_x^+ \\ u_y^+ \end{bmatrix} - \begin{bmatrix} \tau_x & \tau_y \\ n_x & n_y \end{bmatrix} \begin{bmatrix} u_x^- \\ u_y^- \end{bmatrix} = \begin{bmatrix} [u]_s \\ [u_n] \end{bmatrix}. \tag{3.178}$$

Noting that the determinant of the above system is $\tau_x^2 + \tau_y^2 = 1$, we solve for the jumps in the first derivatives

$$\begin{bmatrix} [u_x] \\ [u_y] \end{bmatrix} = \begin{bmatrix} u_x^+ \\ u_y^+ \end{bmatrix} - \begin{bmatrix} u_x^- \\ u_y^- \end{bmatrix} = \begin{bmatrix} \tau_x & n_x \\ \tau_y & n_y \end{bmatrix} \begin{bmatrix} [u]_s \\ [u_n] \end{bmatrix}. \tag{3.179}$$

Since this expression was obtained purely from our knowledge of the jump of the solution and its normal derivative across the interface, it is essentially independent of the equation that we are solving. These jumps can be differentiated to supply information about how the second derivatives behave near the interface. Noting that

$$\frac{\partial}{\partial s}\left(u_x^+(X,Y)\right) = \tau_x u_{xx}^+ + \tau_y u_{xy}^+, \tag{3.180}$$

and

$$\frac{\partial}{\partial s}\left(u_y^+(X,Y)\right) = \tau_x u_{xy}^+ + \tau_y u_{yy}^+, \tag{3.181}$$

with similar expressions for the derivatives of u_x^- and u_y^-, we find

$$\begin{bmatrix} \tau_x & \tau_y & 0 \\ 0 & \tau_x & \tau_y \end{bmatrix} \left\{ \begin{bmatrix} u_{xx}^+ \\ u_{xy}^+ \\ u_{yy}^+ \end{bmatrix} - \begin{bmatrix} u_{xx}^- \\ u_{xy}^- \\ u_{yy}^- \end{bmatrix} \right\} = \begin{bmatrix} [u_x]_s \\ [u_y]_s \end{bmatrix}. \tag{3.182}$$

In order to close the system and solve for the jumps in the derivatives in this case we need an additional equation. The only other information that we have available comes from the Poisson equation itself. Evaluating the equation on both sides of the interface and subtracting the result yields

$$\left(u_{xx}^{+}+u_{yy}^{+}\right)-\left(u_{xx}^{-}+u_{yy}^{-}\right)=[f], \tag{3.183}$$

where $[f](s)$ is the (known) jump in the (known) forcing f,

$$[f]=f^{+}(X,Y)-f^{-}(X,Y). \tag{3.184}$$

Note that if the forcing is continuous, $[f]=0$. Combining the above we obtain a system for the jumps in the second derivatives

$$\begin{bmatrix} \tau_x & \tau_y & 0 \\ 0 & \tau_x & \tau_y \\ 1 & 0 & 1 \end{bmatrix} \left\{ \begin{bmatrix} u_{xx}^{+} \\ u_{xy}^{+} \\ u_{yy}^{+} \end{bmatrix} - \begin{bmatrix} u_{xx}^{-} \\ u_{xy}^{-} \\ u_{yy}^{-} \end{bmatrix} \right\} = \begin{bmatrix} [u_x]_s \\ [u_y]_s \\ [f] \end{bmatrix}. \tag{3.185}$$

The determinant of the above matrix is $\tau_x^2+\tau_y^2=1$, so we can solve for the jumps in the second derivatives and find, after some manipulation

$$\begin{bmatrix} [u_{xx}] \\ [u_{xy}] \\ [u_{yy}] \end{bmatrix} = \begin{bmatrix} u_{xx}^{+} \\ u_{xy}^{+} \\ u_{yy}^{+} \end{bmatrix} - \begin{bmatrix} u_{xx}^{-} \\ u_{xy}^{-} \\ u_{yy}^{-} \end{bmatrix} = \begin{bmatrix} \tau_x & -\tau_y & \tau_y^2 \\ \tau_y & \tau_x & -\tau_x\tau_y \\ -\tau_x & \tau_y & \tau_x^2 \end{bmatrix} \begin{bmatrix} [u_x]_s \\ [u_y]_s \\ [f] \end{bmatrix}. \tag{3.186}$$

It is possible to continue the process and explicitly calculate the jumps of all the higher order derivatives provided information about the derivatives of the forcing, f, is available. The general procedure for calculating the jumps in the higher order derivatives is best demonstrated by calculating the jumps in the third derivatives.

The calculations required to determine the jumps in the third derivatives are very similar to those for calculating the jumps in the second derivatives. First, we differentiate (3.186) with respect to s:

$$\begin{bmatrix} \tau_x & \tau_y & 0 & 0 \\ 0 & \tau_x & \tau_y & 0 \\ 0 & 0 & \tau_x & \tau_y \end{bmatrix} \left\{ \begin{bmatrix} u^+_{xxx} \\ u^+_{xxy} \\ u^+_{xyy} \\ u^+_{yyy} \end{bmatrix} - \begin{bmatrix} u^-_{xxx} \\ u^-_{xxy} \\ u^-_{xyy} \\ u^-_{yyy} \end{bmatrix} \right\} = \begin{bmatrix} [u_{xx}]_s \\ [u_{xy}]_s \\ [u_{yy}]_s \end{bmatrix}. \tag{3.187}$$

As before, an additional equation is required to close the system so that the jumps may be determined. This equation comes from the normal derivative of the governing equation. Evaluating the normal derivative of (3.169) on each side of the interface and subtracting the result yields

$$\left(n_x u^+_{xxx} + n_y u^+_{xxy}\right) - \left(n_x u^-_{xxx} + n_y u^-_{xxy}\right) = [f_n], \tag{3.188}$$

where

$$[f_n] = \frac{\partial f^+}{\partial n}(X,Y) - \frac{\partial f^-}{\partial n}(X,Y). \tag{3.189}$$

Incorporating this into the above system, we find

$$\begin{bmatrix} \tau_x & \tau_y & 0 & 0 \\ 0 & \tau_x & \tau_y & 0 \\ 0 & 0 & \tau_x & \tau_y \\ n_x & n_y & n_x & n_y \end{bmatrix} \left\{ \begin{bmatrix} u^+_{xxx} \\ u^+_{xxy} \\ u^+_{xyy} \\ u^+_{yyy} \end{bmatrix} - \begin{bmatrix} u^-_{xxx} \\ u^-_{xxy} \\ u^-_{xyy} \\ u^-_{yyy} \end{bmatrix} \right\} = \begin{bmatrix} [u_{xx}]_s \\ [u_{xy}]_s \\ [u_{yy}]_s \\ [f_n] \end{bmatrix}. \tag{3.190}$$

The determinant of this system is $\left(\tau_x^2 + \tau_y^2\right)^2 = 1$, so we can invert it to find

$$\begin{bmatrix} [u_{xxx}] \\ [u_{xxy}] \\ [u_{xyy}] \\ [u_{yyy}] \end{bmatrix} = \begin{bmatrix} \tau_x\left(1+\tau_y^2\right) & -\tau_y & \tau_x\tau_y^2 & -\tau_y^3 \\ \tau_y^3 & \tau_x & -\tau_x^2\tau_y & \tau_x\tau_y^2 \\ -\tau_x\tau_y^2 & \tau_y & \tau_x^3 & -\tau_x^2\tau_y \\ \tau_x^2\tau_y & -\tau_x & \tau_y\left(1+\tau_x^2\right) & \tau_x^3 \end{bmatrix} \begin{bmatrix} [u_{xx}]_s \\ [u_{xy}]_s \\ [u_{yy}]_s \\ [f_n] \end{bmatrix}. \tag{3.191}$$

This procedure can be repeated as many times as necessary to find the jumps in arbitrarily high order derivatives. So, for instance, to compute the jumps in the fourth derivatives, we would combine the derivative of (3.191) with the second derivative of the Poisson equation in the normal direction. The resulting system has a determinant of one and can be inverted to determine the jumps in the fourth derivative.

In practice, the algebraic and computational complexity increases with the order of the derivative being considered. Further, the calculation of higher order jumps requires higher derivatives of the interface parametrization be computed. So, although it is possible to compute corrections to arbitrarily high order, the practical limit is probably sixth order derivatives.

With the above jump conditions determined, the calculation of the stencils is straightforward. Recalling that standard finite difference stencils can be used for symmetric problems, we assume that the stencil coefficients, $\alpha_{\sigma nk}$, are known in

$$\frac{\partial^n u\,(0)}{\partial^k x \partial y^{n-k}} = \sum_{\sigma=0}^{M} \alpha_{\sigma nk} \left[u\,(\sigma) - C_\sigma \right], \tag{3.192}$$

leaving only the additive corrections to be calculated. Using (3.149) and (3.155) we see that the additive correction associated with an intersected stencil leg is given by

$$C_\sigma = + \sum_{n=0}^{N} \sum_{k=0}^{n} \left[\frac{(x_\sigma - X)^k}{k!} \right] \left[\frac{(y_\sigma - Y)^{n-k}}{(n-k)!} \right] \left[\frac{\partial^n u}{\partial x^k \partial y^{n-k}} \right], \tag{3.193}$$

when $\mathbf{x}_0 \in \Gamma^-$, and by

$$C_\sigma = - \sum_{n=0}^{N} \sum_{k=0}^{n} \left[\frac{(x_\sigma - X)^k}{k!} \right] \left[\frac{(y_\sigma - Y)^{n-k}}{(n-k)!} \right] \left[\frac{\partial^n u}{\partial x^k \partial y^{n-k}} \right], \tag{3.194}$$

when $\mathbf{x}_0 \in \Gamma^+$. The additive corrections are

$$C_\sigma = 0, \tag{3.195}$$

when the stencil leg connecting $\mathbf{x}_0$ to $\mathbf{x}_\sigma$ does not intersect the interface. The calculation of the jump conditions for the general non-symmetric case is significantly more involved.

Non-symmetric jump conditions For the non-symmetric Poisson problem we seek jump conditions of the form (3.110) and (3.111). We begin by deriving jump conditions for the first derivatives. From (3.168) and the derivative of (3.167) we have

$$\tau_x u_x^+ + \tau_y u_y^+ - \tau_x u_x^- - \tau_y u_y^- = [u]_s\,, \tag{3.196}$$

and

$$n_x h^+ u_x^+ + n_y h^+ u_y^+ - n_x h^- u_x^- - n_y h^- u_y^- = [h u_n]\,. \tag{3.197}$$

We can solve (3.196) and (3.197) for either u_x^+ and u_y^+ in terms of u_x^- and u_y^-,

$$\begin{aligned} u_x^+ &= \left[\frac{h^- + (h^+ - h^-)\,\tau_x^2}{h^+}\right] u_x^- + \left[\frac{(h^+ - h^-)\,\tau_x \tau_y}{h^+}\right] u_y^- \\ &\quad + \tau_x [u]_s + \frac{n_x}{h^+}[h u_n]\,, \end{aligned} \tag{3.198}$$

$$\begin{aligned} u_y^+ &= \left[\frac{(h^+ - h^-)\,\tau_x \tau_y}{h^+}\right] u_x^- + \left[\frac{h^- + (h^+ - h^-)\,\tau_y^2}{h^+}\right] u_y^- \\ &\quad + \tau_y [u]_s + \frac{n_y}{h^+}[h u_n]\,, \end{aligned} \tag{3.199}$$

which implies that

$$D_{11}^{11-} = \left[\frac{h^- + (h^+ - h^-)\,\tau_x^2}{h^+}\right], \tag{3.200}$$

$$D_{10}^{11-} = \left[\frac{(h^+ - h^-)\,\tau_x \tau_y}{h^+}\right], \tag{3.201}$$

$$D_{01}^{11-} = 0, \tag{3.202}$$

$$D_{00}^{11-} = 0, \tag{3.203}$$

$$D_{11}^{10-} = \left[\frac{(h^+ - h^-)\,\tau_x \tau_y}{h^+}\right], \tag{3.204}$$

$$D_{10}^{10-} = \left[\frac{h^- + (h^+ - h^-)\,\tau_y^2}{h^+}\right], \tag{3.205}$$

$$D_{01}^{10-} = 0, \tag{3.206}$$

$$D_{00}^{10-} = 0, \tag{3.207}$$

$$J_{11}^{-} = \tau_x [u]_s + \frac{n_x}{h^+} [hu_n], \tag{3.208}$$

$$J_{10}^{-} = \tau_y [u]_s + \frac{n_y}{h^+} [hu_n], \tag{3.209}$$

or u_x^- and u_y^- in terms of u_x^+ and u_y^+,

$$u_x^- = \left[\frac{h^+ + (h^- - h^+)\tau_x^2}{h^-}\right] u_x^+ + \left[\frac{(h^- - h^+)\tau_x\tau_y}{h^-}\right] u_y^+ \tag{3.210}$$
$$-\tau_x [u]_s - \frac{n_x}{h^-} [hu_n],$$

$$u_y^- = \left[\frac{(h^- - h^+)\tau_x\tau_y}{h^-}\right] u_x^+ + \left[\frac{h^+ + (h^- - h^+)\tau_y^2}{h^-}\right] u_y^+ \tag{3.211}$$
$$-\tau_y [u]_s - \frac{n_y}{h^-} [hu_n],$$

which implies that,

$$D_{11}^{11+} = \left[\frac{h^+ + (h^- - h^+)\tau_x^2}{h^-}\right], \tag{3.212}$$

$$D_{10}^{11+} = \left[\frac{(h^- - h^+)\tau_x\tau_y}{h^-}\right], \tag{3.213}$$

$$D_{01}^{11+} = 0, \tag{3.214}$$

$$D_{00}^{11+} = 0, \tag{3.215}$$

$$D_{11}^{10+} = \left[\frac{(h^- - h^+)\tau_x\tau_y}{h^-}\right], \tag{3.216}$$

$$D_{10}^{10+} = \left[\frac{h^+ + (h^- - h^+)\tau_y^2}{h^-}\right], \tag{3.217}$$

$$D_{01}^{10+} = 0, \tag{3.218}$$

$$D_{00}^{10+} = 0, \tag{3.219}$$

$$J_{11}^{+} = -\tau_x [u]_s - \frac{n_x}{h^-} [hu_n], \tag{3.220}$$

$$J_{10}^{+} = -\tau_y [u]_s - \frac{n_y}{h^-} [hu_n]. \tag{3.221}$$

Note that the expressions for the "+" and "-" derivative coefficients simply have the "+" and "-" labels exchanged. We can determine the jump conditions for the second derivatives by solving

$$u_{xx}^{+} - u_{xx}^{-} + u_{yy}^{+} - u_{yy}^{-} = [f] \tag{3.222}$$

and the equations resulting from taking the derivative of (3.196) and (3.197) with respect to arc length for u_{xx}^{+}, u_{xy}^{+}, and u_{yy}^{+} in terms of u_{xx}^{-}, u_{xy}^{-}, and u_{yy}^{-} and vice-versa. The resulting expressions are straightforward to determine, but too lengthy to reproduce here. Although the expressions become exceedingly long, it possible to continue on to higher order derivatives as well. For instance, to determine the jump conditions for the third derivatives, we would solve the equations resulting from taking the second derivative in arc length of (3.196) and (3.197), the first derivative in arc length of (3.222) and the jump in the normal derivative of the Poisson equation,

$$n_x \left(u_{xxx}^{+} - u_{xxx}^{-}\right) + n_y \left(u_{xxy}^{+} - u_{xxy}^{-}\right) + n_x \left(u_{xyy}^{+} - u_{xyy}^{-}\right) + n_y \left(u_{yyy}^{+} - u_{yyy}^{-}\right) = [f_n], \tag{3.223}$$

for all the "+" derivatives in terms of the "-" derivatives and vice versa. Using the known jump conditions, we can then construct the stencil coefficients and additive corrections for each intersected stencil as outlined above.

3.4 Temporal derivatives

In this section we will develop a family of time stepping schemes that can accurately compute the evolution of a function which experiences a discontinuity in its time derivative. This is of interest because, as we shall demonstrate later, the time derivative of the temperature of a solidifying material is discontinuous across the interface. We can demonstrate why a special scheme is required, however, by examining a much simpler problem.

Consider the problem of numerically computing the value of $\theta(t)$ which satisfies

$$\frac{d\theta}{dt} = g(t), \tag{3.224}$$

$$\theta(0) = \theta_0 \tag{3.225}$$

where

$$g = \begin{cases} 0 & t < t_0 \\ a + bt & t \geq t_0 \end{cases} . \tag{3.226}$$

The exact solution of this problem is

$$\theta = \begin{cases} \theta_0 & t < t_0 \\ \theta_0 + a\left(t - t_0\right) + \frac{1}{2} b \left(t^2 - t_0^2\right) & t \geq t_0 \end{cases} . \tag{3.227}$$

If we remove the jump in the time derivative ($t_0 < 0$), any second order accurate time stepping scheme will reproduce the exact solution to within machine precision. Specifically, using

$$\theta^{n+1} = \theta^n + \Delta t \left(\alpha g^{n+1} + \left(1 - \alpha\right) g^n\right) \tag{3.228}$$

with $\theta^n = \theta\left(n\Delta t\right)$, $g^n = g\left(n\Delta t\right)$ and $\alpha = \frac{1}{2}$ reproduces the exact solution. As demonstrated in Fig. 3.1, however, when the time derivative is discontinuous during the simulation ($t_0 = 1/3$ in this case), the numerical solution is no longer exact. Furthermore, the error is constant over all time steps during which the time derivative is continuous. Thus, it appears that the time step during which the discontinuity occurs is solely responsible for the error in the solution. This is not surprising since there is no information in the numerical scheme concerning the exact time at which the forcing (i.e., the time derivative) suffers the discontinuity. We now show that this information is required to accurately compute the solution to problems with discontinuous time derivatives.

We are interested in correcting the family of time stepping schemes of the form

$$\theta^{n+1} = \theta^n + \Delta t \left\{\alpha \theta_t^{n+1} + \left(1 - \alpha\right) \theta_t^n\right\}, \tag{3.229}$$

where $\alpha \in [0, 1]$. The above can be considered a discretization of

$$\theta^{n+1} = \theta^n + \int_0^{\Delta t} \theta_t \left(t_n + t\right) dt. \tag{3.230}$$

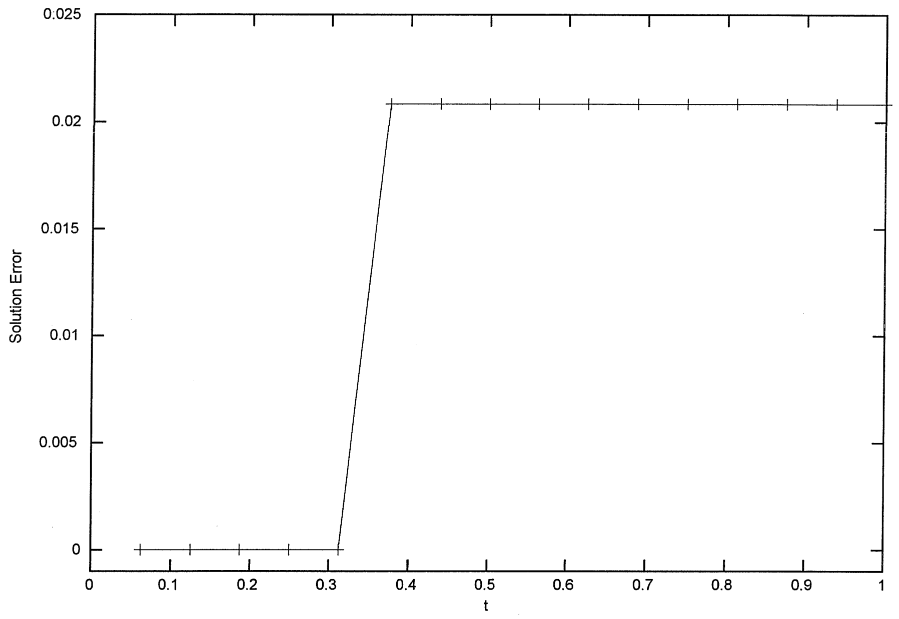

Figure 3.1: Error of the uncorrected discrete approximation using second order accurate time stepping when $t_0 = 1/3$

We will focus our attention on accurately approximating integrals of the form

$$I = \int_0^{\Delta t} g(t)\, dt, \tag{3.231}$$

where the integrand is everywhere bounded and smooth except at $t = t_I$, $0 < t_I \leq \Delta t$, where it can have discontinuities in its value and derivative,

$$\begin{aligned} [g] &= g\left(t_I^+\right) - g\left(t_I^-\right) = g^+ - g^-, & (3.232)\\ [g_t] &= g_t\left(t_I^+\right) - g_t\left(t_I^-\right) = g_t^+ - g_t^-. & (3.233) \end{aligned}$$

We assume that the values of $[g]$ and $[g_t]$ are known. Our goal is to develop discrete approximations to (3.231) that depend only on $[g]$, $[g_t]$ and one or both of $g(0)$ and $g(\Delta t)$. Because of the discontinuity, we employ separate approximations for g over the intervals before and after $t = t_I$. In particular, we use

$$g(t) = \begin{cases} g(0) + t g_t(0) & 0 \leq t \leq t_I \\ g(\Delta t) + (t - \Delta t) g_t(\Delta t) & t_I < t \leq \Delta t \end{cases} + O\left(\Delta t^2\right), \tag{3.234}$$

which consists of truncated Taylor series for g expanded around the beginning and end of the integration interval. The above is a second order accurate approximation to $g(t)$ which, after substitution into (3.231), yields

$$\begin{aligned} I &= \int_0^{t_I} \left(g(0) + t g_t(0)\right) dt + \int_{t_I}^{\Delta t} \left(g(\Delta t) + (t - \Delta t) g_t(\Delta t)\right) dt + O\left(\Delta t^3\right), \\ I &= t_I g(0) + \frac{t_I^2}{2} g_t(0) + (\Delta t - t_I) g(\Delta t) - \frac{(\Delta t - t_I)^2}{2} g_t(\Delta t) + O\left(\Delta t^3\right). \end{aligned} \tag{3.235}$$

Now we develop an expression for I that depends only on g evaluated at $t = \Delta t$. This means the $g(0)$ and $g_t(0)$ terms must be replaced. Noting that these quantities can be approximated by

$$g(0) = g^- - t_I g_t^- + O\left(\Delta t^2\right), \tag{3.236}$$

$$g_t(0) = g_t^- + O(\Delta t), \tag{3.237}$$

we can express them in terms of quantities evaluated on the same side of the jump as $t = \Delta t$. Using the jump conditions, (3.232) and (3.233),

$$g(0) = g^+ - t_I g_t^+ - [g] + t_I [g_t] + O\left(\Delta t^2\right), \tag{3.238}$$

$$g_t(0) = g_t^+ - [g_t] + O(\Delta t), \tag{3.239}$$

and expanded g^+ and g_t^+ in a truncated Taylor series,

$$g^+ = g(\Delta t) + (t_I - \Delta t) g_t(\Delta t) + O\left(\Delta t^2\right), \tag{3.240}$$

$$g_t^+ = g_t(\Delta t) + O(\Delta t), \tag{3.241}$$

we find

$$\begin{aligned} g(0) &= g(\Delta t) - (\Delta t)\, g_t(\Delta t) - [g] + t_I [g_t] + O\left(\Delta t^2\right), && (3.242) \\ g_t(0) &= g_t(\Delta t) - [g_t] + O(\Delta t). && (3.243) \end{aligned}$$

Using (3.242) and (3.243) in (3.235) yields

$$I = (\Delta t)\, g(\Delta t) - \frac{\Delta t^2}{2} g_t(\Delta t) - t_I [g] + \frac{t_I^2}{2} [g_t] + O\left(\Delta t^3\right), \qquad (3.244)$$

which does not depend on $g(0)$. Comparing (3.230) and (3.231) we see that (3.244) enables us to use the backward Euler scheme,

$$\theta^{n+1} = \theta^n + (\Delta t)\, \theta_t^{n+1} - t_I [\theta_t], +O\left(\Delta t^2\right) \qquad (3.245)$$

even if the time derivative jumps discontinuously during the course of a time step.

Next, we develop an expression for I that depends only on g evaluated at $t = 0$. This means the $g(\Delta t)$ and $g_t(\Delta t)$ terms must be replaced. Noting that these quantities can be approximated by

$$\begin{aligned} g(\Delta t) &= g^+ + (\Delta t - t_I)\, g_t^+ + O\left(\Delta t^2\right), && (3.246) \\ g_t(\Delta t) &= g_t^+ + O(\Delta t), && (3.247) \end{aligned}$$

we can express them in terms of quantities evaluated on the same side of the jump as $t = 0$. Using the jump conditions, (3.232) and (3.233),

$$\begin{aligned} g(\Delta t) &= g^- + (\Delta t - t_I)\, g_t^- + [g] + (\Delta t - t_I)[g_t] + O\left(\Delta t^2\right), && (3.248) \\ g_t(\Delta t) &= g_t^- + [g_t] + O(\Delta t), && (3.249) \end{aligned}$$

and expanding g^- and g_t^- in a truncated Taylor series,

$$g^- = g(0) + t_I g_t(0) + O\left(\Delta t^2\right), \qquad (3.250)$$

$$g_t^- = g_t(0) + O(\Delta t), \tag{3.251}$$

we find

$$g(\Delta t) = g(0) + \Delta t g_t(0) + [g] + (\Delta t - t_I)[g_t] + O\left(\Delta t^2\right), \tag{3.252}$$

$$g_t(\Delta t) = g_t(0) + [g_t] + O(\Delta t). \tag{3.253}$$

Using (3.252) and (3.253) in (3.235) yields

$$I = (\Delta t) g(0) + \left(\frac{\Delta t^2}{2}\right) g_t(0) + (\Delta t - t_I)[g] + \left(\frac{(\Delta t - t_I)^2}{2}\right)[g_t] + O\left(\Delta t^3\right), \tag{3.254}$$

which does not depend on $g(\Delta t)$. Comparing (3.230) and (3.231) we see that (3.254) enables us to use the forward Euler scheme,

$$\theta^{n+1} = \theta^n + (\Delta t)\theta_t^n + (\Delta t - t_I)[\theta_t], + O\left(\Delta t^2\right) \tag{3.255}$$

even if the time derivative jumps discontinuously during the course of a time step.

We can develop a general scheme by averaging the above results together. Adding α times (3.244) with $(1-\alpha)$ times (3.254) yields

$$\begin{aligned} I &= \Delta t\left[(1-\alpha) g(0) + \alpha g(\Delta t)\right] + \left[(1-\alpha)(\Delta t - t_I) - \alpha t_I\right][g] \\ &+ \frac{\Delta t^2}{2}\left[(1-\alpha) g_t(0) - \alpha g_t(\Delta t)\right] + \left[(1-\alpha)\frac{(\Delta t - t_I)^2}{2} + \alpha\frac{t_I^2}{2}\right][g_t]. \end{aligned} \tag{3.256}$$

When $\alpha \neq \frac{1}{2}$, the above is first order accurate and the higher order terms can be neglected. The corrected first order scheme is

$$I = \Delta t\left[(1-\alpha) g(0) + \alpha g(\Delta t)\right] + \left[(1-\alpha)(\Delta t - t_I) - \alpha t_I\right][g] + O\left(\Delta t^2\right). \tag{3.257}$$

For the $\alpha = \frac{1}{2}$ second order case, we use (3.253) to eliminate the g_t terms and find the corrected second order scheme,

$$I = \frac{\Delta t}{2}\left[g(0) + g(\Delta t)\right] + \left(\frac{1}{2}\Delta t - t_I\right)[g] - \frac{1}{2}t_I(\Delta t - t_I)[g_t]. \tag{3.258}$$

Equating g to θ_t we see that we have developed a corrected time stepping scheme of the desired form,

$$\theta^{n+1} = \theta^n + \Delta t\left[\alpha\theta_t^{n+1} + (1-\alpha)\theta_t^n\right] + E, \tag{3.259}$$

where

$$E = \begin{cases} \left[(1-\alpha)(\Delta t - t_I) - \alpha t_I\right][\theta_t] & \alpha \neq \frac{1}{2} \\ \left(\frac{1}{2}\Delta t - t_I\right)[\theta_t] - \frac{1}{2}t_I(\Delta t - t_I)[\theta_{tt}] & \alpha = \frac{1}{2} \end{cases}. \tag{3.260}$$

Clearly if the time derivative is continuous during a time step, $[\theta_t] = [\theta_{tt}] = 0$ and correction is unnecessary.

Now we return to the simple problem given by (3.224)-(3.226). In order to use the above scheme, we need to know the jumps in the time derivatives, $[\theta_t]$ and $[\theta_{tt}]$. For this simple test problem, these values come directly from the definition of g

$$[\theta_t] = a + bt_0, \tag{3.261}$$

$$[\theta_{tt}] = b. \tag{3.262}$$

Using these jumps and the above time stepping method, the numerical solution to (3.224)-(3.226) is exact when $\alpha = \frac{1}{2}$ or when $b = 0$ and $\theta \in [0, 1]$, regardless of the time that the discontinuity in the forcing occurs. In general, we can not expect to obtain exact solutions, only approximations that converge at a first order ($\alpha \neq \frac{1}{2}$) or second order ($\alpha = \frac{1}{2}$) rate.

When discretizing partial differential equations, it is quite common to use different time marching schemes on different terms in the equation. For instance, the non-linear convection terms in the Navier-Stokes equations are often treated explicitly even if an implicit scheme is applied to the viscous terms. Suppose we wish to evolve

$$\frac{\partial\theta}{\partial t} = F(t) + B(t) + T(t) \tag{3.263}$$

forward in time using forward Euler on F, backward Euler on B and the trapezoidal rule on T. We can view this scheme as the sum of three integrals,

$$\theta^{n+1} = \theta^n + \int_0^{\Delta t} F dt + \int_0^{\Delta t} B dt + \int_0^{\Delta t} T dt, \tag{3.264}$$

each of which can be corrected using the formulas developed above. Unfortunately, deriving the corrections can be difficult for such schemes. Often, it is easy to calculate

$$[\theta_t] = [F] + [B] + [T] \tag{3.265}$$

for a given equation. When a different time stepping scheme is applied to separate terms, however, we need to calculate $[F]$, $[B]$, $[T]$ and $[T_t]$ individually. Depending on the equation being considered, this may be quite challenging. We shall see a simple example of how this can done later, when we discuss the simulation of dendritic solidification in the presence of natural convection.

Chapter 4 Solidification without convection

4.1 Introduction

In the real world there is almost always a buoyancy induced flow in liquids that are not at a uniform temperature. Thus, the temperature gradients in a liquid undergoing dendritic solidification tend to produce fluid motion. Unfortunately, including convection in a mathematical phase change model is quite difficult so, traditionally, the effects of fluid motion have been ignored. In this chapter we develop a numerical method capable of simulating dendritic solidification problems in one and two dimensions when convection in the liquid phase is neglected.

We begin the chapter with a survey of different numerical methods that researchers have developed to simulate dendritic solidification. We then review the mathematical model that we will use to simulate dendritic solidification in the absence of convection. Focusing on one-dimensional problems at first, we demonstrate how the generalized immersed interface discretization, developed in Chapter 3, can be used to solve a non-symmetric dendritic solidification problem. The main step in this process is the derivation of the jump conditions on the spatial and temporal derivatives. An interpolation scheme is then developed based on the immersed interface stencils. The scheme is required so that the temperature on the interface can be determined and the Gibbs-Thomson condition can be enforced. The interpolation method is notable because it produces an interpolant that varies in a continuous manner as the interface is moved through the mesh. We wrap up our one-dimensional study by presenting the complete algorithm and validating the scheme using the exact solutions discussed in Chapter 2. A comparison of results generated by our method and the immersed boundary method of Juric (see [29] and below) is also presented. Moving onto two-dimensional problems, we discuss how our method tracks the interface and determines the phase of the computational nodes. Restricting our attention to symmetric problems, we then calculate the jump conditions on the spatial and temporal derivatives. After a quick discussion of the discretization of the governing equations, we extend our interpolation scheme to two dimensions. We wrap up our two-dimensional study by presenting the complete algorithm and validating the approach using the approximate solutions determined using linear perturbations in Chapter 2. A comparison between the performance of our method and the immersed boundary method is also done. In addition, we investigate the effect of surface tension anisotropy by performing simulations starting with the same initial conditions for three different materials. After a short summary, we close the chapter with a few suggestions for future work.

4.2 Existing numerical methods

The simulation of dendritic solidification (i.e., solidification into an undercooled liquid) is quite challenging. The unstable nature of the interface requires that methods be robust enough to compute the temperature in domains that are evolving and typically quite deformed. Even so, there is a rich body of literature dedicated to the development of numerical schemes capable of simulating the purely diffusive unstable solidification problem. An exhaustive survey of all previous work in this area would be quite lengthy indeed. Instead, we will discuss a number of works that are representative of the different general approaches which researchers have used to study the problem.

The enthalpy method is an extremely popular technique for simulating stable solidification (i.e., solidification into a liquid above its freezing temperature). Several different formulations exist (see [75]), but the basic idea is to rewrite the heat equation in terms of enthalpy, ζ,

$$\frac{\partial \zeta}{\partial t} = \left(\frac{K}{\rho c}\right) \nabla^2 T, \tag{4.1}$$

where, for stable solidification, ζ and T are related by

$$T = \begin{cases} \dfrac{\zeta}{c} & \zeta < cT_M \\ T_M & cT_M \leq \zeta \leq T_M + L \\ \dfrac{(\zeta - L)}{c} & \zeta > cT_M + L \end{cases} . \tag{4.2}$$

Once the enthalpy is determined at the next time step, the new temperature is deduced from (4.2). The method can be implemented on a fixed grid with no need to construct special stencils near the interface. In practice, these methods have difficultly dealing with problems that have a sharp melting temperature, so it is common to assume that the phase change occurs over a range of temperatures, see [75] for details. Another difficulty with these methods is that they are rather inaccurate, with an error scaling like $O\left(\Delta t \log\left(1 + T/\Delta t\right)^{1/2}\right)$, where T is the time the error is bounded (see [63]). An advantage of these schemes is that they do not require explicit knowledge of the interface location. For stable solidification problems, the position of the interface can be deduced from the temperature field because it is a level set of the temperature (i.e., the interface is located at all points where $T = T_M$). If the melting temperature of the substance depends on the geometry of the interface, however, it is necessary to have explicit knowledge of the interface location and geometry to properly modify and evaluate (4.2). Unfortunately, because of the Gibbs-Thomson condition, this is exactly the case we have for the dendritic solidification problem. It is still possible to construct an enthalpy based scheme for dendritic solidification, but it is necessary to track the interface either explicitly (see [61]) or through a fractional volume representation (see [14]). Both Smith and Chorin were able to simulate some simple dendritic solidification problems, but they were unable to produce agreement with linear stability theory and anisotropy due to grid orientation was reported.

The immersed boundary method (see [29]) is another example of a fixed grid scheme for dendritic solidification. It also requires that the interface be explicitly tracked. For this method, the heat equation is augmented by a singular forcing term,

Q,

$$\frac{\partial (\rho c T)}{\partial t} = \nabla \cdot K \nabla T + Q \tag{4.3}$$

that accounts for the release of latent heat at the interface during solidification. The heat source, in two dimensions, is given by

$$Q = L \int V_N (q,t) \, \delta (x - X (q,t) , y - Y (q,t)) \, dq, \tag{4.4}$$

where δ is a two-dimensional delta function and the normal velocity and the interface, parametrized by q, are given by $V_N (q,t)$ and $(X (q,t) , Y (q,t))$, respectively. The solid and liquid phases are treated as a single substance with varying material properties, and (4.3) is discretized on a fixed finite difference grid. The material properties in (4.3) are given by

$$K = K_L + (K_S - K_L) \, I (x,y) , \tag{4.5}$$

$$\rho c = \rho_L c_L + (\rho_S c_S - \rho_L c_L) \, I (x,y) , \tag{4.6}$$

where I, the indicator function, is constructed such that it is equal to one in the solid, zero in the liquid and smoothly varies between these values over several mesh spacings (typically 3 or 4) in the region adjacent to the interface (see [29] for details). We will restrict our attention to symmetric problems from now on so that consideration of the indicator function is unnecessary. In order to discretize (4.3), it is necessary to have a discrete representation of (4.4). This requires knowledge of the interface location and a discrete representation for the delta function, δ. The interface is explicitly tracked by a series of marker particles, $(X (i) , Y (i))$, that are moved through the domain at the local normal velocity, $V_N (i)$, of the interface. These markers are connected together to form an explicit parametrization for both the interface, (X, Y), and the normal velocity, V_N, whose values are also stored at the markers. The delta function is approximated by

$$\delta (x,y) = d (x) \, d (y) , \tag{4.7}$$

where

$$d\left(x\right)=\begin{cases}\dfrac{1}{\left(4\Delta x\right)}\left(1+\cos\left(\dfrac{\pi x}{2\Delta x}\right)\right) & \text{if } \left|x\right|<2\Delta x \\ 0 & \text{otherwise}\end{cases}. \tag{4.8}$$

Assuming the normal velocity of the interface is known, it is easy to evaluate (4.4) and solve (4.3) for the new temperature. With the exception of the first time step, however, the normal velocity of the interface is not known and must be determined as part of the solution. It is straightforward to show that solutions of (4.3) and (4.4) satisfy the heat equation in each phase and the jump condition on the heat flux at the interface. There is nothing in the above, however, to enforce the Gibbs-Thomson condition. Indeed, this extra constraint is required in order to determine the motion of the interface. The Gibbs-Thomson condition is satisfied by iterating on the normal velocity, $V_N\left(i\right)$, until the error

$$e\left(i\right)=T\left(i\right)-T_M\left(1-\frac{\kappa\left(i\right)}{L}\gamma\right), \tag{4.9}$$

at each marker particle, $\left(X\left(i\right),Y\left(i\right)\right)$, is sufficiently small. The temperature at the marker particles, $T\left(i\right)=T\left(X\left(i\right),Y\left(i\right)\right)$, is found by interpolation off the finite difference grid using the sampling property of the delta function

$$T\left(i\right)=\left(\Delta x^2\right)\sum_{a,b}T\left(x_a,y_b\right)\delta\left(X\left(i\right)-x_a,Y\left(i\right)-y_b\right), \tag{4.10}$$

where $\left(x_a,y_b\right)$ is the location of a computational node. The interface curvature, $\kappa\left(i\right)$, is determined using a fourth order polynomial passing through the marker in question and its two neighbors on each side. The complete algorithm can be found in [29]. The immersed boundary method is quite robust and is capable of tracking solidification interfaces of impressive complexity. We will demonstrate later, however, that the immersed boundary method is only first order accurate and has significant difficulty agreeing with linear stability theory.

The finite element method is widely applied to problems in which the domain of interest is irregularly shaped. In the case of dendritic solidification, the domain is both irregularly shaped and deforming in time. In the deforming finite element approach, the computational mesh is allowed to deform as the interface moves in such a way that the individual elements continually maintain their identity as exclusively solid or exclusively liquid elements. It is difficult to deform the mesh of an entire domain without causing highly skewed elements, however, so frequent regridding of the domain is required. There are two basic approaches that can be employed when simulating dendritic solidification with the deforming finite element method. In each case it is necessary to iterate on the normal velocity of the interface. The difference lies in which interface constraint is incorporated into the discretization directly and which is left to be satisfied by the proper selection of the normal velocity. In the first approach (see [44] and [45]), the Gibbs-Thomson condition,

$$T = T_M \left(1 - \frac{\kappa}{L}\gamma\right), \tag{4.11}$$

is used to fix the value of finite element nodes that lie on the interface. For arbitrary values of the normal velocity, this leaves the condition on the jump in the heat flux at the interface,

$$(K_S \nabla T_S - K_L \nabla T_L) \cdot \mathbf{n} = \rho_S L \left(\frac{\partial \mathbf{X}}{\partial t} \cdot \mathbf{n} \right), \tag{4.12}$$

unsatisfied. These schemes compute the jump in the heat flux directly and, in effect, use (4.12) as an error condition for an iterative scheme to determine the normal velocity of the interface. The second approach (see [66] and [67]) is essentially a finite element discretization of (4.3). The advantage of the finite element method is that it is unnecessary to explicitly discretize the delta function since it is eliminated during the course of the standard discretization (i.e., by the inner product between the (4.3) and the finite element bases). In this way, the normal velocity is explicitly introduced into the discrete system for the temperature and (4.12) is automatically satisfied. Lynch and Sullivan have compared the accuracy of these two different formulations and found that the second is an order of magnitude more accurate (see [46]). They show that this is a result of the conservation of heat which is only exact for the formulation based on (4.3) because the heat equation is explicitly applied everywhere in the domain. In [66], a simulation of dendritic solidification using the deforming finite element method was shown to agree with linear stability theory. In [67], it is demonstrated that this method is capable of simulating the evolution of modestly deformed dendrites.

Another scheme based on the finite element method was recently presented in [58]. It requires a slightly more general form of the Gibbs-Thomson condition,

$$T = T_M\left(1 - \frac{\kappa}{L}\gamma - \frac{\mu}{L}V_N\right), \tag{4.13}$$

where μ is introduced to account for the finite rate at which molecules can attach to a solidifying surface, and V_N is the normal velocity of the interface. Schmidt explicitly tracks the interface and discretizes the domain using finite elements but the mesh is not aligned with the interface. That is, some of the elements in the mesh are intersected by the interface and contain both solid and liquid. In order to avoid ambiguity, the method is only applied to symmetric problems for which the material properties of the liquid and solid are identical. The method discretizes (4.3) so that both the heat equation and the jump in normal heat flux are automatically satisfied, as discussed above. The value of the temperature on the interface is not forced to satisfy the Gibbs-Thomson equation, however. Instead, the temperature is treated as a known forcing (i.e., the value from the previous time step is used) and (4.13) is solved as an independent equation for the new interface position. Thus the system is evolved without ever having to iterate on the temperature field, which, in general, is quite expensive. While this approach is limited to symmetric problems and non-zero values of μ, it is capable of simulating solidification in three dimensions and has produced some extremely impressive results.

In [13],Chen and her co-workers recently developed a level set method capable of simulating dendritic solidification problems. The idea behind level set methods is to implicitly track the position of the interface by introducing a new field variable, ϕ, that is defined to be the signed distance from the interface ($\phi > 0$ in the liquid and $\phi < 0$ in the solid) everywhere in the domain. By definition, the interface is the level set $\phi = 0$. The position of the interface is updated through the solution of

$$\frac{\partial \phi}{\partial t} + \mathbf{u} \cdot \boldsymbol{\nabla} \phi = 0, \tag{4.14}$$

where, on the interface, $\mathbf{u}$ is the normal velocity of the interface and away from the interface it is an ad hoc extension of the interface velocity. For this method, the interface velocity is calculated directly by computing the normal derivative on each side of the interface with one-sided stencils (i.e., stencils that only use nodes from a single phase) and using (4.12). A viable extension of the velocity, $\mathbf{u}$, for the level set is then constructed by solving four additional advection equations (consult [13] for details). The temperature is discretized using standard second order accurate finite difference stencils except at stencils cut by the interface. For any node close enough to the interface that the standard stencil would involve both liquid and solid neighbors, special second order accurate one-sided stencils are constructed. These stencils treat the interface as a Dirchlet (T known) boundary using (4.11) or (4.13) to determine the value there. Because of the construction of $\mathbf{u}$ and the interface stencils, the above solution satisfies all of the equation necessary for the simulation of dendritic solidification and is repeated at each time step. The authors claim that the method agrees with linear stability theory and demonstrate that it is capable of simulating moderately deformed dendrites.

Like Chen, Udaykumar and Shyy (see [70]) use a one-sided stencil approach. In their method, however, the interface is tracked explicitly using a series of discrete marker particles. The heat equation is discretized on an almost uniform grid using a control volume formulation. The control volumes intersected by the interface (i.e., containing both liquid and solid) are locally deformed so that they enclose only one phase. An approximation to the normal derivative is required on each side of the discrete control volumes in this formulation. This is problematic because one side of the deformed control volumes typically lies along the interface where standard differencing can not be employed. The normal derivative on these faces is determined using a one-sided stencil with the temperature on the interface itself determined from the Gibbs-Thomson condition. The interface markers are moved at a normal velocity calculated directly using (4.12) where estimates to the normal derivative mentioned above are utilized. This procedure satisfies all the governing equations and allows the solution to be advanced forward in time. The authors do not test their method against linear theory but do compare it against an exact one-dimensional solution and report good accuracy. They do report, however, that their growing dendrites exhibit a lack of symmetry even at early stages in their evolution. This is most likely the result of numerical perturbations introduced by their one-sided stencils (see the discussion below).

A number of researchers have approached the dendritic solidification problem by casting it in terms of an integral equation. A significant advantage of this formulation is that the dimension of the problem is reduced by one since only quantities on the interface are required to evolve the system forward. Traditional integral formulations of the problem are costly due to the need to evaluate a "memory integral" that accounts for the evolution of the latent heat that has been released since the start of the simulation. Due to this difficulty most investigations using this approach have utilized a traveling reference frame and sought steady state solutions (see [48] and [49]). Strain, however, has studied the time dependent evolution of a perturbed flat interface using this approach. He reports agreement with linear theory and demonstrates the ability of the method to produce experimentally observed phenomena such as tip splitting. The method is expensive, however, with a cost per time step that increases linearly. Recently researchers have developed clever approaches which eliminate the need to evaluate the memory integral directly. These "fast" methods have a fixed cost per time step (see [7] and [59]). Sethian and Strain combined a "fast" integral formulation with the level set method discussed above and demonstrated their method was capable of simulating problems with moderately deformed interfaces.

Simulations of dendritic solidification based on phase field methods have become very popular in recent years (see [8], [11], [17], [18], [30], [53] and [76]). The method uses an order parameter φ which is a new field variable defined everywhere in the domain. The order parameter is a solution to a reaction-diffusion equation which takes on different constant values in each phase ($\varphi = 1$ in the solid and $\varphi = -1$ in the liquid is a common but not universal choice) and has a region of rapid variation in the vicinity of the interface. The heat equation, written in terms of the dimensionless temperature θ, is coupled to φ by

$$\frac{\partial \theta}{\partial t} = \nabla^2 \theta + \frac{1}{2}\frac{\partial \varphi}{\partial t}, \tag{4.15}$$

for the $\varphi = \pm 1$ choice mentioned above. Because the order parameter is essentially a smoothed out step function, its time derivative effectively introduces a smoothed out delta function equivalent to (4.4). The phase field method has been controversial since its inception because the standard interface boundary conditions are never explicitly enforced. The method has been shown to be asymptotically equivalent to the standard sharp interface dendritic solidification model in the limit as the interface thickness goes to zero, however, and research into its properties continues. Although the results are not fully resolved, recent three-dimensional calculations by Kobayashi (see [33]) and Karma and Rappel (see [30]) demonstrate the significant promise of this approach.

Finally, some researchers have taken completely unconventional approaches to simulating dendritic solidification. Roosen and Taylor in [55], for instance, simplify the problem by restricting the interface to a polygon and assume non-zero kinetic mobility (i.e., $\mu \neq 0$ in (4.13)). They produce some impressive results but report a grid induced anisotropy. Algren, on the other hand, replaces the standard model by a variational formulation in which the Gibbs-Thomson condition is enforced through a balance between bulk and surface energies (see [3]). He reports agreement with linear theory but the error of the algorithm only scales like $O\left(\Delta t^{1/2}\right)$.

4.3 Governing equations

We will consider problems that are periodic in the horizontal (x) direction with an x-periodic interface dividing the domain into a solid region below the interface and a liquid region above it (see Fig. 4.1). The scaled equations governing the dendritic

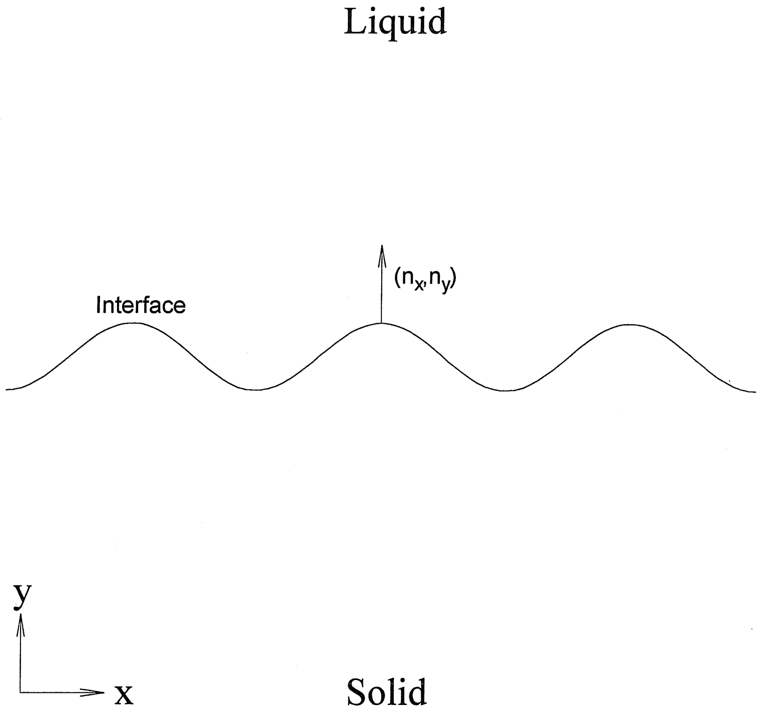

Figure 4.1: Typical Interface Geometry

solidification of a pure substance in the absence of convection are

$$\frac{\partial \theta_L}{\partial t} = H_L \nabla^2 \theta_L, \tag{4.16}$$

which applies in the liquid phase and

$$\frac{\partial \theta_S}{\partial t} = H_S \nabla^2 \theta_S, \tag{4.17}$$

which applies in the solid. On the interface the temperature must satisfy

$$\left(h_S \nabla \theta_S - h_L \nabla \theta_L\right) \cdot \mathbf{n} = n_x \frac{\partial X}{\partial t} + n_y \frac{\partial Y}{\partial t}, \tag{4.18}$$

and

$$\theta_S = \theta_L = -\varepsilon_c\left(\mathbf{n}\right)\kappa, \tag{4.19}$$

where the interface is given by $\left(X\left(q,t\right), Y\left(q,t\right)\right)$, where q is a parametric variable. Several of the above quantities depend on the interface geometry. These are the interface normal,

$$n_x = \sin\left(\phi\right) = -\frac{\partial Y}{\partial q}\left[\left(\frac{\partial X}{\partial q}\right)^2 + \left(\frac{\partial Y}{\partial q}\right)^2\right]^{-\frac{1}{2}}, \tag{4.20}$$

$$n_y = \cos\left(\phi\right) = \frac{\partial X}{\partial q}\left[\left(\frac{\partial X}{\partial q}\right)^2 + \left(\frac{\partial Y}{\partial q}\right)^2\right]^{-\frac{1}{2}}, \tag{4.21}$$

the interface curvature,

$$\kappa = \left[\frac{\partial^2 X}{\partial q^2}\frac{\partial Y}{\partial q} - \frac{\partial^2 Y}{\partial q^2}\frac{\partial X}{\partial q}\right]\left[\left(\frac{\partial X}{\partial q}\right)^2 + \left(\frac{\partial Y}{\partial q}\right)^2\right]^{-\frac{3}{2}}, \tag{4.22}$$

and the capillary parameter,

$$\varepsilon_c\left(\phi\right) = \delta_0\left[1 + \delta_1 \sin^2\left(\frac{k_A \phi}{2}\right)\right]. \tag{4.23}$$

The above form for the capillary parameter is equivalent to the conventionally assumed form,

$$\varepsilon_c(\phi) = \bar{\varepsilon}_c \left[1 - A_c \cos(k_A \phi)\right], \tag{4.24}$$

if we let

$$\delta_0 = \bar{\varepsilon}_c (1 - A_c) \tag{4.25}$$

and

$$\delta_1 = \frac{2A_c}{1 - A_c} \tag{4.26}$$

where the constant $\bar{\varepsilon}_c$ is directly related to the material properties of the substance and the values of the constants A_c and k_A are selected to model the anisotropy of the material.

4.4 One-dimensional problems

A large variety of different methods are available for discretizing partial differential equations. The discretization of the dendritic solidification problem is complicated by the presence of the interface. We saw in Chapter 2 that the derivatives of the temperature are not continuous across the interface during solidification. In Chapter 3, we developed formulas for finite difference stencils that are applicable to non-smooth functions such as the temperature in this problem. We will use these finite difference stencils to discretize the spatial derivatives.

4.4.1 Discretization of spatial derivatives

The discretization of the spatial derivatives can be broken down into two cases: the discretization of derivatives at regular nodes and the discretization of derivatives at irregular nodes. From a numerical point of view, a node is considered to be regular if all the required finite difference stencils centered at its location do not intersect the interface. That is, if the calculation of the spatial derivatives at a given node, using the standard centered finite difference stencils (see below), depend only on nodes that all lie on the same side of the interface, then that node is defined to be regular. Any node which is not regular is defined to be irregular. We discuss the discretization of the governing equations at regular nodes first.

At all regular nodes the function is smooth in a sufficiently large region surrounding the node that we can ignore the interface. Therefore, the derivatives at regular nodes may be calculated using standard finite difference formulas. In one dimension, we will use the standard formula for the Laplacian given symbolically by

$$\nabla^2 = \frac{1}{\Delta y^2}\begin{bmatrix} 1 & -2 & 1 \end{bmatrix} + O\left(\Delta y^2\right). \tag{4.27}$$

At the irregular nodes, the above stencil can not be used because at least one of the stencil legs spans the interface. Depending on the jump conditions that the temperature satisfies, the calculation of the stencils is either trivial or laborious. In Chapter 3 we found that if a function satisfies symmetric jump conditions, it is possible to use standard stencils with simple additive corrections to discretize its derivatives. As we shall see, dendritic solidification is a symmetric problem when the material properties are the same in both phases. The calculation of appropriate stencils is significantly more complicated and expensive for non-symmetric problems. When the material properties are not identical in each phase, dendritic solidification is an example of a non-symmetric problem. We consider the fully general non-symmetric case only in one space dimension.

Derivation of the jump conditions

In Chapter 3 we developed a method for accurately discretizing the derivatives of functions that have discontinuities in their values and/or derivatives across an interface. Restricting our attention to one-dimensional problems, recall that the theory applies to functions that satisfy jump conditions of the form

$$\left(\frac{\partial^m \theta}{\partial y^m}\right)^+ = \sum_{k=0}^{m} D_k^{m-} \left(\frac{\partial^k \theta}{\partial y^k}\right)^- + J_m^-, \tag{4.28}$$

and

$$\left(\frac{\partial^m \theta}{\partial y^m}\right)^- = \sum_{k=0}^{m} D_k^{m+} \left(\frac{\partial^k \theta}{\partial y^k}\right)^+ + J_m^+, \tag{4.29}$$

where, if the interface is located at Y,

$$\left(\frac{\partial^m \theta}{\partial y^m}\right)^+ = \lim_{\varepsilon \to 0+} \frac{\partial^m \theta (Y+\varepsilon, t)}{\partial y^m}, \tag{4.30}$$

$$\left(\frac{\partial^m \theta}{\partial y^m}\right)^- = \lim_{\varepsilon \to 0+} \frac{\partial^m \theta (Y-\varepsilon, t)}{\partial y^m}. \tag{4.31}$$

For the general case where either $D_m^{m\pm} \neq 1$ and/or at least one $D_k^{m\pm} \neq 0$ for $0 \leq k < m$, it is necessary to derive custom stencil coefficients for all stencils intersected by the interface. Examining the dendritic solidification equations, we see that equation (4.18) in one dimension,

$$h_S \frac{\partial \theta_S}{\partial y} - h_L \frac{\partial \theta_L}{\partial y} = \frac{dY}{dt} = V, \tag{4.32}$$

is a jump condition for the derivative of the temperature. Rewriting this in the form of (4.28), it becomes

$$\theta_y^+ = \frac{h_S}{h_L} \theta_y^- - \frac{V}{h_L}, \tag{4.33}$$

if the normal points out of the solid region (solid for $y \leq Y$). Comparing (4.33) and (4.28) we find

$$\left[\begin{array}{cc} D_0^{1-} & D_1^{1-} \end{array}\right] = \left[\begin{array}{cc} 0 & \frac{h_S}{h_L} \end{array}\right], \tag{4.34}$$

so, when the material properties in the solid and liquid phases are different, it is necessary to derive custom stencils near the interface.

From Chapter 3, we know that once the coefficients of the jump conditions in (4.28) are determined, we can derive the stencils required to differentiate solutions of the dendritic solidification problem accurately. The temperature is continuous at the interface, so

$$D_0^{0+} = D_0^{0-} = 1 \tag{4.35}$$

and

$$J_0^+ = J_0^- = 0. \tag{4.36}$$

From (4.33) we have

$$\begin{bmatrix} D_0^{1+} & D_1^{1+} \end{bmatrix} = \begin{bmatrix} 0 & \frac{h_L}{h_S} \end{bmatrix}, \tag{4.37}$$

$$J_1^+ = \frac{V}{h_S} \tag{4.38}$$

and

$$\begin{bmatrix} D_0^{1-} & D_1^{1-} \end{bmatrix} = \begin{bmatrix} 0 & \frac{h_S}{h_L} \end{bmatrix}, \tag{4.39}$$

$$J_1^- = -\frac{V}{h_L}. \tag{4.40}$$

The derivation of the jump conditions on the time derivative and higher order spatial derivatives is somewhat more involved, and is explained below.

In one dimension, the equations governing the temperature in the solid and liquid phases are

$$\frac{\partial \theta_S}{\partial t} = H_S \frac{\partial^2 \theta_S}{\partial y^2}, \tag{4.41}$$

and

$$\frac{\partial \theta_L}{\partial t} = H_L \frac{\partial^2 \theta_L}{\partial y^2}, \tag{4.42}$$

respectively. Evaluating (4.41) and (4.42) on the interface and subtracting them, we find

$$\frac{\partial \theta_L(Y,t)}{\partial t} - \frac{\partial \theta_S(Y,t)}{\partial t} = H_L \frac{\partial^2 \theta_L(Y,t)}{\partial y^2} - H_S \frac{\partial^2 \theta_S(Y,t)}{\partial y^2}, \tag{4.43}$$

or, recalling that the normal points away from the solid,

$$[\theta_t] = H_L \theta_{yy}^+ - H_S \theta_{yy}^-. \tag{4.44}$$

We do not know the value of the jump in the time derivative a priori, so this is not yet a proper jump condition (the values of J_2^+ and J_2^- must be known). We can calculate the jump in the time derivative, however, by using the continuity of the temperature field at the interface

$$\theta^{+}\left(Y\left(t\right),t\right)=\theta^{-}\left(Y\left(t\right),t\right). \tag{4.45}$$

Differentiating the above with respect to time, we find

$$\theta_t^{+}+V\theta_y^{+}=\theta_t^{-}+V\theta_y^{-}, \tag{4.46}$$

or

$$[\theta_t]=-V\left(\theta_y^{+}-\theta_y^{-}\right), \tag{4.47}$$

where V denotes the interface velocity

$$\frac{dY}{dt}=V. \tag{4.48}$$

Taken together, (4.33), (4.44) and (4.47) yields a system of three equations. This means that we can solve the system for $[\theta_t]$, θ_y^{+} and θ_{yy}^{+} in terms of θ_y^{-} and θ_{yy}^{-},

$$\theta_y^{+}=\frac{h_S}{h_L}\theta_y^{-}-\frac{V}{h_L}, \tag{4.49}$$

$$\theta_{yy}^{+}=\left(\frac{H_S}{H_L}\right)\theta_{yy}^{-}+\left[\left(\frac{h_S-h_L}{H_Lh_L}\right)V\right]\theta_y^{-}+\frac{V^2}{H_Lh_L}, \tag{4.50}$$

$$[\theta_t]=\left[\left(\frac{h_L-h_S}{h_L}\right)V\right]\theta_y^{-}+\frac{V^2}{h_L}, \tag{4.51}$$

or solve it for $[\theta_t]$, θ_y^{-} and θ_{yy}^{-} in terms of θ_y^{+} and θ_{yy}^{+},

$$\theta_y^{-}=\frac{h_L}{h_S}\theta_y^{+}+\frac{V}{h_S}, \tag{4.52}$$

$$\theta_{yy}^{-}=\left(\frac{H_L}{H_S}\right)\theta_{yy}^{+}+\left[\left(\frac{h_S-h_L}{H_Sh_S}\right)V\right]\theta_y^{+}-\frac{V^2}{H_Sh_S}, \tag{4.53}$$

$$[\theta_t]=\left[\left(\frac{h_L-h_S}{h_S}\right)V\right]\theta_y^{+}+\frac{V^2}{h_S}. \tag{4.54}$$

Note that the jump in the time derivative explicitly depends on the slope of the temperature at the interface. This is an unfortunate complication which we will address later. For now, we simply identify the values of the coefficients of the second derivatives in the jump conditions (4.28),

$$\begin{bmatrix} D_0^{2-} & D_1^{2-} & D_2^{2-} \end{bmatrix} = \begin{bmatrix} 0 & \left(\frac{h_L - h_S}{H_L h_L}\right) V & \frac{H_S}{H_L} \end{bmatrix}, \tag{4.55}$$

$$J_2^- = \frac{V^2}{H_L h_L} \tag{4.56}$$

and (4.29),

$$\begin{bmatrix} D_0^{2+} & D_1^{2+} & D_2^{2+} \end{bmatrix} = \begin{bmatrix} 0 & \left(\frac{h_S - h_L}{H_S h_S}\right) V & \frac{H_L}{H_S} \end{bmatrix}, \tag{4.57}$$

$$J_2^+ = -\frac{V^2}{H_S h_S}. \tag{4.58}$$

As we discussed in Chapter 3, knowledge of the jump conditions through the second derivative is sufficient to determine a first order accurate approximation for the Laplacian at nodes with stencils intersected by the interface. Since all the stencils for the Laplacian away from the interface provide second order accurate approximations, the larger discretization error at the interface stencils is insufficient to reduce the overall accuracy of the numerical solution. While we have enough information at this point to develop a scheme that is second order accurate in space, it is instructive to demonstrate how the jumps for the next set of derivatives can be calculated.

To compute the jumps in the third derivatives, we will need the y-derivative of the equation evaluated on each side of the interface:

$$\theta_{ty}^- = H_S \theta_{yyy}^- \tag{4.59}$$

and

$$\theta_{ty}^+ = H_L \theta_{yyy}^+. \tag{4.60}$$

We will also need the time derivative of (4.33),

$$h_S \left(\theta_{ty}^- + V^2 \theta_{yy}^-\right) - h_L \left(\theta_{ty}^+ + V^2 \theta_{yy}^+\right) = A \tag{4.61}$$

and (4.47),

$$[\theta_{tt}] = -A\left(\theta_y^+ - \theta_y^-\right) - 2V\left(\theta_{ty}^+ - \theta_{ty}^-\right) - V^2\left(\theta_{yy}^+ - \theta_{yy}^-\right), \tag{4.62}$$

where A denotes the acceleration of the interface

$$\frac{d^2Y}{dt^2} = A. \tag{4.63}$$

Using equations (4.33)-(4.47) and (4.59)-(4.62) allows us to determine, among other things, the jump conditions for the third derivatives

$$\begin{aligned}\theta_{yyy}^- &= \left(\frac{h_L H_L}{h_S H_S}\right)\theta_{yyy}^+ + \left[\left(\frac{h_L H_S - h_S H_L}{h_S H_S^2}\right)V\right]\theta_{yy}^+ + \\ &\quad \left[\left(\frac{h_L - h_S}{h_S H_S^2}\right)V^2\right]\theta_y^+ + \frac{1}{h_S H_S}\left[A + \frac{V^3}{H_S}\right],\end{aligned} \tag{4.64}$$

$$\begin{aligned}\theta_{yyy}^+ &= \left(\frac{h_S H_S}{h_L H_L}\right)\theta_{yyy}^- + \left[\left(\frac{h_S H_L - h_L H_S}{h_L H_L^2}\right)V\right]\theta_{yy}^- + \\ &\quad \left[\left(\frac{h_S - h_L}{h_L H_L^2}\right)V^2\right]\theta_y^- - \left(\frac{A H_L + V^3}{h_L (H_L)^2}\right),\end{aligned} \tag{4.65}$$

and expressions for the jump in the second derivative with respect to time

$$\begin{aligned}[\theta_{tt}] &= \left[\left(\frac{2H_L(h_L - h_S)}{h_S}\right)V\right]\theta_{yyy}^+ - \\ &\quad \left[\left(\frac{H_L h_S - 2H_S h_L + H_S h_S}{H_S h_S}\right)V^2\right]\theta_{yy}^+ + \\ &\quad \left[\left(\frac{h_L - h_S}{H_S h_S}\right)\left(H_S A + V^3\right)\right]\theta_y^+ + \\ &\quad \left(\frac{AV^3 + 3H_S}{H_S h_S}\right)V,\end{aligned} \tag{4.66}$$

$$[\theta_{tt}] = \left[\frac{2VH_S(h_L - h_S)}{h_L}\right]\theta_{yyy}^- + \tag{4.67}$$

$$\left[\left(\frac{H_S h_L - 2H_L h_S + H_L h_L}{H_L h_L}\right) V^2\right] \theta_{yy}^- +$$
$$\left[\left(\frac{h_L - h_S}{H_L h_L}\right) \left(H_L A + V^3\right)\right] \theta_y^- +$$
$$\left(\frac{V^4 + 3H_L A V}{H_L h_L}\right).$$

As we shall see shortly, the dependence of the jump in the second derivatives with respect to time on the third order spatial derivatives implies that it is necessary to know all of the above quantities to develop a scheme that is second order accurate in time. To that end, we now examine the correction of the discrete time derivatives near the interface.

4.4.2 Discretization of the time derivative

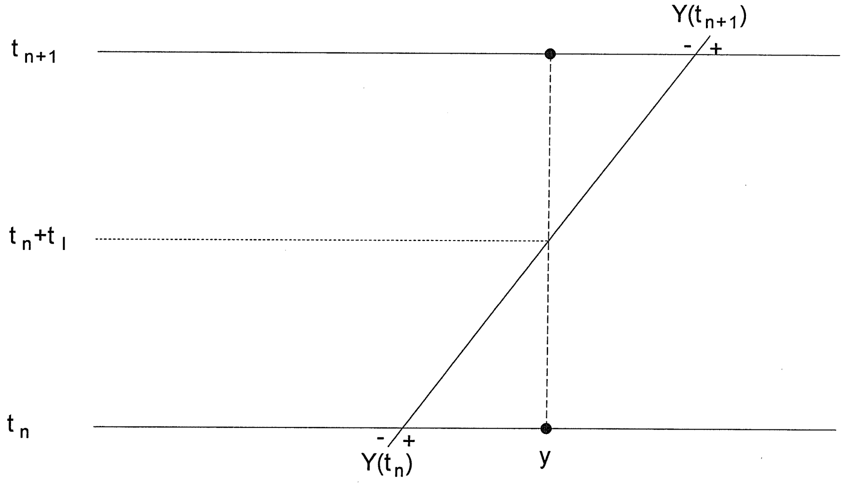

Figure 4.2: Time jump definitions

We will utilize a one-step method of the form

$$\theta^{n+1}(i) = \theta^n(i) + \Delta t \left\{ \alpha H^{n+1}(i)\, \theta_{yy}^{n+1}(i) + (1-\alpha) H^n(i)\, \theta_{yy}^n(i) \right\} + E(i), \quad (4.68)$$

to advance the equation forward in time where

$$H^n(i) = \begin{cases} H_S & \text{if } y(i) \le Y^n \\ H_L & \text{if } y(i) > Y^n \end{cases} ; \tag{4.69}$$

the value of $E(i)$ is zero unless the node changes phase during the time step and $\alpha \in [0,1]$. When the node does change phase during the time step, it is necessary to set $E(i)$ to a non-zero value to account for the jump in the time derivative across the interface. Using the correction derived in Chapter 3, we find

$$E(i) = \begin{cases} 0 & \text{if no phase change} \\ +[\Delta t(1-\alpha) - t_I][\theta_t] + \frac{1}{2}t_I(t_I - \Delta t)[\theta_{tt}] & \text{if solid at } t_n \\ -[\Delta t(1-\alpha) - t_I][\theta_t] - \frac{1}{2}t_I(t_I - \Delta t)[\theta_{tt}] & \text{if liquid at } t_n \end{cases} , \tag{4.70}$$

where the time the interface intersects the node, $t_n + t_I$, is determined from

$$\frac{(\Delta t - t_I)Y^n + t_I Y^{n+1}}{\Delta t} = y(i) . \tag{4.71}$$

We found above that $[\theta_t]$ and $[\theta_{tt}]$ can only be calculated in terms of the spatial derivatives evaluated at the interface. Thus, for the non-symmetric case, the time correction is not simply an additive correction. Instead, we will need to modify the spatial stencil discretizing the Laplacian so that the new stencil also calculates the value of $E(i)$. Suppose that the spatial node located at $y(i)$ is crossed by the interface during the current time step $(Y(t_n) \le y(i) < Y(t_{n+1}))$, as shown in Fig. 4.2. The node begins the time step as liquid (i.e., on the "+" side of the interface) and completes the time step as solid (i.e., on the "-" side of the interface). We can, therefore, compute the values of the derivatives at the interface at the beginning

$$\theta_y^+ = \theta_y^n(i) + (Y^n - y(i))\,\theta_{yy}^n(i) + O\left(\Delta y^2\right), \tag{4.72}$$

$$\theta_{yy}^+ = \theta_{yy}^n(i) + (Y^n - y(i))\,\theta_{yyy}^n(i) + O\left(\Delta y^2\right), \tag{4.73}$$

$$\theta^{+}_{yyy} = \theta^{n}_{yyy}(i) + O(\Delta y), \tag{4.74}$$

and end

$$\theta^{-}_{y} = \theta^{n+1}_{y}(i) + \left(Y^{n+1} - y(i)\right)\theta^{n+1}_{yy}(i) + O\left(\Delta y^{2}\right), \tag{4.75}$$

$$\theta^{-}_{yy} = \theta^{n+1}_{yy}(i) + \left(Y^{n+1} - y(i)\right)\theta^{n+1}_{yyy}(i) + O\left(\Delta y^{2}\right), \tag{4.76}$$

$$\theta^{-}_{yyy} = \theta^{n+1}_{yyy}(i) + O(\Delta y), \tag{4.77}$$

of the time step. Using (4.51), (4.54), (4.66) and (4.67) the values of the jumps in the time derivative at time t_n and time t_{n+1} can be estimated. Then, using interpolation we can calculate the jump in the time derivatives,

$$[\theta_t] = \frac{1}{\Delta t}\left((\Delta t - t_I)[\theta_t]^{n} + t_I[\theta_t]^{n+1}\right), \tag{4.78}$$

$$[\theta_{tt}] = \frac{1}{\Delta t}\left((\Delta t - t_I)[\theta_{tt}]^{n} + t_I[\theta_{tt}]^{n+1}\right), \tag{4.79}$$

at the time of the interface intersection. The spatial derivatives required to calculate (4.78) and (4.79) and, hence $E(i)$, are incorporated into the system along with the discretization for the Laplacian. It is important to remember, however, that $E(i) \neq 0$ only for the time steps during which the computational node is intersected by the interface. The above information is enough to construct a fully second order accurate scheme in space and time. We have not yet addressed the issue of determining the interface velocity, however, so we do this now.

A quick glance at the above expressions for the jump conditions on the temperature derivatives reveals that they all depend on the interface velocity. In order to calculate stencils for the temperature near the interface, we must know its jump conditions and, therefore, the interface velocity. If the interface position, $Y(t)$, is given, then the above jump conditions and the formulas for modified stencils in Chapter 3 allows us to discretize and solve (4.68). It is important to note that, because of its use in the derivation of the jump conditions and incorporation into the irregular interface stencils, the condition

$$h_S \frac{\partial \theta_S}{\partial y} - h_L \frac{\partial \theta_L}{\partial y} = V, \tag{4.80}$$

will be automatically satisfied by all numerical solutions of (4.68). The only equation which may not be satisfied by the above is the Gibbs-Thomson condition, which in one spatial dimension is

$$\theta\left(Y\left(t\right), t\right) = 0. \tag{4.81}$$

For any given interface motion, $Y\left(t\right)$, (4.68) and (4.80) will be satisfied by our numerical solution. Thus, for our numerical method, the interface motion is determined by satisfying (4.81). In order to satisfy (4.81), it is necessary to determine the value of the temperature at the interface. Unfortunately, the interface is not, in general, located at a computational node. Thus, we need an interpolation scheme that is accurate for non-smooth functions such as the temperature.

4.4.3 Interpolation of the interface temperature

There are several approaches that we could use to interpolate the value of the temperature on the interface. Perhaps the most obvious would be to use a one-sided stencil (all nodes on one side of the interface or the other) to extrapolate the interface temperature. While this approach is feasible and can even be generalized to higher dimensions, it has a subtle difficulty that we wish to avoid. To illustrate the potential problem, consider three adjacent nodes located at $y\left(i\right) < y\left(i+1\right) < y\left(i+2\right)$ where, at the beginning of the time step, the interface satisfies $y\left(i+1\right) \leq Y\left(t\right) < y\left(i+2\right)$. The value of the interface temperature at the beginning of the time step

$$\Theta = \theta\left(Y\left(t\right), t\right), \tag{4.82}$$

can be approximated using

$$\Theta = \theta\left(i\right)\left(\frac{Y - y\left(i+1\right)}{y\left(i\right) - y\left(i+1\right)}\right) + \theta\left(i+1\right)\left(\frac{Y - y\left(i\right)}{y\left(i+1\right) - y\left(i\right)}\right) + O\left(\Delta y^2\right). \tag{4.83}$$

Depending on our guess for the new interface position, the above representation may or may not be valid at the end of the time step. If, at the end of the time step, the interface still satisfies $y(i+1) \leq Y(t+\Delta t) < y(i+2)$, then the above approximation for the interface temperature holds. On the other hand, if the interface moves far enough that $y(i+2) \leq Y(t+\Delta t)$ is satisfied, we must instead use

$$\Theta = \theta(i+1)\left(\frac{Y - y(i+2)}{y(i+1) - y(i+2)}\right) + \theta(i+2)\left(\frac{Y - y(i+1)}{y(i+2) - y(i+1)}\right) + O\left(\Delta y^2\right). \tag{4.84}$$

Thus, depending on our guess for the interface position, the numerical domain of dependence of our interpolation formula can change discontinuously. Note that the coefficient of $\theta(i)$ just before Y crosses $y(i+2)$ is one but discontinuously jumps to zero just after Y crosses $y(i+2)$. A jump in the discretization error is also associated with this change. Discontinuous jumps in the error associated with minute changes in Y introduce numerical perturbations into the system and can cause difficulties with the convergence of iterative schemes. We have developed a simple interpolation scheme that does not suffer from these problems.

Consider the problem of determining the interface temperature, Θ, where the interface lies between two computational nodes, $y(i+1) \leq Y(t) < y(i+2)$. Since we have developed a method for accurately calculating derivatives at nodes near the interface, one thing we could do is estimate the interface temperature using a Taylor series expansion about the computational nodes,

$$\Theta_1 = \theta(i+1) + (Y - y(i+1))\,\theta_y(i+1) + \frac{1}{2}(Y - y(i+1))^2\,\theta_{yy}(i+1) + O\left(\Delta y^2\right) \tag{4.85}$$

and

$$\Theta_2 = \theta(i+2) + (Y - y(i+2))\,\theta_y(i+2) + \frac{1}{2}(Y - y(i+2))^2\,\theta_{yy}(i+2) + O\left(\Delta y^2\right). \tag{4.86}$$

Note that the accuracy of the above is limited to second order despite the order of the expansion because that is the accuracy to which the nodal values are known. Neither of the above are one-sided approximations because the stencils used to estimate the derivatives utilize nodes on both sides of the interface. If we only choose one of the above, however, we would still have an estimate for the interface temperature that jumps discontinuously as the interface moved. That is, if we estimate the interface temperature using a Taylor series expansion about a single node, eventually the location of the expansion node will have to be moved and the numerical domain of dependence and discretization error will change with it. Instead of using one node, we use the estimates for interface temperature at both nodes that bracket the interface and average the results

$$\Theta = \left(\frac{Y - y\,(i+2)}{y\,(i+1) - y\,(i+2)}\right)\Theta_1 + \left(\frac{Y - y\,(i+1)}{y\,(i+2) - y\,(i+1)}\right)\Theta_2 + O\left(\Delta y^2\right). \qquad (4.87)$$

This yields an interpolation formula that has a continuously varying numerical domain of dependence because the expansion points bracket the interface. For instance, the coefficient of Θ_1 just before Y crosses $y\,(i+2)$ is zero and will also be zero after Y crosses $y\,(i+2)$ since the interface temperature is then approximated using

$$\Theta = \left(\frac{Y - y\,(i+3)}{y\,(i+2) - y\,(i+3)}\right)\Theta_2 + \left(\frac{Y - y\,(i+2)}{y\,(i+3) - y\,(i+2)}\right)\Theta_3 + O\left(\Delta y^2\right). \qquad (4.88)$$

We will demonstrate shortly that this approach can be readily extended to higher dimensions as well. Now that an interpolation scheme for the interface temperature has been developed, we are ready to describe the complete algorithm for the solution of one-dimensional dendritic solidification problems.

4.4.4 Complete scheme

For our discussion below, we assume that the values of the interface position, Y^n, and interface velocity, V^n, are known at the beginning of the time step. In general, these must be supplied as initial conditions to start the algorithm. We estimate the interface position and derivatives at the end of the time step using

$$Y^{n+1} = Y^n + \frac{\Delta t}{2}\left(V^{n+1} + V^n\right), \tag{4.89}$$

$$A^n = \left(\frac{V^{n+1} - V^n}{\Delta t}\right) \tag{4.90}$$

where V^{n+1} is the current guess for the interface velocity. The discrete system for θ^{n+1} and the interpolation formulas then act to implicitly define a non-linear system for V^{n+1}

$$\theta\left(Y\left(t_{n+1}\right); V^{n+1}\right) = 0. \tag{4.91}$$

The algorithm is as follows:

1. Using the current guess for the new interface velocity V^{n+1}, determine Y^{n+1} and A^n using (4.89)-(4.90).

2. Using the values from step 1, calculate the new temperature using corrected stencils near the interface.

3. Calculate the velocity error, $e = \theta\left(Y^{n+1}; V^{n+1}\right)$, using (4.87).

4. If error is too large, $|e| > tol$, update the velocity guess using

$$V^{n+1} := V^{n+1} - e/\omega \tag{4.92}$$

   and go back to step 1.

The value of ω is an estimate to the system Jacobian. Its value is picked by hand during the iterations for the first time step and held at that value for the remainder of the simulation. While this is a crude iterative scheme we have found it to be both robust and efficient for the problems we present below. The value of *tol* in the above algorithm is taken to be 10^{-8}. The solution is fairly insensitive to *tol* with very little difference observed if *tol* is taken to be substantially larger ($tol = 10^{-5}$, for example). Now we use the above algorithm to simulate two simple one-dimensional dendritic solidification problems.

4.4.5 Results

First we compute a solution to the traveling wave problem discussed in Chapter 2. In this case, the temperature in the solid is given by

$$\theta_S = 0 \tag{4.93}$$

and the temperature in the liquid is given by

$$\theta_L = \frac{H_L}{h_L}\left\{\exp\left[\frac{-V\,(y-Vt)}{H_L}\right] - 1\right\}. \tag{4.94}$$

Recall that the interface velocity for the traveling wave problem is not unique. We wish to compute the solution corresponding to

$$V = \frac{1}{2}. \tag{4.95}$$

The material properties used in our simulation are set to

$$H_S = h_S = 1 \tag{4.96}$$

and

$$H_L = h_L = 2. \tag{4.97}$$

Note that although the material properties in the solid do not influence the analytic solution, they do influence the numerical method so this problem is a useful test case.

We numerically compute the solution of the traveling wave problem starting the simulation at $t = 0$ using (4.93) and (4.94) and ending the simulation at $t = 10$. The numerical solution is computed using the immersed boundary method (referred to as IBM from now on, see the discussion above) and our immersed interface method outlined above. The immersed interface solutions are carried out using the fully second order accurate in space, second order accurate in time method ($\alpha = 1/2$, referred to as IIM2 from now on) which corrects all spatial derivatives through third order and all time derivatives through second order. We also compute immersed interface solutions using the second order accurate in space and first order accurate in time scheme ($\alpha = 1$, referred to as IIM1 from now on) which corrects all spatial derivatives through second order and the first derivative with respect to time. The accuracy with respect to time is different for the individual methods. The time accuracy of IBM and IIM1 is only first order while the accuracy of IIM2 is second order. To facilitate fair comparisons between the different methods, we perform all calculations using $\Delta t = 0.2\,(\Delta x)^2$, so that the error associated with time stepping decreases by a factor of four as the mesh spacing, $\Delta x = (x_R - x_L)/N$ with $x_L = -2$ and $x_R = 14$, is halved. The infinity norm of the temperature error at $t = 10$ for several different mesh spacing is found in Tables (4.1), (4.2) and (4.3) for IBM, IIM1 and IIM2, respectively. The first order accuracy of IBM and the second order accuracy of IIM1 and IIM2 is clear from these results. A plot of the solutions for $N = 64$ is shown in Fig. 4.3. Note that there is no observable difference between IIM1, IIM2 and the exact solution on the plot so only IIM2 is displayed.

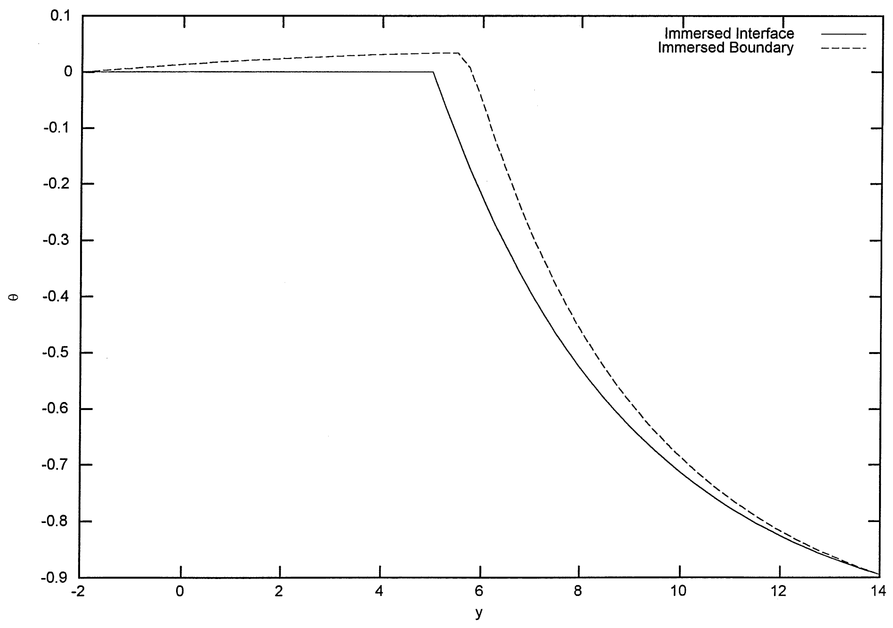

Figure 4.3: Comparison of the immersed interface and immersed boundary solutions at $t = 10$ for the traveling wave problem

N	$\|\theta_N - \theta\|_\infty$	Ratio
32	0.37362	
64	0.17677	2.1136
128	0.08727	2.0262
256	0.04334	2.0137

Table 4.1: Immersed boundary resolution study of the traveling wave problem

N	$\|\theta_N - \theta\|_\infty$	Ratio
32	4.4822x10^{-3}	
64	9.4943x10^{-4}	4.7209
128	2.7586x10^{-4}	3.4417
256	6.5895x10^{-5}	4.1863

Table 4.2: Resolution study for the immersed interface method with first order time accuracy for the traveling wave problem

N	$\|\theta_N - \theta\|_\infty$	Ratio
32	2.2323x10^{-3}	
64	4.5673x10^{-4}	4.8875
128	1.2673x10^{-4}	3.6038
256	3.0018x10^{-5}	4.2220

Table 4.3: Resolution study for the immersed interface method with second order time accuracy for the traveling wave problem

The above results yield a good picture of the quality of the solution at a given instant in time ($t = 10$ in this case). It is also useful to record the time history of the temperature at a specific point ($y = 3$, in our case) which yields an accurate representation of temperature before, during, and after an interface crossing. The infinity norm of the temperature error at $y = 3$ for several different mesh spacings is found in Tables (4.4), (4.5) and (4.6) for IBM, IIM1 and IIM2, respectively. The first order accuracy of IBM and the second order accuracy of IIM1 and IIM2 is clear from these results. A plot of the solutions at $y = 3$ for $N = 64$ is shown in Fig. 4.4. Note that there is no observable difference between IIM1, IIM2 and the exact solution on the plot, so only IIM2 is displayed.

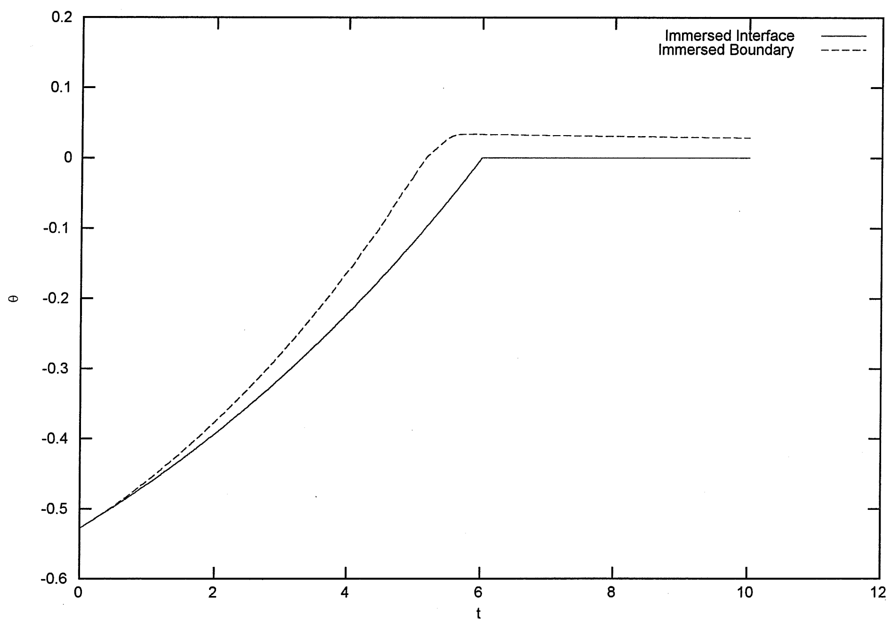

Figure 4.4: Comparison of the immersed interface and immersed boundary solutions at $y = 3$ for the traveling wave problem

Finally, we have plotted the numerical interface velocity for the different schemes using $N = 64$ in Fig. 4.5. Note the oscillatory behavior of the interface velocity. The sharp cusps in the oscillations correspond to points in time where the interface crosses a computational node. Continuing the trend found in the other results, the errors associated with the velocity obtained using the immersed interface methods

N	$\|\theta_N - \theta\|_\infty$	Ratio
32	0.18598	
64	0.09893	1.8798
128	0.05140	1.9247
256	0.02629	1.9550

Table 4.4: Immersed boundary error at $y = 3$ for the traveling wave problem

N	$\|\theta_N - \theta\|_\infty$	Ratio
32	$1.8095\text{x}10^{-3}$	
64	$4.4162\text{x}10^{-4}$	4.0974
128	$1.0873\text{x}10^{-4}$	4.0617
256	$2.6958\text{x}10^{-5}$	4.0332

Table 4.5: Immersed interface error at $y = 3$ with first order accurate time stepping for the traveling wave problem

N	$\|\theta_N - \theta\|_\infty$	Ratio
32	$9.8977\text{x}10^{-4}$	
64	$2.2935\text{x}10^{-4}$	4.3155
128	$5.5008\text{x}10^{-5}$	4.1694
256	$1.3460\text{x}10^{-5}$	4.0879

Table 4.6: Immersed interface error at $y = 3$ with second order accurate time stepping for the traveling wave problem

are significantly smaller than the error associated with the IBM velocity. In Fig. 4.6 we have plotted the interface velocity obtained just by the immersed interface methods. Clearly the interface velocity obtained by IIM2 (the fully second order accurate scheme) is more accurate than the interface velocity calculated using IIM1. The improvement in the velocity error for IIM2 comes from two sources. The reduction in the amplitude of the oscillations is due to the improved spatial resolution. That is, the calculation of the jumps in all the spatial derivatives out to third order improves the accuracy of the interpolation of the interface temperature. More accurate interpolation leads to a reduction in the amplitude of the velocity oscillations. The improvement of the overall error in the IIM2 velocity is due to the use of a second order accurate time stepping scheme.

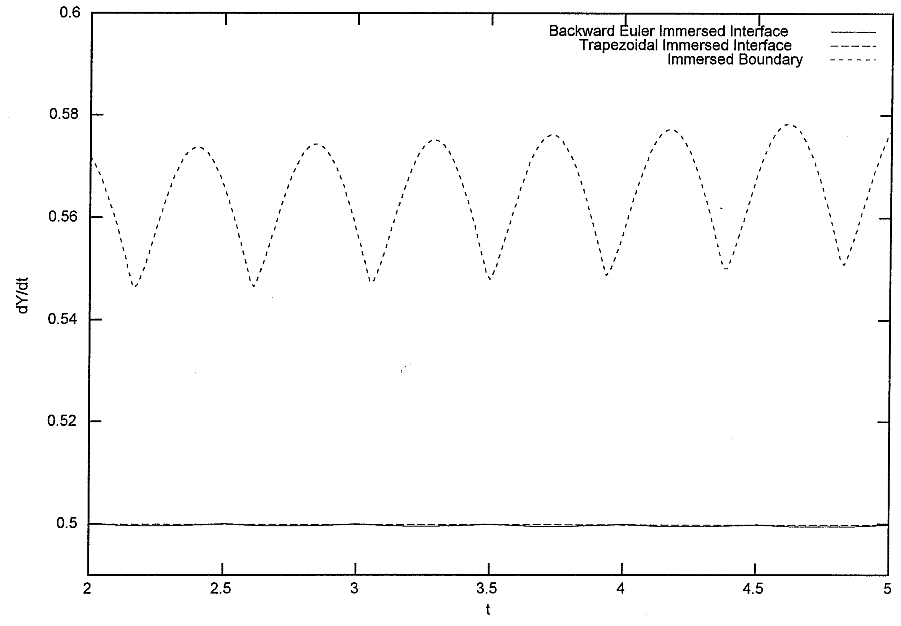

Figure 4.5: Comparison of the immersed interface and immersed boundary interface velocity for the traveling wave problem

Next we consider the sub-critically cooled dendritic solidification problem associated with a solid located at $y \leq 0$ initially at a uniform temperature $\theta_S = \theta_{-\infty} \geq 0$ and a liquid located at $y > 0$ initially at a uniform temperature $\theta_L = \theta_{+\infty} \leq 0$. As we saw in Chapter 2, this problem has an exact solution. The interface position is

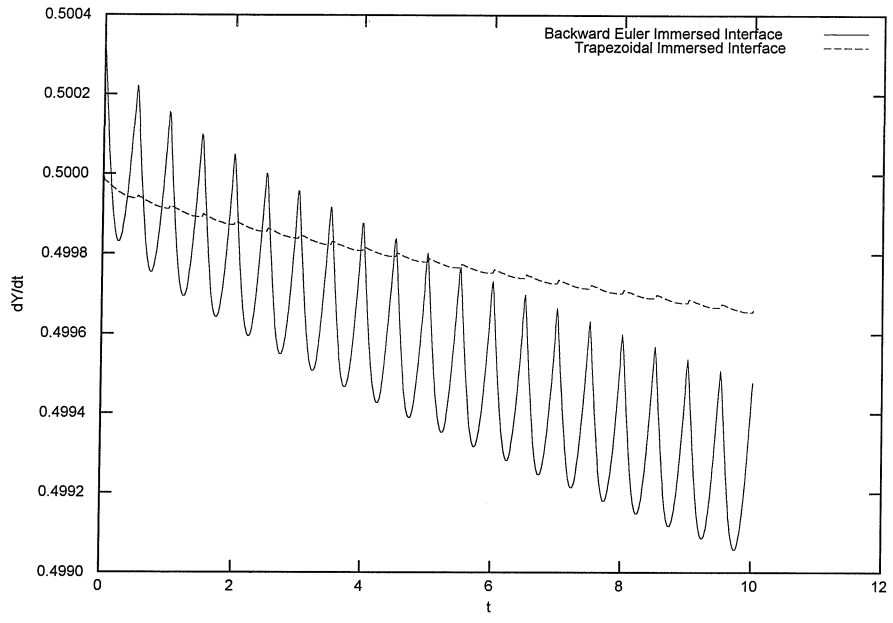

Figure 4.6: Comparison of the interface velocity computed using different time stepping methods for the traveling wave problem

given by

$$Y(t) = 2a\sqrt{t}, \tag{4.98}$$

the temperature in the solid ($y \leq Y(t)$) is given by

$$\theta_S = \theta_{-\infty} - \theta_{-\infty} \frac{\left[1 + \operatorname{erf}\left(\frac{y}{2\sqrt{H_S t}}\right)\right]}{\left[1 + \operatorname{erf}\left(\frac{a}{\sqrt{H_S}}\right)\right]}, \tag{4.99}$$

and the temperature in the liquid ($y > Y(t)$) is given by

$$\theta_L = \theta_{+\infty} - \theta_{+\infty} \frac{\left[1 - \operatorname{erf}\left(\frac{y}{2\sqrt{H_L t}}\right)\right]}{\left[1 - \operatorname{erf}\left(\frac{a}{\sqrt{H_L}}\right)\right]}. \tag{4.100}$$

Using the same material properties as before,

$$H_S = h_S = 1, \tag{4.101}$$

$$H_L = h_L = 2, \tag{4.102}$$

and an initial temperature distribution in the solid and liquid is given by

$$\theta_{-\infty} = 0.2, \tag{4.103}$$

and

$$\theta_{+\infty} = -0.6, \tag{4.104}$$

respectively. We can solve for a as described in Chapter 2 to find

$$a = 0.77070929661959257638... \tag{4.105}$$

We numerically compute the solution of this step problem (the initial condition is a step function) starting the simulation at $t = 0.1$ using (4.99) and (4.100) and end the simulation at $t = 10$. The numerical solution is computed using IBM (the immersed boundary method) and IIM2 (the second order accurate in space and time immersed interface method). We again perform all calculations using $\Delta t = 0.2\,(\Delta x)^2$, so that the error associated with time stepping decreases by a factor of four as the mesh spacing, $\Delta x = (x_R - x_L)/N$ with $x_L = -7.5$ and $x_R = 12.5$, is halved. The infinity norm of the temperature error at $t = 10$ for several different mesh spacings is found in Tables (4.7) and (4.8) for IBM and IIM2, respectively. The first order accuracy of IBM and the second order accuracy of IIM2 is clear from these results. A plot of the solutions for $N = 64$ is shown in Fig. 4.7. Note that there is no observable difference between IIM2 and the exact solution on the plot so only IIM2 is displayed.

The infinity norm of the temperature error at $y = 3$ for several different mesh spacing is found in Tables (4.9) and (4.10) for IBM and IIM2, respectively. The first order accuracy of IBM and the second order accuracy of IIM2 is clear from these

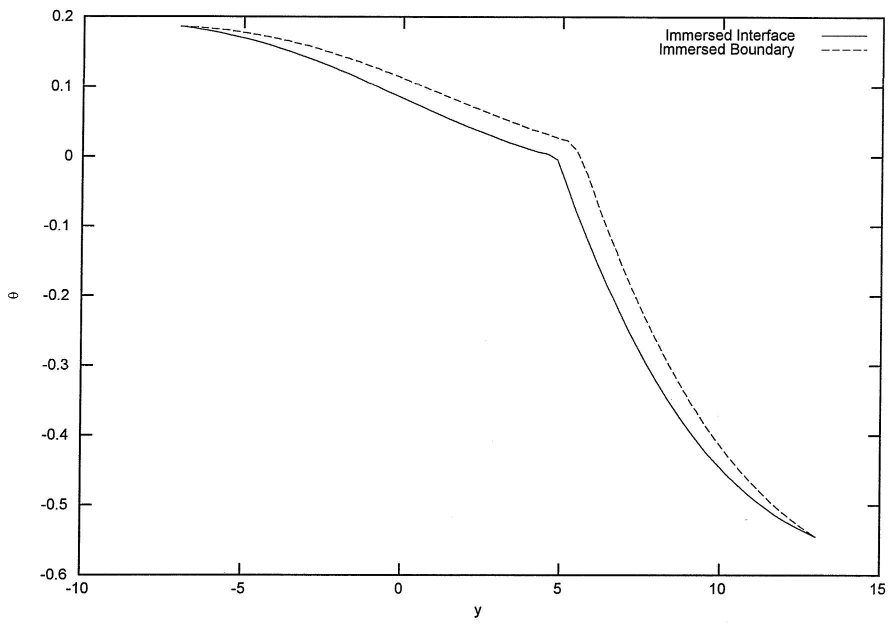

Figure 4.7: Comparison of the immersed interface and immersed boundary solutions at $t = 10$ for the step problem

N	$\|\|\theta_N - \theta\|\|_\infty$	Ratio
32	0.15596	
64	0.08845	1.7633
128	0.04448	1.9883
256	0.02203	2.0190

Table 4.7: Immersed boundary resolution study of the step problem

N	$\|\|\theta_N - \theta\|\|_\infty$	Ratio
32	$2.4469\text{x}10^{-2}$	
64	$4.6130\text{x}10^{-3}$	5.3044
128	$1.2177\text{x}10^{-3}$	3.7884
256	$3.1552\text{x}10^{-4}$	3.8593

Table 4.8: Immersed interface with second order accurate time stepping for the step problem

N	$\|\theta_N - \theta\|_\infty$	Ratio
32	0.26892	
64	0.14466	1.8595
128	0.06716	2.1538
256	0.03192	2.1041

Table 4.9: Immersed boundary error at $y = 3$ for the step problem

results. A plot of the solutions at $y = 3$ for $N = 64$ is shown in Fig. 4.8. Note that there is no observable difference between IIM2 and the exact solution on the plot so only IIM2 is displayed.

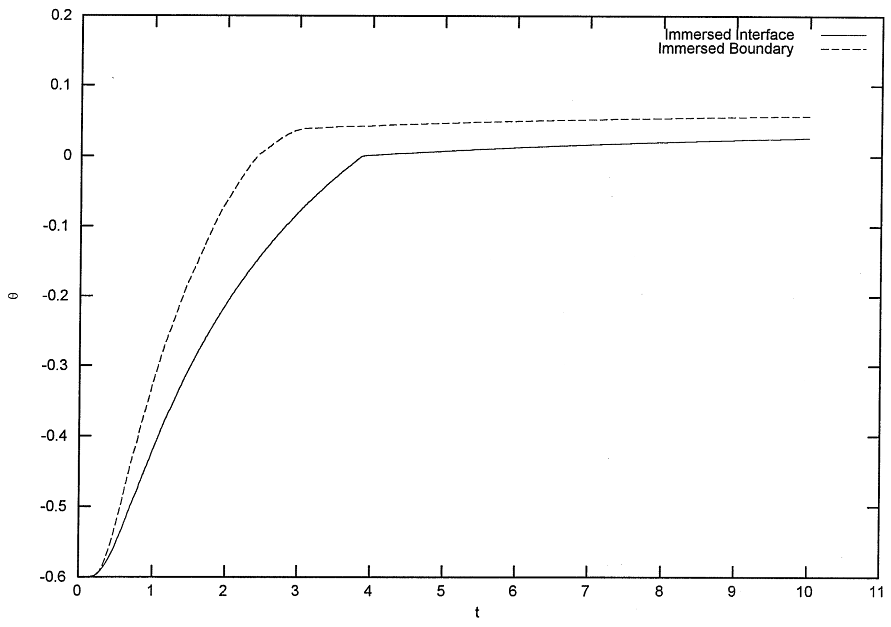

Figure 4.8: Comparison of the immersed interface and immersed boundary solutions at $y = 3$ for the step problem

Finally, we have plotted the numerical interface velocity for the two different schemes using $N = 64$ in Fig. 4.9. The initially very large error in the IBM velocity is due to the poor accuracy of its interface temperature interpolation method. Because the interpolated temperature obtained by the IBM is so inaccurate, the initial condition appears to significantly violate the requirement that the interface temperature be zero. This leads to large initial errors as the method artificially releases latent

N	$\|\|\theta_N - \theta\|\|_\infty$	Ratio
32	3.4214x10^{-2}	
64	6.1143x10^{-3}	5.5657
128	1.6239x10^{-3}	3.7652
256	4.2217x10^{-4}	3.8466

Table 4.10: Immersed interface error at $y = 3$ with second order accurate time stepping for the step problem

heat until the temperature is smoothed out and satisfies the zero interface temperature condition according to the IBM interpolation scheme. The interface velocity generated by IIM2, on the other hand, is so close to the exact value that we have not even bothered to plot the exact interface velocity in Fig. 4.9. We have performed a resolution study of the interface velocity error for IIM2 in Fig. 4.10. The second order accurate convergence of the interface velocity for IIM2 is clear.

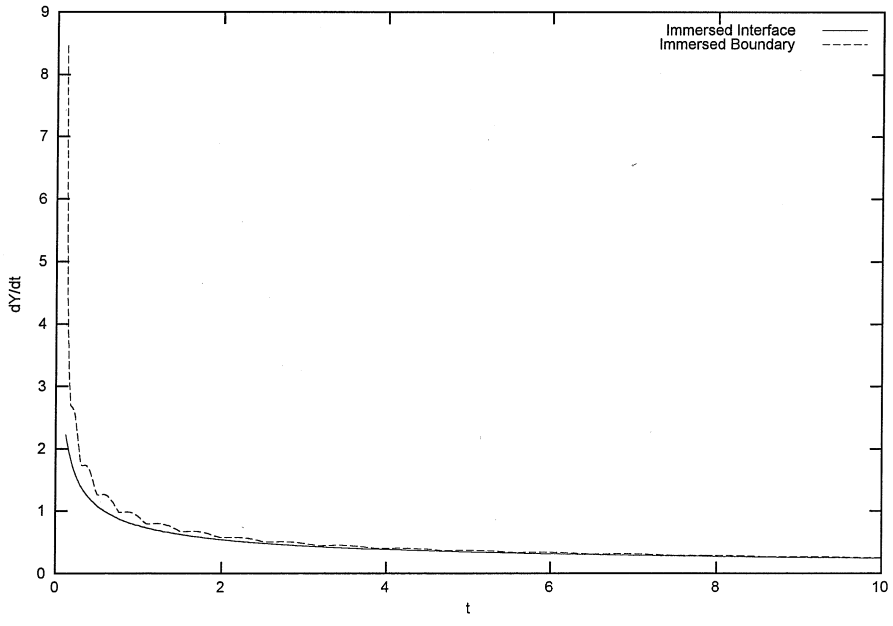

Figure 4.9: Comparison of the immersed interface and immersed boundary interface velocity for the step problem

We have now demonstrated the accuracy of the immersed interface method for one-dimensional dendritic solidification problems. Most of the phenomena we observed are generic and will also be present in our two-dimensional simulations.

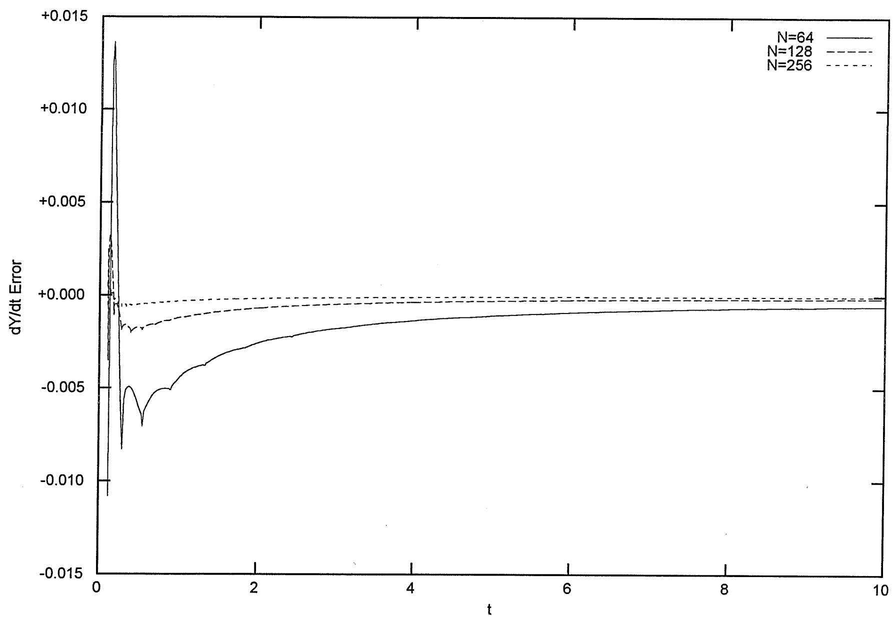

Figure 4.10: Resolution study of the interface velocity error for the immersed interface method

4.5 Two-dimensional problems

The addition of a spatial dimension to our simulations increases the complexity and computational cost of the problem significantly. In one dimension the interface can be represented by a single marker particle that translates along the y-axis. In two dimensions the interface becomes a curve which can not only translate, but deform as well. In one dimension, an algorithm to identify the regions occupied by liquid and solid is trivial to implement. Such an algorithm is much harder to develop in two dimensions. Finally, in one dimension the discrete systems that must be solved have a very small band width and can always be inverted efficiently. The band width of the discrete system in two dimensions is significantly larger. As we shall see later, efficiency concerns will force us to restrict our attention to symmetric problems in two dimensions. While many physical materials are not even remotely symmetric, succinonitrile is widely used in experiments and can be modeled quite accurately as a symmetric substance. So, while the symmetry assumption is restrictive, it is not completely unphysical.

4.5.1 Representation of the interface

The problem we are solving is periodic in the horizontal direction with a periodic interface dividing the domain into two parts, as shown in Fig. 4.1. The interface is explicitly tracked through the computational domain by a discrete set of marker particles. The interface is then represented parametrically, $(X(q,t), Y(q,t))$ where $q \in [0,1]$, using a periodic cubic spline that passes through the marker particles. The determination of the interpolant requires the solution of a periodic tri-diagonal system which can be solved efficiently using Keller's bordering algorithm (see [31]). The interface is evolved forward in time by moving each marker particle at the local normal velocity of the interface. The numerical evolution equations we employ depend on the order of the time accuracy desired. The evolution can be done in either a first or second order manner. In either case, a prediction is calculated for the new position of the marker particles using V_N^{n+1}, the normal velocity of the interface at time t_{n+1},

$$X^* = X^n + \Delta t \left(n_x^n V_N^{n+1} \right), \tag{4.106}$$

$$Y^* = Y^n + \Delta t \left(n_y^n V_N^{n+1} \right). \tag{4.107}$$

For a first order accurate in time scheme, this prediction is sufficient,

$$X^{n+1} = X^*, \tag{4.108}$$

$$Y^{n+1} = Y^*. \tag{4.109}$$

To obtain second order accuracy, we use the predicted interface location to calculate a trapezoidal rule update for the marker particle locations,

$$X^{n+1} = X^n + \frac{\Delta t}{2} \left(n_x^* V_N^{n+1} + n_x^n V_N^n \right), \tag{4.110}$$

$$Y^{n+1} = Y^n + \frac{\Delta t}{2} \left(n_y^* V_N^{n+1} + n_y^n V_N^n \right), \tag{4.111}$$

where $\left(n_x^*, n_y^*\right)$ is the normal calculated from a periodic cubic spline interpolant passing through (X^*, Y^*). The determination of the normal velocity is somewhat involved and will be discussed later.

4.5.2 Determination of the phase of each node

We are restricting our attention to symmetric problems so it is not necessary to know if a point is liquid or solid to determine the local material properties. It is still important, however, to determine the phase of each computational node. When the effects of convection are included, we will need to know which nodes require the additional convection terms. For purely diffusive problems, we need to know the phase of the nodes that are adjacent to the interface to correct the spatial derivatives. We will also need to determine if any nodes have changed phase during a time step and, if they have, at exactly what time the phase change occurred.

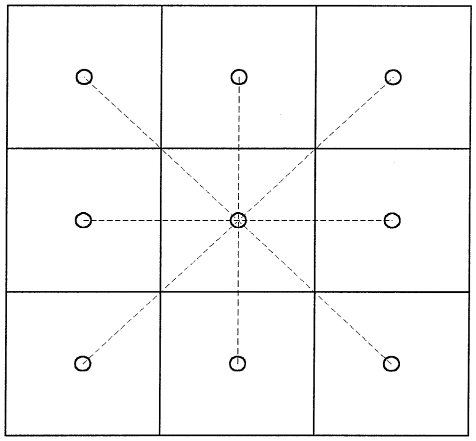

Figure 4.11: Legs associated with a nine point stencil

The phase of the computational nodes can be determined from the geometry of the interface. The first step is to determine all of the stencil legs that are intersected by the interface. The stencil legs associated with each computational node are shown in Fig. 4.11. The exact location and parameter value (i.e., q value) of every stencil leg-interface intersection is determined and stored so that these intersections are only determined once per iteration. The phase types of all nodes with intersected stencils are then determined from an examination of the interface normals at the intersection points. If the normal points away from a node, it is a "minus" or solid node. If the normal points towards a node, it is a "plus" or liquid node. Nodes that lie exactly on the interface are assumed to be solid. This determines the phase of all the nodes adjacent to the interface. Since the phase of the nodes is known initially and can only change when the node is adjacent to the interface, this is sufficient to track the phase of all the nodes for the duration of the computation. Note that the cost of these operations only increases linearly as the mesh is refined (i.e., $O\left(\Delta x^{-1}\right)$).

Once the nodal phases and stencil intersections are known at time t_n and, using the current guess at the interface location (discussed later), at time t_{n+1}, the nodal phase changes can be determined. Using a list of all nodes adjacent to the interface, we determine those nodes whose phases are different at time t_n and t_{n+1}. For each of these nodes, we use the pre-calculated stencil intersection data to determine a nearby point on the interface. Using the closest intersection as an initial guess, (q_I, t_I), the time and exact point of intersection between the interface and computational node can be found using Newton's method. The system that must be solved is

$$x = \frac{t_I X^{n+1}(q_I) + (\Delta t - t_I) X^n(q_I)}{\Delta t}, \tag{4.112}$$

$$y = \frac{t_I Y^{n+1}(q_I) + (\Delta t - t_I) Y^n(q_I)}{\Delta t}, \tag{4.113}$$

where (x, y) is the location of the node undergoing the phase change, q_I is the parameter value associated with the point on the interface that hits the node and t_I is the time that elapses between the start of the time step and the intersection. Note that the above system is independent of the equations being used to evolve the interface. The cost associated with detecting nodal phase changes and calculating the required information only increases linearly as the mesh is refined (i.e., $O\left(\Delta x^{-1}\right)$).

4.5.3 Discretization of spatial derivatives

In Chapter 3 we developed a method for accurately discretizing the spatial derivatives of functions whose values and/or derivatives are discontinuous across an interface. In that chapter, we found that, in the general case of a function satisfying non-symmetric jump conditions, it is necessary to derive custom stencils for all nodes with stencil legs intersected by the interface. For the special case of a function that satisfies symmetric jump conditions, however, the standard stencil formulas can be used. In this case, a simple additive correction is all that is required. We will demonstrate, by direct calculation, that the temperature in the symmetric (equal material properties in both phases) dendritic solidification problem obeys symmetric jump conditions at the interface.

Calculation of the jump conditions

Recall that a function, θ, satisfies symmetric jump relations at the interface, (X, Y), if it is possible to calculate the jump of each derivative independently. That is, we assume that the value of the left-hand side of

$$\left[\frac{\partial^n \theta\left(X,Y\right)}{\partial x^k \partial y^{n-k}}\right] = \left(\frac{\partial^n \theta\left(X,Y\right)}{\partial x^k \partial y^{n-k}}\right)^+ - \left(\frac{\partial^n \theta\left(X,Y\right)}{\partial x^k \partial y^{n-k}}\right)^- \tag{4.114}$$

is known for $0 \leq k \leq n$, where

$$\left(\frac{\partial^n \theta\left(X,Y\right)}{\partial x^k \partial y^{n-k}}\right)^+ = \lim_{\varepsilon \to 0+} \frac{\partial^n \theta\left(X + \varepsilon n_x, Y + \varepsilon n_y\right)}{\partial x^k \partial y^{n-k}} \tag{4.115}$$

and

$$\left(\frac{\partial^n \theta\left(X,Y\right)}{\partial x^k \partial y^{n-k}}\right)^- = \lim_{\varepsilon \to 0+} \frac{\partial^n \theta\left(X - \varepsilon n_x, Y - \varepsilon n_y\right)}{\partial x^k \partial y^{n-k}}, \tag{4.116}$$

and (n_x, n_y) is interface normal defined previously. As discussed in Chapter 3, such jump conditions are called symmetric because of their mathematical character. However, we should note that symmetric jump conditions often arise from physical problems that have symmetric (equal) material properties on each side of the interface (i.e., in each phase). Such is the case for the temperature of a symmetric material undergoing dendritic solidification.

The equations governing the temperature of a symmetric material undergoing dendritic solidification are

$$\frac{\partial \theta_L}{\partial t} = H\nabla^2\theta_L \tag{4.117}$$

in the liquid phase and

$$\frac{\partial \theta_S}{\partial t} = H\nabla^2\theta_S \tag{4.118}$$

in the solid phase. On the interface, represented parametrically by $(X\,(q,t)\,, Y\,(q,t))$, the temperature must satisfy

$$\theta_S = \theta_L = -\varepsilon_c\,(\mathbf{n})\,\kappa \tag{4.119}$$

and

$$(h\nabla\theta_S - h\nabla\theta_L)\cdot\mathbf{n} = n_x\frac{\partial X}{\partial t} + n_y\frac{\partial Y}{\partial t}. \tag{4.120}$$

We can combine (4.117) and (4.118) and write them, formally at least, as a single Poisson equation,

$$\nabla^2\theta = f = \frac{1}{H}\frac{\partial\theta}{\partial t}, \tag{4.121}$$

subject to known jump conditions,

$$\begin{aligned}\theta^+\,(X,Y,t) - \theta^-\,(X,Y,t) &= [\theta]\,, &(4.122)\\ \theta_n^+\,(X,Y,t) - \theta_n^-\,(X,Y,t) &= [\theta_n]\,, &(4.123)\end{aligned}$$

across the interface. For dendritic solidification the temperature is continuous across the interface, so

$$[\theta] = 0, \tag{4.124}$$

and the jump in the normal derivative is related to the interface velocity by

$$[\theta_n] = \frac{1}{h}\left(n_x \frac{\partial X}{\partial t} + n_y \frac{\partial Y}{\partial t}\right). \tag{4.125}$$

In Chapter 3 we derived the jump conditions for a Poisson problem (4.121) subject to jump conditions (4.122) and (4.123). From those results we know that the jumps in the first derivatives are given by

$$\begin{aligned} [\theta_x] &= n_x [\theta_n], &(4.126)\\ [\theta_y] &= n_y [\theta_n], &(4.127)\end{aligned}$$

the jumps in the second derivatives are given by

$$\begin{aligned} [\theta_{xx}] &= +\tau_x \frac{\partial [\theta_x]}{\partial s} - \tau_y \frac{\partial [\theta_y]}{\partial s} + \tau_y^2 [f], &(4.128)\\ [\theta_{xy}] &= +\tau_y \frac{\partial [\theta_x]}{\partial s} + \tau_x \frac{\partial [\theta_y]}{\partial s} - \tau_x \tau_y [f], &(4.129)\\ [\theta_{yy}] &= -\tau_x \frac{\partial [\theta_x]}{\partial s} + \tau_y \frac{\partial [\theta_y]}{\partial s} + \tau_x^2 [f], &(4.130)\end{aligned}$$

and the jumps in the third derivatives are given by

$$\begin{aligned} [\theta_{xxx}] &= +\tau_x\left(1+\tau_y^2\right) \frac{\partial [\theta_{xx}]}{\partial s} - \tau_y \frac{\partial [\theta_{xy}]}{\partial s} + \tau_x \tau_y^2 \frac{\partial [\theta_{yy}]}{\partial s} - \tau_y^3 [f_n], &(4.131)\\ [\theta_{xxy}] &= +\tau_y^3 \frac{\partial [\theta_{xx}]}{\partial s} + \tau_x \frac{\partial [\theta_{xy}]}{\partial s} - \tau_x^2 \tau_y \frac{\partial [\theta_{yy}]}{\partial s} + \tau_x \tau_y^2 [f_n], &(4.132)\\ [\theta_{xyy}] &= -\tau_x \tau_y^2 \frac{\partial [\theta_{xx}]}{\partial s} + \tau_y \frac{\partial [\theta_{xy}]}{\partial s} + \tau_x^3 \frac{\partial [\theta_{yy}]}{\partial s} - \tau_x^2 \tau_y [f_n], &(4.133)\\ [\theta_{yyy}] &= +\tau_x^2 \tau_y \frac{\partial [\theta_{xx}]}{\partial s} - \tau_x \frac{\partial [\theta_{xy}]}{\partial s} + \tau_y\left(1+\tau_x^2\right) \frac{\partial [\theta_{yy}]}{\partial s} + \tau_x^3 [f_n]. &(4.134)\end{aligned}$$

Before we can evaluate the above jumps, two issues need to be addressed. First, all of the above quantities are actually functions of the parameterization variable $q \in [0,1]$, not arc length. Derivatives with respect to arc length, therefore, should be taken to mean

$$\frac{\partial}{\partial s} = \left[\left(\frac{\partial X}{\partial q} \right)^2 + \left(\frac{\partial Y}{\partial q} \right)^2 \right]^{-1} \frac{\partial}{\partial q}. \tag{4.135}$$

The distinction is subtle but important. The length of the interface, $S(t)$, during dendritic solidification typically grows. In this case, parametrizing the interface variables in terms of the arc length, $s \in [0, S(t)]$, introduces an implicit function of time which is inconvenient. When all the interface variables are parametrized in terms of a variable that varies over a fixed range, this implicit time dependence is eliminated. The other issue is that the evaluation of $[\theta_x]$, $[\theta_y]$ and the other jumps requires that the discontinuity in the forcing, $[f]$, and its normal derivative, $[f_n]$, be known. From (4.121) we have

$$f = \frac{1}{H} \frac{\partial \theta}{\partial t} \tag{4.136}$$

which implies

$$[f] = \frac{1}{H} [\theta_t] \tag{4.137}$$

and

$$[f_n] = \frac{1}{H} [\theta_{nt}]. \tag{4.138}$$

The jump in the time derivative follows from (4.124) and the differentiation of (4.122) with respect to time,

$$\frac{\partial [\theta]}{\partial t} = 0 = \left(\theta_t^+ + \frac{\partial X}{\partial t} \theta_x^+ + \frac{\partial Y}{\partial t} \theta_y^+ \right) - \left(\theta_t^- + \frac{\partial X}{\partial t} \theta_x^- + \frac{\partial Y}{\partial t} \theta_y^- \right), \tag{4.139}$$

which implies

$$[\theta_t] = -\left(\frac{\partial X}{\partial t} [\theta_x] + \frac{\partial Y}{\partial t} [\theta_y] \right). \tag{4.140}$$

It is possible to simplify this even further. Substituting (4.126) and (4.127) into (4.140) and using (4.125) yields

$$[\theta_t] = -\left(\frac{\partial X}{\partial t} n_x + \frac{\partial Y}{\partial t} n_y \right) [\theta_n] = -h [\theta_n]^2. \tag{4.141}$$

The value of $[\theta_{nt}]$ can not be represented quite so succinctly. It is determined by differentiating (4.126) and (4.127) with respect to time which, after some manipulation, yields

$$[\theta_{xt}] = \frac{\partial [\theta_x]}{\partial t} - \left(\frac{\partial X}{\partial t} [\theta_{xx}] + \frac{\partial Y}{\partial t} [\theta_{xy}] \right), \tag{4.142}$$

$$[\theta_{yt}] = \frac{\partial [\theta_y]}{\partial t} - \left(\frac{\partial X}{\partial t} [\theta_{xy}] + \frac{\partial Y}{\partial t} [\theta_{yy}] \right), \tag{4.143}$$

and allows us to calculate

$$[\theta_{nt}] = n_x [\theta_{xt}] + n_y [\theta_{yt}] . \tag{4.144}$$

Note that given the value of $[\theta_n] = [\theta_n](q, t)$, we can calculate $[f]$ and $[f_n]$ and, therefore, all the discontinuities in the temperature derivatives across the interface through third order in space and first order in time. In fact, using the above results it is quite simple to also calculate the discontinuity in the second time derivative of the temperature. Indeed, differentiating the definition of $[\theta_t]$ with respect to time yields

$$[\theta_{tt}] = \frac{\partial [\theta_t]}{\partial t} - \left(\frac{\partial X}{\partial t} [\theta_{xt}] + \frac{\partial Y}{\partial t} [\theta_{yt}] \right), \tag{4.145}$$

which expresses $[\theta_{tt}]$ in terms of the known value of $[\theta_n]$ and its derivatives.

Examination of the above jump conditions reveals that the temperature of a symmetric material undergoing dendritic solidification does indeed satisfy symmetric jump conditions. This makes the calculation of the spatial derivatives significantly cheaper and easier.

Calculation of the stencils

We demonstrated above that the temperature of a symmetric material satisfies symmetric jump conditions. From Chapter 3, we know that there is no need to derive custom stencils for such functions. Indeed, we can use a single form for the discrete

Laplacian, given by

$$\nabla^2\theta(i,j)=\frac{1}{6\Delta x^2}\begin{bmatrix}1&4&1\\4&-20&4\\1&4&1\end{bmatrix}\theta(i,j)+C(i,j), \tag{4.146}$$

or

$$\begin{aligned}\nabla^2\theta(i,j) &= \frac{1}{6\Delta x^2}\{4\theta(i+1,j)+4\theta(i-1,j)+4\theta(i,j+1)+4\theta(i,j-1)\\ &\quad -20\theta(i,j)+\theta(i+1,j+1)+\theta(i-1,j+1)\\ &\quad +\theta(i+1,j-1)+\theta(i-1,j-1)\}+C(i,j),\end{aligned} \tag{4.147}$$

at all grid nodes regardless of their position relative to the interface. Note that we have elected to use (4.147) instead of the more traditional formula for the Laplacian given by

$$\nabla^2\theta(i,j)=\frac{1}{\Delta x^2}\begin{bmatrix}0&1&0\\1&-4&1\\0&1&0\end{bmatrix}\theta(i,j)+C(i,j). \tag{4.148}$$

The traditional stencil is perfectly appropriate far from the interface. We have found, however, that the robustness of the numerical method is improved if we instead use a stencil that samples the grid in all possible directions instead of just along the coordinate axes. This extra sampling is mainly important close to the interface but, as we shall see shortly, it is important for efficiency reasons to uniformly apply the same stencil at all points in the domain.

The interface correction, $C(i,j)$, is zero at all stencils not intersected by the interface. The value of the correction for intersected stencils will generally be nonzero and can be calculated in the standard way we described in Chapter 3. For instance, suppose we wish to calculate the Laplacian at the $\theta(i,j)$ node but the interface lies between the $\theta(i,j)$ and $\theta(i+1,j+1)$ nodes, as shown in Fig. 4.12. If the position of $\theta(i+1,j+1)$ relative to the interface intersection point (X^*,Y^*) is $(\bar{x},\bar{y})$, then

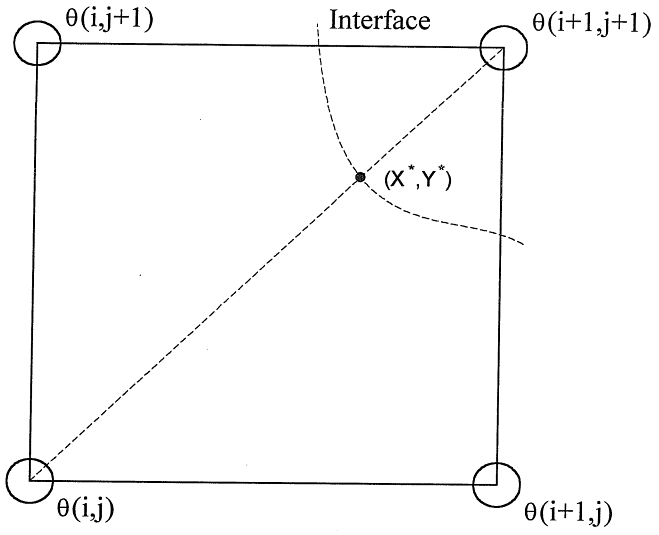

Figure 4.12: Two temperature nodes separated by the interface

the correction associated with the intersection is

$$C^* = \bar{x}\left[\theta_x\right] + \bar{y}\left[\theta_y\right] + \frac{1}{2!}\left(\bar{x}^2\left[\theta_{xx}\right] + 2\bar{x}\bar{y}\left[\theta_{xy}\right] + \bar{y}^2\left[\theta_{yy}\right]\right), \tag{4.149}$$

to second order or

$$\begin{aligned} C^* \quad = \quad & \bar{x}\left[\theta_x\right] + \bar{y}\left[\theta_y\right] + \frac{1}{2!}\left(\bar{x}^2\left[\theta_{xx}\right] + 2\bar{x}\bar{y}\left[\theta_{xy}\right] + \bar{y}^2\left[\theta_{yy}\right]\right) \\ & + \frac{1}{3!}\left(\bar{x}^3\left[\theta_{xxx}\right] + 3\bar{x}^2\bar{y}\left[\theta_{xxy}\right] + 3\bar{x}\bar{y}^2\left[\theta_{xyy}\right] + \bar{y}^3\left[\theta_{yyy}\right]\right), \end{aligned} \tag{4.150}$$

to third order. The above correction can then be incorporated into the stencil formula by substituting

$$\theta\left(i+1, j+1\right) \rightarrow \theta\left(i+1, j+1\right) + C^* \tag{4.151}$$

into (4.147) if $\theta\left(i, j\right)$ is a plus node (i.e., the interface normal points towards $\theta\left(i, j\right)$), or substituting

$$\theta\left(i+1, j+1\right) \rightarrow \theta\left(i+1, j+1\right) - C^* \tag{4.152}$$

into (4.147). This is equivalent to (4.147) with

$$C\left(i,j\right) = \pm\frac{C^*}{\Delta x^2}. \tag{4.153}$$

If the interface intersected other legs of the stencil as well, the correction associated with each intersection would be calculated and included in $C\left(i,j\right)$ as described above.

4.5.4 Discretization of governing equations

Our numerical scheme determines the temperature in the liquid and solid phases simultaneously through the solution of

$$\theta^{n+1}\left(i,j\right) = \theta^n\left(i,j\right) + H\Delta t\left\{\alpha\nabla^2\theta^{n+1}\left(i,j\right) + \left(1-\alpha\right)\nabla^2\theta^n\left(i,j\right)\right\} + E\left(i,j\right), \tag{4.154}$$

where the Laplacian is discretized using the stencils presented above and where $E\left(i,j\right)$ is a time correction term which is only non-zero at nodes that change phase during the time step. The exact value of the time correction is determined using the time derivative corrections we derived in Chapter 3. Using those results we have

$$E\left(i,j\right) = \begin{cases} 0 & \text{no phase change at node } \left(i,j\right) \\ \left\{\left(1-\alpha\right)\Delta t - t_I\right\}\left[\theta_t\right] & \text{phase change with } \alpha \neq \frac{1}{2} \\ \left\{\frac{1}{2}\Delta t - t_I\right\}\left[\theta_t\right] + \frac{1}{2}t_I\left(t_I - \Delta t\right)\left[\theta_{tt}\right] & \text{phase change with } \alpha = \frac{1}{2} \end{cases}, \tag{4.155}$$

where the time the interface intersects the node, $t_n < t_n + t_I \leq t_{n+1}$, is used to calculate the jump in the first time derivative. That is, we use

$$\left[\theta_t\right] = \frac{1}{\Delta t}\left\{\left(\Delta t - t_I\right)\left[\theta_t\right]\left(q_I, t_n\right) + t_I\left[\theta_t\right]\left(q_I, t_{n+1}\right)\right\}, \tag{4.156}$$

where q_I is the parameter value associated with the interface point that intersects the (i,j) node. The jump in the second derivative is evaluated at t_{n+1}. Note that, like the spatial stencils, the time derivative can be calculated using a standard uncorrected stencil with only a simple additive correction required.

From the construction of the spatial stencils, we know that the solution of (4.154) will automatically satisfy

$$[\theta_n] = \frac{1}{h}\left(n_x\frac{\partial X}{\partial t} + n_y\frac{\partial Y}{\partial t}\right). \tag{4.157}$$

In order for the Gibbs-Thomson condition,

$$\theta\left(X, Y, t\right) = -\varepsilon_c\left(\mathbf{n}\right)\kappa, \tag{4.158}$$

to be satisfied, however, the interface must move at the correct velocity. Indeed, we will see shortly that (4.158) can be used as a measure of the error of the interface velocity. Unfortunately, the interface marker particles are generally not located at the computational nodes so before we can use (4.158), we need an interpolation scheme to determine the temperature at the marker locations.

4.5.5 Interpolation of interface temperature

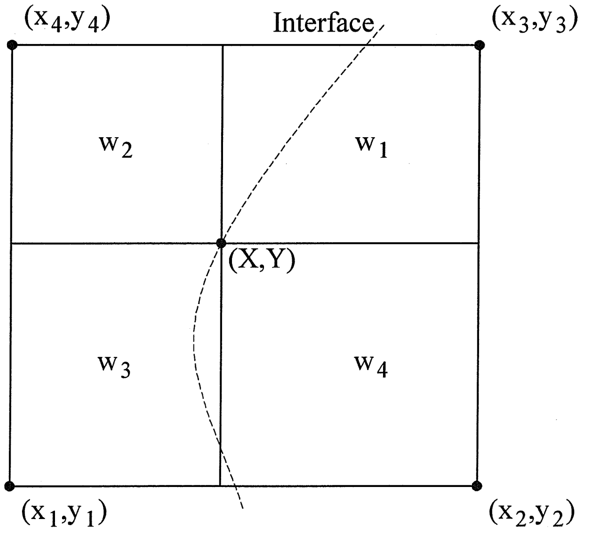

Figure 4.13: Geometric definition of the Taylor interpolation weights

We interpolate the temperature at the marker particles using a generalization of the weighted Taylor series technique we developed in one dimension. Consider an interface marker particle located at (X, Y) which lies between the computational nodes located at (x_i, y_i), $i = 1, 2, 3, 4$, as shown in Fig. 4.13. Using properly corrected stencils, we estimate the value of the temperature at (X, Y) using a Taylor series expansion about each computational node, $i = 1, 2, 3, 4$,

$$\begin{aligned} \theta(X,Y) &= \theta_i = \theta(x_i,y_i) + (X - x_i)\,\theta_x(x_i,y_i) \\ &+\frac{1}{2!}(X-x_i)^2\,\theta_{xx}(x_i,y_i) + (Y-y_i)\,\theta_y(x_i,y_i) \\ &+\frac{1}{2!}(Y-y_i)^2\,\theta_{yy}(x_i,y_i) + (X-x_i)(Y-y_i)\,\theta_{xy}(x_i,y_i) + O\left(\Delta x^2\right). \end{aligned} \tag{4.159}$$

Note that the accuracy of the above is limited to second order despite the order of the expansion because that is the accuracy to which the nodal values are known. The temperature at the marker particle is then estimated using a weighted average,

$$\theta(X,Y) = \sum_{i=1}^{4} w_i \theta_i, \tag{4.160}$$

where the weights, w_i, are area fractions given by

$$w_1 = \frac{(x_2 - X)(y_4 - Y)}{\Delta x^2}, \tag{4.161}$$

$$w_2 = \frac{(X - x_1)(y_4 - Y)}{\Delta x^2}, \tag{4.162}$$

$$w_3 = \frac{(X - x_1)(Y - y_1)}{\Delta x^2}, \tag{4.163}$$

and

$$w_4 = \frac{(x_2 - X)(Y - y_1)}{\Delta x^2}. \tag{4.164}$$

This defines an interpolation scheme that has continuously varying dependence on the nodal values as the interface moves through the grid.

4.5.6 Iterative scheme for interface velocity

The interface velocity at each time step is unknown and must be determined to successfully evolve the temperature forward in time. It is only defined on the interface and we discretize it using a periodic cubic spline, $V_N\left(q\right)$, that interpolates between the values determined at the interface marker particles. The temperature has a non-linear dependence on V_N so an iterative scheme is required. In our scheme, the error associated with a guess for the normal velocity is defined by the Gibbs-Thomson condition,

$$\theta = -\varepsilon_c\left(\mathbf{n}\right)\kappa. \tag{4.165}$$

When the above is satisfied, we have correctly determined V_N. The goal of our iterative solver, therefore, is to satisfy the Gibbs-Thomson condition at each interface marker particle to within a desired tolerance (typically, $tol = 10^{-5}$). The error at the i^{th} marker is given by

$$e_i = \theta^{n+1}\left(X_i, Y_i\right) + \varepsilon_c\left(\mathbf{n}_i\right)\kappa_i. \tag{4.166}$$

As we shall see, to determine e_i we need an initial starting guess for V_N^{n+1}, the unknown normal velocity at time t_{n+1}. For the first time step we use the normal velocity supplied with the initial conditions of the problem. For subsequent time steps, $V_N^{n+1} = V_N^n$ is the initial guess. The algorithm to determine e_i is,

1. Using the current guess for the normal velocity of the interface, advance the interface marker particles forward to their t_{n+1} positions.

2. Using the new interface position, compute all stencil intersections, node types and stencil corrections.

3. Compute the new temperature field.

4. Determine the new temperature at each interface marker particle.

5. Calculate the amount by which the Gibbs-Thomson condition is violated at each marker,

$$e_i = \theta^{n+1}\left(X^{n+1}(q_i), Y^{n+1}(q_i)\right) + \varepsilon_c\left(\mathbf{n}^{n+1}(q_i)\right)\kappa^{n+1}(q_i). \tag{4.167}$$

The above algorithm allows us to calculate the amount by which the non-linear system defining the normal velocity at the marker particles, $V_N^{n+1}(q_i)$ or V_N^{n+1} for short, is not satisfied. The details associated with the calculation of e_i are nested in the above algorithm so any iterative scheme for an implicitly defined discrete non-linear system could be employed. This problem is challenging because the calculation of the error is relatively expensive. Note that each calculation of the error involves the solution of the heat equation. If we wished to simulate a problem with non-symmetric material properties, this would require the determination of custom stencils adjacent to the interface and an expensive system inversion for the temperature. If the material properties are symmetric, on the other hand, we need only determine an additive correction for the stencils adjacent to the interface and the temperature system can be inverted efficiently using a FFT (Fast Fourier Transform) based solver (see [68]). This is why we have restricted attention to problems with symmetric material properties in two dimensions. Even using a "fast solver," however, the evaluation of the error is expensive which makes the construction of a Jacobian by numerical differentiation undesirable. It is possible to use the pseudo time stepping scheme that was utilized in one dimension,

$$V_N^{n+1} := V_N^{n+1} - e_i/\omega, \tag{4.168}$$

but the Jacobian appears to vary too much with time to allow a single guess for ω to be effective over long periods of time. To address this problem, we have developed a variant of Broyden's method (see [9]) that automatically calculates a guess for ω after each iteration. We have also implemented a line search algorithm that attempts to minimize the error associated with the normal velocity along the "search direction" defined by $-e_i/\omega$. The complete algorithm is

1. Determine the error associated with the current guess, e_i.

2. Save the current error,
$$e_i^* = e_i. \tag{4.169}$$

3. Save the current guess,
$$V^* = V_N^{n+1}. \tag{4.170}$$

4. Calculate a measure of the error associated with current guess,
$$g_0 = \|e_i\|_2^2 = \sum_{i=0}^{N_c-1} (e_i)^2. \tag{4.171}$$

5. Set the "search direction" associated with current guess,
$$\delta V_i = -e_i/\omega. \tag{4.172}$$

6. Calculate a new guess using
$$V_N^{n+1} = V^* + \frac{1}{2}\delta V_i, \tag{4.173}$$

   error, e_i, and error measure
$$g_{1/2} = \|e_i\|_2^2 = \sum_{i=0}^{N_c-1} (e_i)^2. \tag{4.174}$$

7. Calculate new guess
$$V_N^{n+1} = V^* + \delta V_i, \tag{4.175}$$

   error, e_i, and error measure
$$g_1 = \|e_i\|_2^2 = \sum_{i=0}^{N_c-1} (e_i)^2. \tag{4.176}$$

8. Using quadratic interpolation ($\mu = 0,\ 1/2,\ 1$), construct an "error function," $g(\mu)$, which is a measure of the error associated with guesses of the form

$$V_N^{n+1} = V^* + \mu \delta V_i. \tag{4.177}$$

9. If $g(\mu)$ has a minimum, $\frac{1}{10} < \mu_{\min} < 10$, use that value to calculate a new guess,

$$V_N^{n+1} = V^* + \mu_{\min} \delta V_i, \tag{4.178}$$

and the associated error, e_i. If there is no minimum, the guess associated with $\mu = 1$ is retained.

10. Compute an approximate Jacobian using

$$\omega = \frac{\sum_{i=0}^{N_c-1} (e_i - e_i^*) \left(V_N^{n+1} - V^*\right)}{\sum_{i=0}^{N_c-1} \left(V_N^{n+1} - V^*\right)^2}. \tag{4.179}$$

11. If the error at any of the markers is too big (i.e., $\|e_i\|_\infty > tol$), go back to 2.

The above calculation for ω is a specialization of Broyden's update formula for a Jacobian of the form $\mathbf{J} = \omega \mathbf{I}$. The heuristic derivation follows from the definition of the Jacobian,

$$J_{ij} = \frac{\partial e_i}{\partial V_j}. \tag{4.180}$$

Replacing the differentials by finite changes and substituting in $J_{ij} = \omega \delta_{ij}$, we find

$$\omega \delta V_i = \delta e_i, \tag{4.181}$$

which, in general, has no solution. If it does have a solution, however, the solution is given by

$$\omega = \frac{\sum_{i=0}^{N_c-1} \delta e_i \delta V_i}{\sum_{i=0}^{N_c-1} (\delta V_i)^2}, \tag{4.182}$$

which we use to calculate the approximate Jacobian above. Once the iterative scheme for the normal velocity has converged, we have solutions for both V_N^{n+1} and θ^{n+1} and we can proceed to the next time step, repeating the process.

4.5.7 Complete scheme

Using the algorithms developed above, we can develop a method to solve purely diffusive dendritic solidification problems. We begin the time step with initial values for the temperature, the position of the interface marker particles and the interface velocity at the marker particles. If this is the first time step, this information must be extracted from the initial condition supplied as part of the problem statement. The initial spacing of the marker particles is set to be Δx. It is possible to make this larger, but making it too small can lead to an ill-conditioned system for the velocity. To advance to the next time step, we do the following:

1. Iterate on the normal velocity, V_N^{n+1}, until the error associated with the Gibbs-Thomson condition is sufficiently small, $\|e_i\|_\infty < tol$.

2. Determine the distance along the curve, s_i, of each marker particle and the total arc length, $S = s_{N_c}$, associated with the new interface position determined in step 1.

3. Assuming there are N_c marker particles, if $S/N_c > 5\Delta x/4$, increase the number of marker particles so that $S/N_c = \Delta x$ is satisfied as closely as possible.

4. Using linear interpolation and the values of s_i calculated above, determine the parameter values, q_i^*, associated with the new position of the marker particles such that they are equally spaced along the interface.

5. Regrid the interface quantities X^{n+1}, $Y^{n+1}$and V_N^{n+1} by interpolating their values at $q = q_i^*$ and use these to establish a new set of evenly spaced marker particles.

6. Update all the time dependent quantities and continue with next time step (go back to step 1).

We have used the above algorithm to compute the solution to several different problems. In all cases we select the time step such that

$$\Delta t = \min\left\{\frac{\Delta x^2}{5H}, \Delta t_I\right\}, \tag{4.183}$$

where Δt_I is the largest time step we can take such that the interface travels a distance less than $\Delta x/10$, i.e.,

$$(V_N)_{\max} \Delta t_I \leq \frac{\Delta x}{10}. \tag{4.184}$$

The backward Euler method ($\alpha = 1$) is used to evolve the equations forward in time. Neither the restriction we place on the time step nor our use of the backward Euler method is required for successful evolution of the system. We have found in practice, however, that the above choices result in the most robust method during the later, highly non-linear stages of the simulation.

4.5.8 Results

There are no periodic fully two-dimensional exact solutions for the dendritic solidification problem to validate our method. In Chapter 2, however, we derived an approximate solution using linear perturbation theory. The solution describes the evolution of small periodic disturbances, of wave number a, to the one-dimensional traveling wave problem. To validate our method, we use the linear perturbation solution as the initial condition and check that we do indeed converge to the predicted behavior. For our first test we use parameters $H = 1$, $h = 1$, $V = \frac{1}{2}$, $\delta_0 = \frac{1}{100}$ and $\delta_1 = 0$ (isotropic material). We then simulate the evolution of the fastest growing disturbance associated with these values, which is the $a = 3$ mode.

The computational domain, $x \in [0, 2\pi]$ and $y \in [-1, 6\pi - 1]$, is discretized using equal mesh spacing in both directions ($\Delta x = \Delta y$). Using $\varepsilon = -10^{-4}$, the temperature is set to the linear perturbation solution derived in Chapter 2. The interface is also

N	$\|\theta_N - \theta\|_\infty$	Ratio
32x96	6.1136x10^{-4}	
64x192	1.5673x10^{-4}	3.9007
128x384	3.5469x10^{-5}	4.4188

Table 4.11: Error in the $a = 3$ mode temperature field at $t = 2$

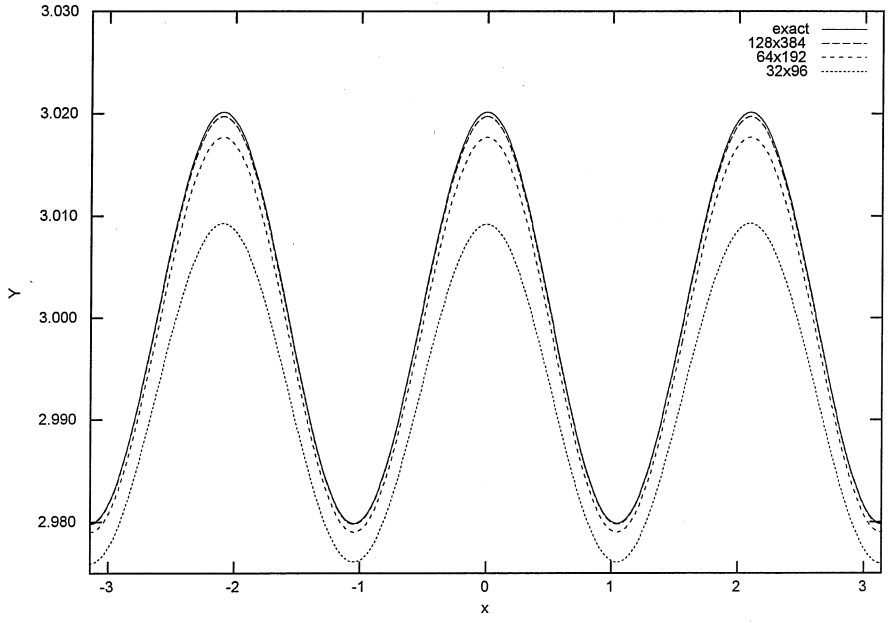

Figure 4.14: Comparison of the position of the $a = 3$ mode interface at $t = 6$ according to linear theory and full simulation with different resolution

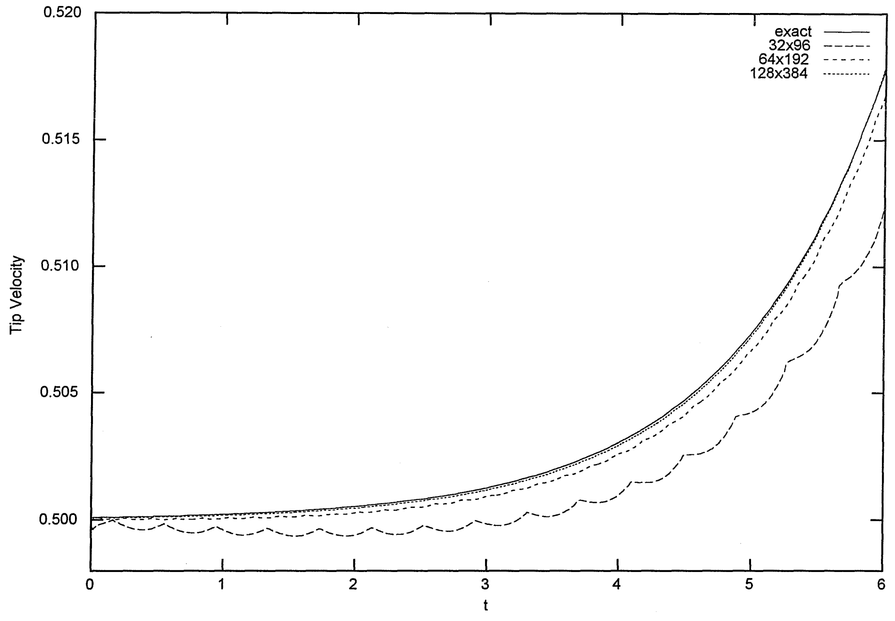

Figure 4.15: Comparison between the linear theory and the full simulation for the $a = 3$ mode tip velocity

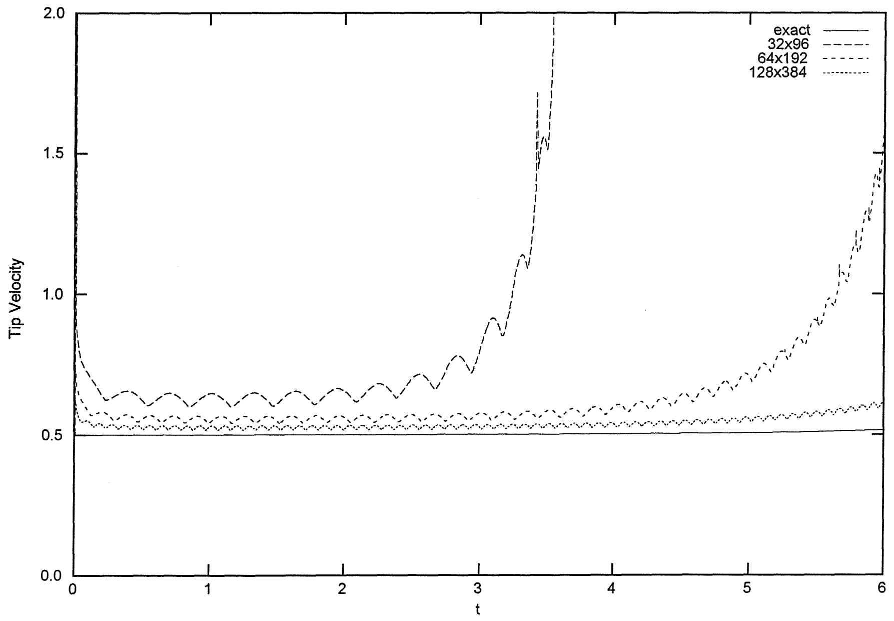

Figure 4.16: Comparison between the linear theory and the full immersed boundary method simulation for the $a = 3$ mode tip velocity

N	σ_N	$\|\sigma_N - \sigma\|_\infty$	Ratio
32x96	0.85332	$3.0919\text{x}10^{-2}$	
64x192	0.87794	$6.2890\text{x}10^{-3}$	4.9163
128x384	0.88284	$1.3913\text{x}10^{-3}$	4.5202

Table 4.12: Resolution study of the growth rate of the $a = 3$ mode

set to its initial shape as dictated by linear perturbation theory,

$$X = 2\pi q, \tag{4.185}$$

$$Y = \varepsilon \cos(aX), \tag{4.186}$$

where $q \in [0, 1]$. Evolving the system forward in time we find good agreement between the numerical and linear perturbation solutions. In Table 4.11, we list the error in the temperature field at time $t = 2$. The second order accuracy of the temperature is apparent. In Fig. 4.14, we have plotted the analytically and numerically determined interfaces at time $t = 6$ for several mesh resolutions. The second order accuracy of the numerically determined interface position is evident. Similarly, in Fig. 4.15 we demonstrate that the tip velocity (maximum interface velocity) generated by the numerical method is second order accurate. It is interesting to compare this with the tip velocities generated by the immersed boundary method. In Fig. 4.16 we have plotted the numerically determined tip velocities for the immersed boundary method over a time range identical to that of Fig. 4.15. As was the case in one dimension, the performance of the immersed boundary method is very poor for this problem. Clearly an extremely small mesh size would be necessary for the immersed boundary method to approach the accuracy of our scheme at even the coarsest resolution.

The convergence of the interface position and velocity implies the convergence of the growth rate. It is useful to be able to extract a numerical value for the growth rate, however, so that we can produce an effective dispersion relation for our method. At the end of each time step, we determine the amplitude of the interface, b, by numerical integration and the orthogonality of trigonometric functions. We then assume that

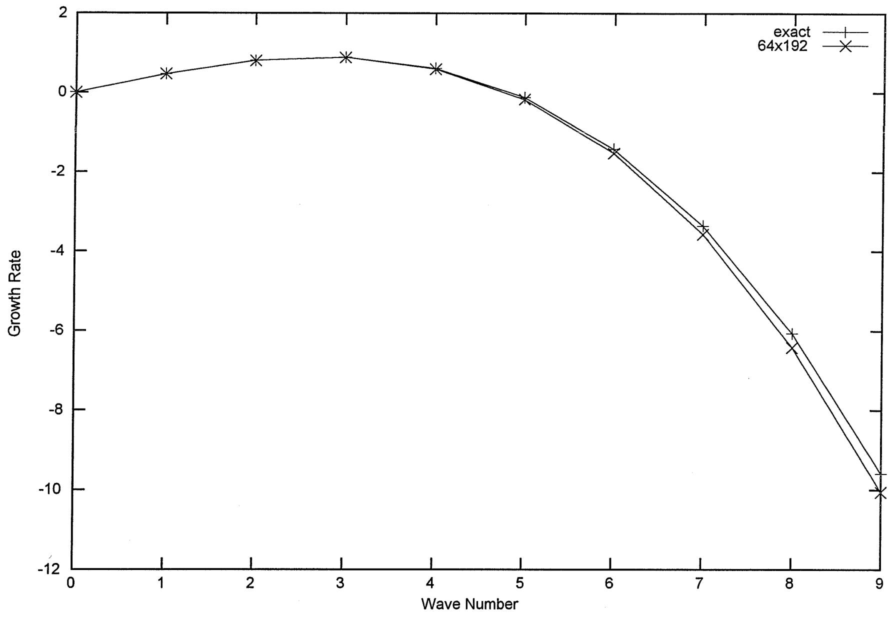

Figure 4.17: Comparison of numerical dispersion relation with linear theory

the amplitude has the form

$$b = c \exp(\sigma t) \tag{4.187}$$

and, taking the log of b, construct a cumulative least squares approximation for σ (i.e., the value of b for all time steps is used to continually update σ). This produces a time dependent value for σ, but it quickly settles down to a steady state value that we take as the growth rate. The growth rates measured by this process for the $a = 3$ mode are listed in Table 4.12 for each resolution. Measuring the growth rates of many different modes allowed us, in Fig. 4.17, to compare the dispersion relation of our numerical scheme with the dispersion relation predicted by linear theory. The agreement is quite good even for the rapidly decaying modes. As a rule, the numerics appear to under predict the growth rate. This observation is consistent with the results in Table 4.12.

Finally, we have plotted the complete history of the numerically determined $a = 3$ interface for the 128x384 simulation. These plots show the superposition of the interface at many different evenly spaced intervals in time. In Fig. 4.18 we have

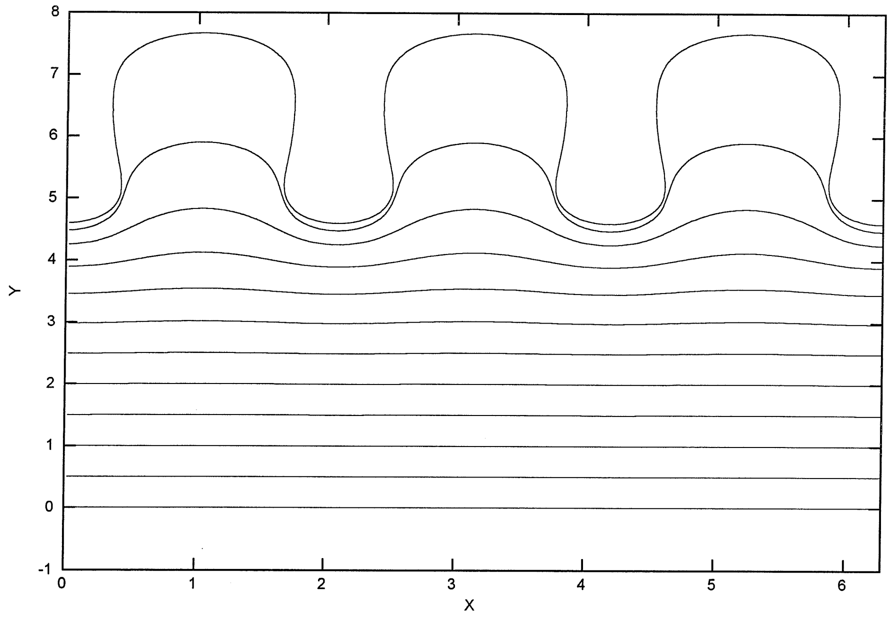

Figure 4.18: Superposition of the $a = 3$ mode for evenly spaced time increments ($\delta t = 1$) begining at $t = 0$

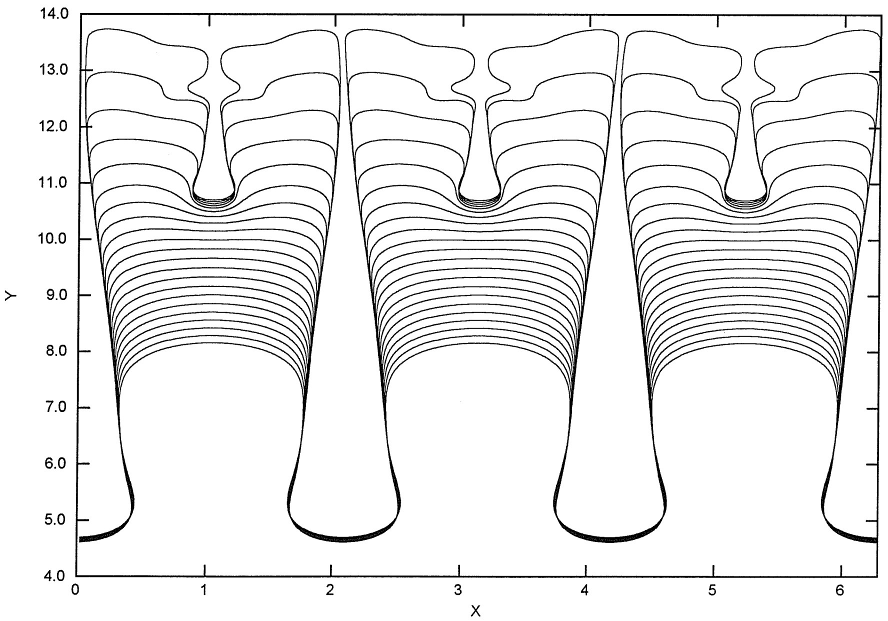

Figure 4.19: Superposition of the $a = 3$ mode solution for evenly spaced time increments ($\delta t = 0.05$) begining at $t = 11.2$

plotted the initial evolution and in Fig. 4.19 we have plotted the evolution at later times. The solid and liquid regions are below the interface and above the interface, respectively, in each plot. Note the tip splitting that occurs late in the simulation. We will see shortly that phenomena such as tip splitting can be influenced by anisotropy in the material properties.

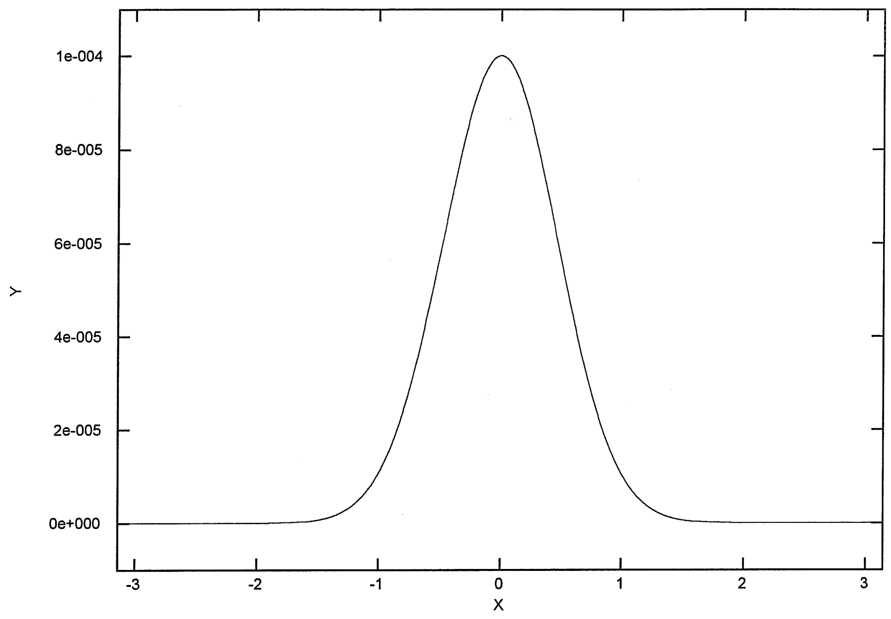

Figure 4.20: Shape of the initial condition for the Jacobi finger problem

For our next example we simulate the evolution of a more concentrated perturbation to the flat interface shown in Fig. 4.20. The shape of the perturbation is given by

$$X = 2\pi\left(q-\frac{1}{2}\right), \tag{4.188}$$

$$Y = \varepsilon\left[\frac{\Theta_2\left(X/2, 0.8\right)}{\Theta_2\left(0, 0.8\right)}\right]^2, \tag{4.189}$$

where $q \in [0,1]$ and Θ_2 is the Jacobi Theta function of the second kind (see [1]) defined by

$$\Theta_2\left(x,q\right) = 2q^{1/4}\sum_{n=0}^{\infty} q^{n(n+1)}\cos\left[\left(2n+1\right)x\right]. \tag{4.190}$$

N	$\|\|\theta_N - \theta\|\|_\infty$	Ratio
32x128	6.0152x10^{-4}	
64x256	1.5460x10^{-4}	3.8907
128x512	3.5314x10^{-5}	4.3780

Table 4.13: Error in Jacobi finger temperature field at $t = 2$

We call the above interface shape a Jacobi finger. It is possible construct an approximate solution to this problem using the solutions to the linear perturbation problem. This is done by numerically computing the Fourier coefficients associated with the interface perturbation and then using these to form the appropriate linear combination of modal solutions to represent the answer. This result is then used to supply the numerical method with conditions and further validate the code. In Table 4.13 we have demonstrated convergence of the temperature field to the analytic solution at time $t = 2$. In Fig. 4.21 we demonstrate the convergence of the numerically predicted interface position. In Fig. 4.22 the convergence of the tip velocity is indicated.

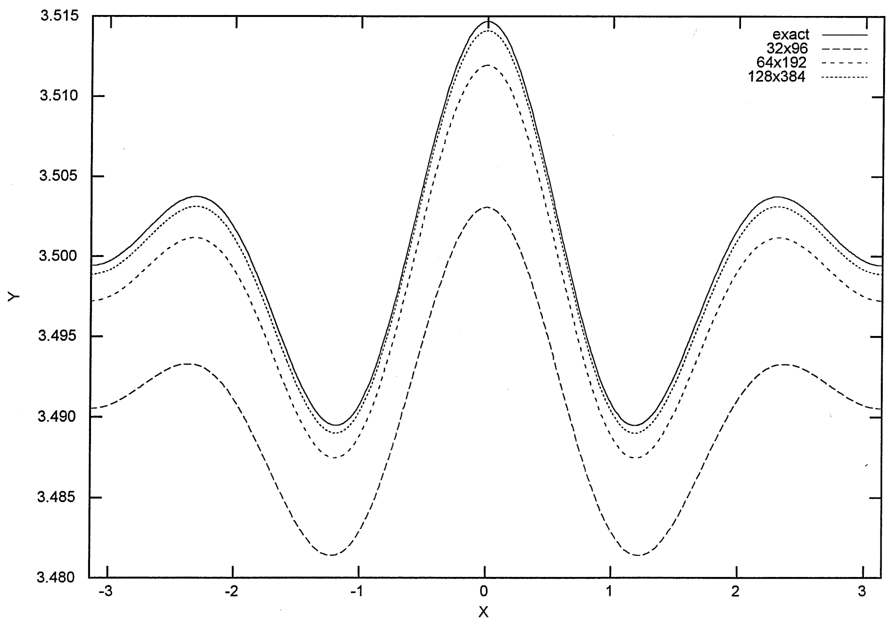

Figure 4.21: Comparison of the position of the Jacobi finger interface at $t = 7$ according to linear theory and full simulation with different resolutions

Finally, we briefly examine the impact that anisotropy can have on the solution.

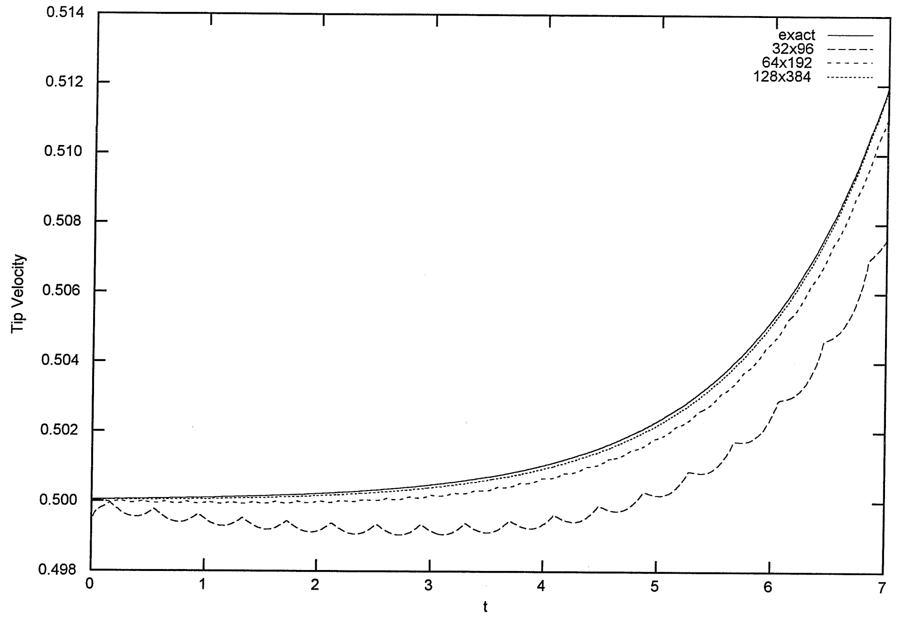

Figure 4.22: Comparison between the linear theory and the full simulation for tip velocity of the Jacobi finger

Note that anisotropy is a higher order effect that has no influence on the linear perturbation solution. Thus, each case starts with exactly the same initial condition. We compute the long time evolution of the Jacobi finger using $V = \frac{1}{2}$, $\delta_0 = \frac{1}{100}$, $\delta_2 = 2$, $\varepsilon = 10^{-4}$ and k_A =0, 4 and 6. The results for $t \geq 11.2$ of the long time simulation for the isotropic case, $k_A = 0$, is shown in Fig. 4.23 while the anisotropic cases of $k_A = 4$ and $k_A = 6$ are shown in Fig. 4.24 and Fig. 4.25, respectively. As before, the solid occupies the region below the interface in each plot. In the isotropic case, $k_A = 0$, the material is equally free to grow in any direction. This case generates the least "pointy" dendrite of the three, and is the only one to undergo tip splitting. In the case of $k_A = 4$, anisotropy introduces preferred growth directions along the x and y-axes. This produces a much more directed "pointy" dendrite that does not tip split. Note the beginnings of side branches near the end of the simulation. Also of interest is the movement of the tip which travels at a constant velocity near the end of the simulation. Similar comments apply to the $k_A = 6$ case. In this material, the anisotropy introduces preferred growth directions along the y-axis and along diagonals offset from this axis by $\pi/3$ radians. This results in the most sharply defined dendrite of the three with the most pronounced side branching near the end of the simulation. The tip in Fig. 4.25 also begins to move with constant velocity near the end of the simulation. The final interface shape generated for the $k_A = 6$ case is plotted in Fig. 4.26.

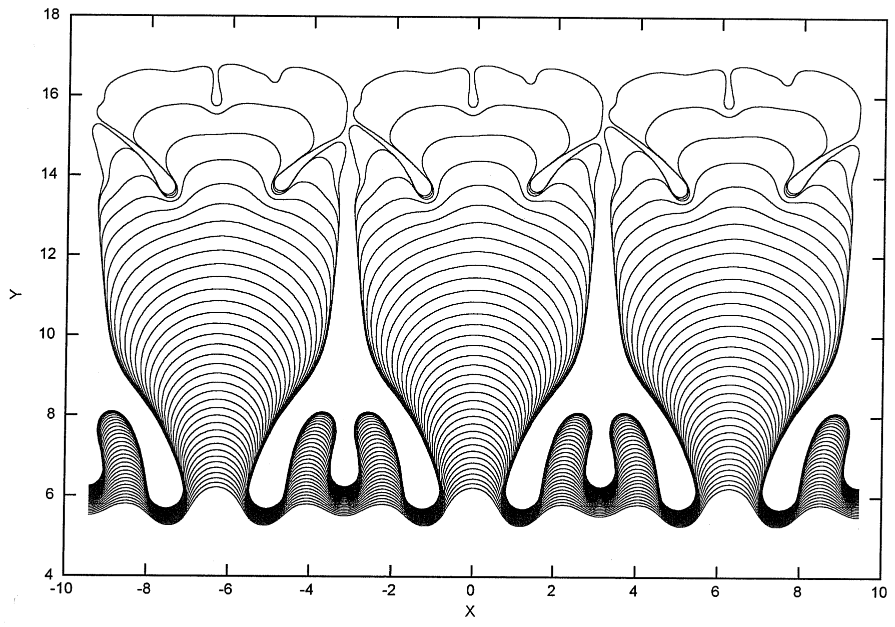

Figure 4.23: Superposition of the isotropic Jacobi finger solution ($k_A = 0$) for evenly spaced time increments ($\delta t = 0.10$) begining at $t = 11.2$

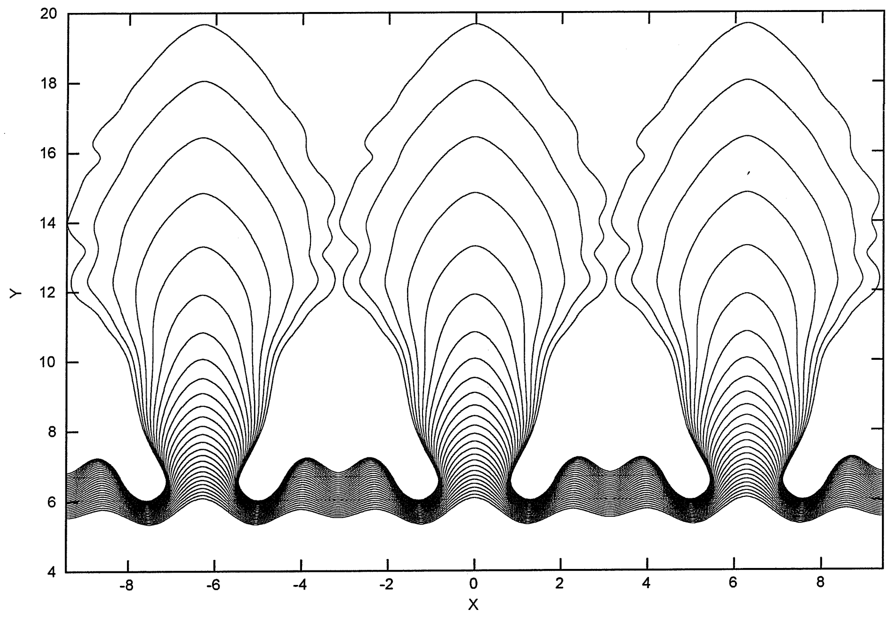

Figure 4.24: Superposition of the anisotropic Jacobi finger solution ($k_A = 4$) for evenly spaced time increments ($\delta t = 0.10$) begining at $t = 11.2$

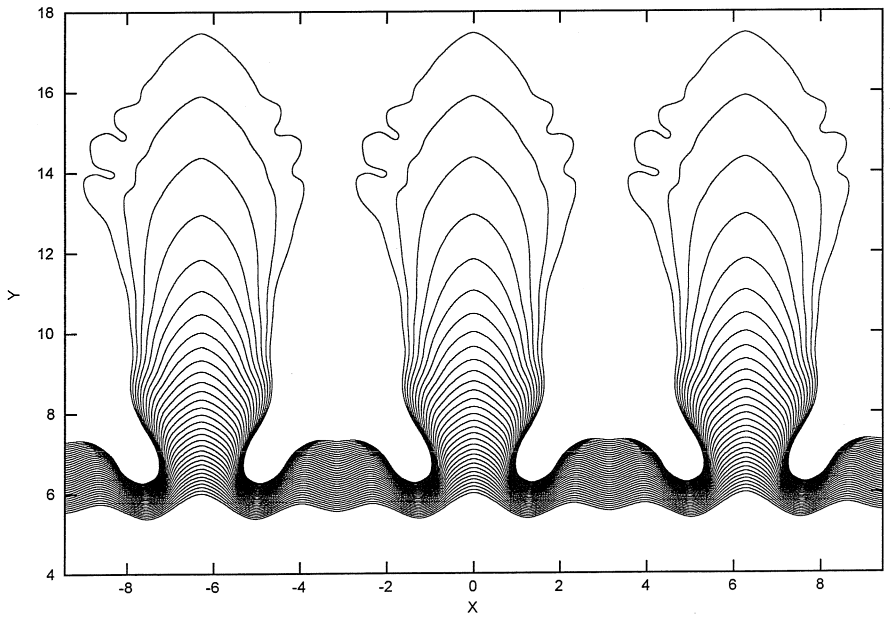

Figure 4.25: Superposition of the anisotropic Jacobi finger solution ($k_A = 6$) for evenly spaced time increments ($\delta t = 0.10$) begining at $t = 11.2$

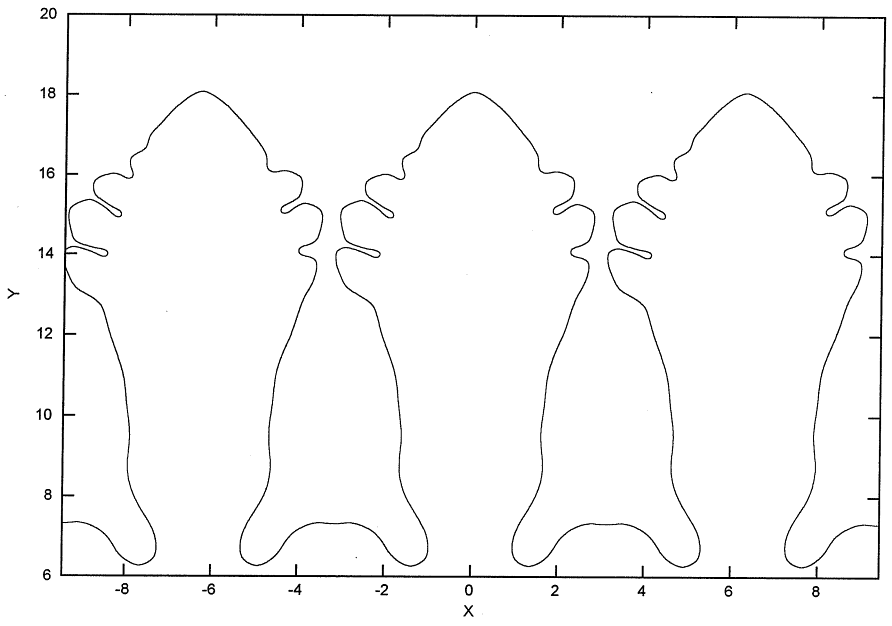

Figure 4.26: Final frame of the anisotropic Jacobi finger solution ($k_A = 6$)

Chapter 5 Incompressible flow in an irregular domain

5.1 Introduction

In order to simulate convection in the liquid phase during solidification, it is necessary to compute the flow in a complicated time-dependent geometry. In this chapter we will focus on determining the flow in an irregular domain with fixed boundaries. The method will be extended to include moving boundaries in Chapter 6.

Our method utilizes an embedded interface similar to the one discussed in Chapter 4. We will treat the entire computational domain as if it were all liquid. Inside the domain, the position of the liquid-solid interface is tracked by equally spaced marker particles or control points. The position of these control points is independent of the computational grid except that we attempt to maintain a spacing between them of approximately $2\Delta x$, where Δx is the uniform spacing between the computational nodes in both the vertical and horizontal directions. The points on the interface in between the control points are represented by a periodic cubic spline that passes through the marker particles. The fluid motion is influenced by the interface through the forcing that it applies along its normal and tangential directions. In our problems the interface is represented parametrically by $(X(q,t), Y(q,t))$, where $q \in [0,1]$. The interface, in general, does not coincide with the computational nodes, so incorporating the interface forcing into the momentum equations can be cumbersome. One convenient way of representing the interface force is in terms of a body force that is added to the right-hand side of the momentum equations. This body force is given by

$$\mathbf{F} = \int_0^1 \mathbf{f}(q,t)\,\delta(x-X, y-Y)\,dq, \tag{5.1}$$

where $\mathbf{f}$ is the interfacial forcing and δ is a two-dimensional delta function. We never actually explicitly evaluate (5.1). Instead, it is used analytically in the derivation of the jumps conditions on the flow velocity and pressure. These jumps are then incorporated directly into the computational stencils that intersect the interface using the techniques developed in Chapter 3. In order to maintain a desired velocity on the interface, we iterate on $\mathbf{f}(q, t)$ until the velocity on the interface, interpolated off the computational grid, takes on its desired value. Thus we compute a fluid flow in both the liquid and physically solid regions of the domain. The flow in each region satisfies the interface velocity boundary condition, but only the velocity in the liquid region is physically relevant.

5.2 Existing numerical methods

An extensive body of literature exists concerning the simulation of fluid flow in fixed irregular geometries. An exhaustive review of the subject is well beyond our current scope. Instead we focus on those methods that are applicable to both the fixed geometries of this chapter and the deforming domains of Chapter 6.

The finite element method has been used extensively by many researchers (see [16], [23] and [25], for instance). Grid generation methods are now mature enough that most irregular geometries can be discretized quite easily and extremely accurate results can be produced using a combination of locally refined meshing and/or high order elements. The use of unstructured grids, however, can make the resulting discrete systems difficult and expensive to solve. In addition, if the geometry of the domain is allowed to deform with time, the mesh can become skewed and frequent regridding is required. To minimize this problem, some researchers have tried to construct boundary conforming meshes by cutting and/or deforming the cells of an underlying regular grid. The idea is to embed the irregular boundary in a larger regular mesh. The cells of the regular grid which are intersected by the irregular boundary (i.e., that contain both solid and liquid) are cut or distorted so that the embedded surface lies only along the edges of the cell (i.e., the cells exclusively contain solid or liquid). The modified cells typically require special treatment when the governing equations are discretized. Examples of this approach can be found in [71] and [80]. Other researchers have started with regular grids but accounted for the irregular geometry in a fundamentally different way. The essential idea is to compute a flow everywhere in the computational domain (which is regular) and to force the fluid in such a way that flow obeys the required boundary conditions on the boundaries of embedded irregular domain. Initial implementations of this approach accounted for the irregular boundary by adjusting the pressure in the mixed cells (cells containing both liquid and solid) so that the liquid did not penetrate the solid surface. The method has been applied to fixed (see [73]) and moving (see [74]) domains but is not capable of enforcing boundary conditions on the tangential velocity. An alternate approach is to introduce the normal and tangential forcing on the embedded domain as additional unknowns. The boundary forcing is coupled into the momentum equations through the use of a singular body force of the form (5.1). The delta function singularity is either discretized directly using smoothed delta functions (see [22]) or through the modification of the stencils intersected by the embedded boundary (as in this work or, for Stokes flow, [39]). The determination of the forcing has proved difficult in the past (see [22]) because a degeneracy in the pressure field was not recognized. We will address this issue below. Finally, some researchers have employed approximate methods of modeling the irregular boundaries. The two most popular are the use of a discontinuous viscosity that is taken to be very large in the solid region (see [57]) and the introduction of a porous media style friction term that is zero in the fluid and very large in the solid (see [56]). The accuracy associated with these approaches tends to be quite poor. For instance, in the porous media model, the momentum equation is given by

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \boldsymbol{\nabla} \mathbf{u} = -\boldsymbol{\nabla} p + D\nabla^2 \mathbf{u} - C\mathbf{u}, \tag{5.2}$$

where

$$C = \begin{cases} 0 & \text{in the liquid} \\ \varepsilon^{-2} & \text{in the solid} \end{cases} . \tag{5.3}$$

It is noted in [56] that the solution of the above (along with continuity) will have an error no smaller than $O\left(\varepsilon^{1/2}\right)$. The value of ε is non-physical so it is possible to take $\varepsilon \ll 1$ but, in practice, these methods have trouble with ill-conditioning if they select parameter values which are too extreme (see [57]).

5.3 Computational mesh

In this work we have used a staggered mesh scheme (see [24]) which, in the absence of an interface, allows the discrete version of the continuity equation to be satisfied exactly. A typical computational cell in our mesh is given in Fig. 5.1.

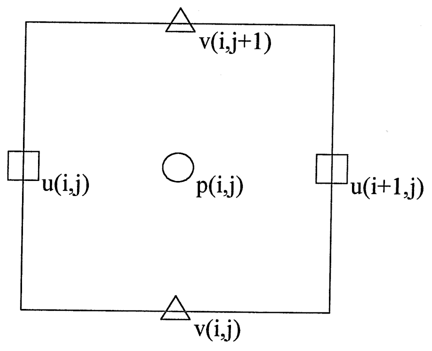

Figure 5.1: Pressure cell with relative indices for all variables

The velocity and pressure nodes in a staggered grid are placed so that there is a one-to-one identification between the pressure and the flow divergence in a computational cell. This one-to-one mapping is lost on a non-staggered grid, and it is not possible to exactly satisfy conservation of mass for second order accurate non-staggered schemes (see [62]). The staggered placement of the unknown variables does introduce some complexity over a discretization on a uniform grid. This is particularly true when singular forcing is allowed to act on the interface. We have found that mass conservation is important in the fully coupled solidification with convection problem, however, so a conservative scheme is called for.

5.4 Discretization without interfacial forcing

In this section we present a discretization of the Navier-Stokes equations that is applicable to a pure fluid with no forcing acting on an immersed interface. Although we do not intend to include the interface forcing yet, it is important that this incorporation be possible. As mentioned above, the interface forcing is only felt through the corrections that are made to stencils intersected by the interface. Ultimately, we need to determine normal and tangential forces that must be applied in order to force the fluid to obey the no-slip condition at the new position of the interface. This requires that at least some of the derivative terms in the system be treated implicitly so that the new forcing influences the flow and acts at the proper location. This requirement has a significant impact on the scheme that we develop.

In our presentation below, we will only discretize the time derivatives explicitly. Clearly the spatial derivatives must also be discretized but we have not explicitly included this. At least at this stage, including all the details in the development only complicates the presentation without providing additional insight. The stencils required for all spatial derivatives can be found in Appendix B.

Our scheme is second order accurate and utilizes a staggered mesh to simulate incompressible fluid flow. Although the use of a staggered mesh allows the discrete incompressibility constraint to be satisfied exactly, solving the resulting system can be expensive. One approach (see [14]) is to use an explicit forward Euler time stepping method to determine an intermediate velocity,

$$\hat{\mathbf{u}} = \mathbf{u}^{n+1} + \Delta t \left(\boldsymbol{\nabla} p^{n+1} \right) = \mathbf{u}^n + \Delta t \left[D \nabla^2 \mathbf{u}^n - \mathbf{u}^n \cdot \boldsymbol{\nabla} \mathbf{u}^n + \mathbf{b}^n \right], \tag{5.4}$$

that, in general, will not satisfy the incompressibility constraint. Note that we have included an arbitrary body force term, $\mathbf{b}$, in the above that will later be replaced by the buoyancy forcing. The intermediate velocity is then used to determine the pressure,

$$\nabla^2 p^{n+1} = \frac{1}{\Delta t}\boldsymbol{\nabla}\cdot\hat{\mathbf{u}}, \tag{5.5}$$

and the physical velocity at the new time

$$\mathbf{u}^{n+1} = \hat{\mathbf{u}} - \Delta t\left(\boldsymbol{\nabla} p^{n+1}\right). \tag{5.6}$$

Unfortunately, we can not use the above in this work because none of the spatial derivative terms are evaluated at the $n+1$ time level. Although we need a method that treats at least some of the derivatives implicitly, this requirement significantly complicates the problem. If the convective terms are treated implicitly, we are confronted with a fully non-linear system that is difficult and expensive to invert. If the viscous terms are treated implicitly, the simple de-coupling of the velocity from the pressure seen above is no longer possible. In the absence of an interface, it is possible to treat the viscous terms implicitly (see [32]). Unfortunately, that method requires that the pressure be replaced by a non-physical variable which complicates interface correction. We have chosen a different approach that maintains both the linearity of the system and the simple de-coupling of the velocity and pressure.

The scheme utilizes a Runge-Kutta (see [28]) time stepping method. First we calculate an explicit forward Euler prediction to the physical velocity

$$\mathbf{u}^* = \mathbf{u}^n + \Delta t\left[D\nabla^2\mathbf{u}^n - \mathbf{u}^n\cdot\boldsymbol{\nabla}\mathbf{u}^n - \boldsymbol{\nabla} p^n + \mathbf{b}^n\right]. \tag{5.7}$$

Note that because the incompressibility constraint is not utilized to calculate $\mathbf{u}^*$ it, in general, will not satisfy conservation of mass exactly. Since it uses an incompressible flow to calculate the time derivative, however, it will be an accurate approximation to the physical velocity ($\mathbf{u}^* = \mathbf{u}^{n+1} + O\left(\Delta t^2\right)$). Next this estimate is used to compute an intermediate velocity,

$$\hat{\mathbf{u}} \;=\; \mathbf{u}^{n+1} + \Delta t\left(\boldsymbol{\nabla} p^{n+1}\right), \tag{5.8}$$

$$\hat{\mathbf{u}} = \mathbf{u}^n + \Delta t\left\{\alpha\left[D\nabla^2\mathbf{u}^*\right] + (1-\alpha)\left[D\nabla^2\mathbf{u}^n\right] - \mathbf{u}^n\cdot\boldsymbol{\nabla}\mathbf{u}^n + \mathbf{b}^{n+1}\right\}, \quad (5.9)$$

which, in turn, is used to determine the pressure,

$$\nabla^2 p^{n+1} = \frac{1}{\Delta t}\boldsymbol{\nabla}\cdot\hat{\mathbf{u}}, \quad (5.10)$$

and the physical velocity at the new time

$$\mathbf{u}^{n+1} = \hat{\mathbf{u}} - \Delta t\left(\boldsymbol{\nabla}p^{n+1}\right). \quad (5.11)$$

Note that the above is a family of time marching schemes with ($\alpha \in [0,1]$). Some comments are in order concerning the treatment of the different terms on the right-hand side of (5.9). Starting from right to left, we first consider the body forcing term **b**. In some of our later simulations we will find that the dominant balance in the Navier-Stokes equations is between the pressure gradient and body forcing (buoyancy) terms. It is important, therefore, that these terms always be evaluated at the same time level. This is the motivation for evaluating the body forcing at the old time, $\mathbf{b}^n$, in (5.7) and at the new time, $\mathbf{b}^{n+1}$, in (5.9). The convection terms must be evaluated at t_n to avoid introducing non-linear dependence of the solution on the interface forcing. Although this touches on the subject of the next few sections, we will address it here briefly. Consider the convection terms in the horizontal direction, which are given by

$$u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y}. \quad (5.12)$$

The value of the vertical velocity, v, is not known at the u nodes so it must be interpolated using

$$v = \frac{1}{4}\left[v(i-1,j) + v(i,j) + v(i-1,j+1) + v(i,j+1)\right]. \quad (5.13)$$

As we shall discuss shortly, if this interpolation stencil is intersected by the interface, a correction must be added to the above so that the value at the u node is accurately computed. If we wish to use (u^*, v^*) to calculate the convection terms, then the corrections applied to the above derivatives and interpolation will depend on the normal and tangential forcing on the interface at t_{n+1}, $(f_\tau, f_n)^{n+1}$. Additive corrections will be required for both the interpolated value of v^* and the vertical derivative of u^* which will introduce a non-linear dependence of the flow on the interface forcing $(f_\tau, f_n)^{n+1}$. It is significantly simpler to determine $(f_\tau, f_n)^{n+1}$ when it has a linear relationship with the flow field. In this work, therefore, we only evaluate the convection terms at t_n to avoid confronting the non-linearity.

Finally we consider the treatment of the viscous terms. For the interface forcing at t_{n+1} to influence the flow field, at least some of the derivatives must be evaluated at t_{n+1} and the viscous terms are the only ones remaining. This implies that we must solve the above using $\alpha > 0$. In fact, a standard linear stability analysis (see [28]) of the Runge-Kutta scheme reveals that for $\alpha = \frac{1}{2}$ (trapezoidal rule) the stability restriction on the time step is identical to the forward Euler method,

$$\Delta t \leq \frac{\Delta x^2}{4D}. \tag{5.14}$$

When $\alpha = 1$ (Backward Euler) the time step can be only half the size of the $\alpha = \frac{1}{2}$ case so we compute using $\alpha = \frac{1}{2}$. Thus, we see that although it is awkward to use a different time stepping method on each term in (5.9), it is necessary so that all the demands placed the scheme can be met.

5.5 Discretization with interfacial forcing

5.5.1 Spatial corrections

Correcting the finite difference approximations for the Navier-Stokes equations follows the same procedure we outlined in Chapter 3. For example, to compute the momentum in the horizontal direction, we need an approximation for the vertical

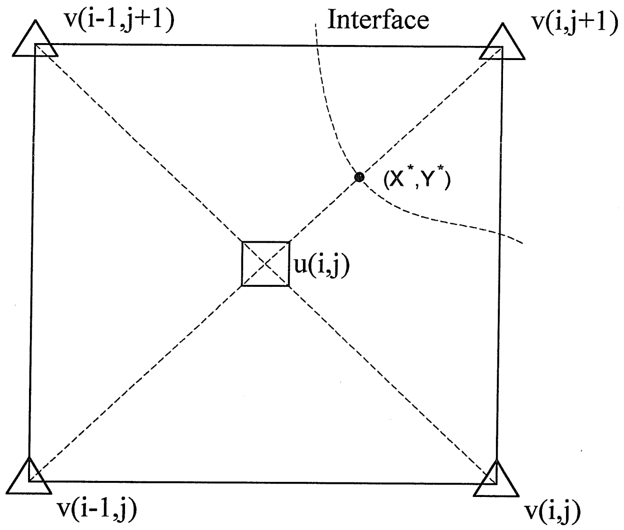

Figure 5.2: Node correction on a staggered grid

velocity at the u-nodes. In the absence of interface intersections, this is given by

$$v = \frac{1}{4}\left[v\left(i-1,j\right) + v\left(i,j\right) + v\left(i-1,j+1\right) + v\left(i,j+1\right)\right]. \tag{5.15}$$

Now, suppose that the interface intersects the diagonal line connecting the $u\left(i,j\right)$ node and the $v\left(i,j+1\right)$ node, as shown in Fig. 5.2. If the position of $v\left(i,j+1\right)$ relative to the interface intersection point $\left(X^*,Y^*\right)$ is $\left(\bar{x},\bar{y}\right)$, then the correction associated with the intersection is

$$C = \bar{x}\left[v_x\right] + \bar{y}\left[v_y\right] + \frac{1}{2}\left(\bar{x}^2\left[v_{xx}\right] + 2\bar{x}\bar{y}\left[v_{xy}\right] + \bar{y}^2\left[v_{yy}\right]\right), \tag{5.16}$$

where $\left[v_x\right]$ is the jump in the x-derivative of v, $\left[v_{xy}\right]$ is the jump in the second derivative of v with respect to x and y and so on. The corrected value for v at the $u\left(i,j\right)$ node is then given by

$$v = \frac{1}{4}\left[v\left(i-1,j\right) + v\left(i,j\right) + v\left(i-1,j+1\right) + v\left(i,j+1\right) - C\right], \tag{5.17}$$

if $u\left(i,j\right)$ is on the minus side of the interface (i.e., the normal points away from the

u node) and by

$$v = \frac{1}{4}\left[v\left(i-1,j\right) + v\left(i,j\right) + v\left(i-1,j+1\right) + v\left(i,j+1\right) + C\right], \tag{5.18}$$

if $u\left(i,j\right)$ is on the plus side of the interface (i.e., the normal points towards the u node). All of the other discretizations also follow this standard approach with one exception, which we now address.

The correction of the intermediate velocity, $\hat{\mathbf{u}} = \left(\hat{u}, \hat{v}\right)$, is complicated by the fact that it is not a physical quantity. The jumps across the interface induced by the forcing are known for the pressure, p, and the physical velocities u and v (see Appendix C). The intermediate velocity is purely a numerical construct, however, so we can only determine its jumps by relating them to jumps in the physical variables. Recall that the intermediate velocity is defined as

$$\hat{\mathbf{u}} = \mathbf{u}^{n+1} + \Delta t\left(\boldsymbol{\nabla} p^{n+1}\right). \tag{5.19}$$

Thus, by definition, the jumps in its horizontal component are given by

$$\begin{aligned}
\left[\hat{u}\right] &= \Delta t\left[p_x\right], && (5.20)\\
\left[\hat{u}_x\right] &= \left[u_x\right] + \Delta t\left[p_{xx}\right], && (5.21)\\
\left[\hat{u}_{xx}\right] &= \left[u_{xx}\right] + \Delta t\left[p_{xxx}\right] = \left[u_{xx}\right] + O\left(\Delta x^2\right), && (5.22)
\end{aligned}$$

and the jumps in its vertical component are given by

$$\begin{aligned}
\left[\hat{v}\right] &= \Delta t\left[p_y\right], && (5.23)\\
\left[\hat{v}_y\right] &= \left[v_y\right] + \Delta t\left[p_{yy}\right], && (5.24)\\
\left[\hat{v}_{yy}\right] &= \left[v_{yy}\right] + \Delta t\left[p_{yyy}\right] = \left[v_{yy}\right] + O\left(\Delta x^2\right). && (5.25)
\end{aligned}$$

The jumps in the derivatives of pressure are all multiplied by $\Delta t = O\left(\Delta x^2\right)$ so they make an almost negligible contribution to the quantities above. It is important to include the pressure jumps through the second order derivatives, however, to satisfy continuity.

The staggered grid discretization of the Navier-Stokes equations is able to conserve mass accurately because the equation for the pressure is derived by direct application of the discrete continuity equation. In the scalar form, we start with the definition of the intermediate velocity

$$\hat{u} = u^{n+1} + \Delta t \left(\frac{\partial p}{\partial x}\right), \tag{5.26}$$

$$\hat{v} = v^{n+1} + \Delta t \left(\frac{\partial p}{\partial y}\right), \tag{5.27}$$

and then apply the discrete continuity equation to find

$$\frac{\partial^2 p}{\partial x^2} + \frac{\partial^2 p}{\partial y^2} = \frac{1}{\Delta t}\left(\frac{\partial \hat{u}}{\partial x} + \frac{\partial \hat{v}}{\partial y}\right). \tag{5.28}$$

In order for continuity to be satisfied, the discrete approximation to the second derivative must be identical whether it is calculated directly using

$$\frac{\partial^2 p}{\partial x^2} = \frac{1}{\Delta x^2}\left[p\,(i+1,j) + p\,(i-1,j) - 2p\,(i,j)\right] \tag{5.29}$$

or indirectly using

$$\frac{\partial}{\partial x}\left(\frac{\partial p}{\partial x}\right) = \frac{1}{\Delta x}\left[\frac{\partial p}{\partial x}(i+1,j) - \frac{\partial p}{\partial x}(i,j)\right] \tag{5.30}$$

where

$$\frac{\partial p}{\partial x}(i+1,j) = \frac{1}{\Delta x}\left[p\,(i+1,j) - p\,(i,j)\right], \tag{5.31}$$

and

$$\frac{\partial p}{\partial x}(i,j) = \frac{1}{\Delta x}\left[p\,(i,j) - p\,(i-1,j)\right]. \tag{5.32}$$

If the equivalence between these two representations is lost, then solving the Poisson equation for the pressure is no longer equivalent to satisfying continuity and mass conservation will be poor. Therefore, even though the contribution of the pressure jumps in (5.20)-(5.25) is small, if these quantities are not included, the corrections associated with (5.30) will be different from those associated with (5.29) and the equivalence will be lost. When the intermediate velocity corrections are calculated using (5.20)-(5.25), continuity can be accurately satisfied, even in the presence of singular interface forcing.

5.6 Identification and removal of singularities

5.6.1 Determination of the flow field

The pressure in the incompressible Navier-Stokes equations is only determined up to an additive constant. This degeneracy is also inherited by the discrete equations. Thus, the Poisson equation for the pressure is singular with a single null vector that has all its entries equal to one. By the Fredholm Alternative (see [65]) we know that this system only has a solution if the inner product of this null vector and the forcing is zero. It follows from the form of the forcing (see Appendix C) that this inner product is given by

$$\begin{aligned}
\mathbf{z}^T\mathbf{f}_p &= \sum_{i=0}^{N_x-1}\sum_{j=0}^{N_y} f_p(i,j) && (5.33)\\
&= \frac{-1}{\Delta t}\left\{\sum_{j=0}^{N_y}\left[\frac{\hat{u}(N_x,j)-\hat{u}(0,j)}{\Delta x}\right]+\sum_{i=0}^{N_x-1}\left[\frac{v(i,N_y+1)-v(i,0)}{\Delta x}\right]\right\} && (5.34)
\end{aligned}$$

which, roughly speaking, is a discrete approximation to the integral of the normal velocity around the computational domain. For our geometry, the first term on the right-hand side vanishes due to periodicity in the horizontal direction and the last term vanishes because the velocities at the lower and upper boundaries are specified to be zero. Thus the discrete system can have an infinite number of solutions corresponding to adding an arbitrary constant to the pressure. From a practical stand point, it is desirable to remove this degeneracy so that we can unambiguously determine the pressure. In this work the pressure Poisson equation is solved using a "fast solver" which utilizes the discrete fast Fourier transform (see [68]). Using this approach, the solution is decomposed into Fourier modes in the horizontal direction that are decoupled and whose vertical dependence is easily determined. That is, the transformed Poisson equation becomes

$$\left(\frac{d^2}{dy^2} + \partial_{xx}(n)\right)\bar{p} = \bar{f} \tag{5.35}$$

where $\bar{p}$, $\bar{f}$ and $\partial_{xx}(n)$ are the discrete Fourier transforms of the pressure, system forcing and discrete second order derivative operator, respectively, in the horizontal direction. The discretization in the vertical direction is neglected above for simplicity. The above equation is non-singular for all the Fourier modes $n = 1$ to $N_x - 1$, but singular for the $n = 0$ mode which corresponds to modes with no variation in the horizontal direction. The exact discrete equations for the $n = 0$ case are, at the bottom of the domain,

$$\frac{1}{(\Delta x)^2}\left[\bar{p}(1;0) - \bar{p}(0;0)\right] = \bar{f}(0;0)\,, \tag{5.36}$$

in the interior, $1 \leq j < N_y$

$$\frac{1}{(\Delta x)^2}\left[\bar{p}(j+1;0) - 2\bar{p}(j;0) + \bar{p}(j-1;0)\right] = \bar{f}(j;0) \tag{5.37}$$

and at the top of the domain

$$\frac{1}{(\Delta x)^2}\left[\bar{p}(N_y-1;0) - \bar{p}(N_y;0)\right] = \bar{f}(N_y;0)\,. \tag{5.38}$$

This system only determines the pressure up to a constant. We can make the system non-singular without any loss of generality by simply replacing the equation at the top of the domain by

$$\bar{p}(N_y;0) = 0 \tag{5.39}$$

which specifies a value for the additive constant. All the pressures modes can now be unambiguously determined and transformed back to physical space.

In the absence of interface forcing, the above procedure yields an accurate solution to the Navier-Stokes equations that satisfies the discrete continuity equation to machine precision. When forcing acts on the interface, the derivatives of the flow are no longer smooth and it is necessary to add corrections to all stencils intersected by the interface. These corrections destroy the special structure of the pressure Poisson forcing and the Fredholm Alternative is no longer satisfied. This difficulty was also encountered by LeVeque (see [39]) when solving the Stokes equations on a regular grid. Although it is no longer possible to obtain an exact solution for the pressure, it is still possible to obtain a solution in the least squares sense (see [68]) by using a perturbed forcing

$$\hat{f}_p(i,j) = f_p(i,j) - \frac{1}{N_x(N_y+1)} \sum_{a=0}^{N_x-1} \sum_{b=0}^{N_y} f_p(a,b) \tag{5.40}$$

that does satisfy the Fredholm Alternative. Note that the sum on the right-hand side is non-zero because we only correct the derivatives up to second order yielding truncation errors of $O(\Delta x)$ at stencils intersected by the interface. There are $O(\Delta x^{-1})$ stencils intersected by the interface implying that the sum of f_p over the grid will be $O(1)$. Therefore, the difference between f_p and $\hat{f}_p$ is $O(\Delta x^2)$ which does not change the order of the accuracy of the solution. Because the pressure Poisson equation can not be solved exactly when interface forcing is present, it is not possible to satisfy the continuity equation to machine precision in this case. The divergence of the flow field instead is equal to a, typically very small, spatially uniform constant that converges to zero as the computational mesh is refined.

5.6.2 Determination of the interface forcing

We are using singular body forcing to model the effect of a solid boundary at the interface position. This means that the forcing must be determined such that the velocity satisfies a no-slip condition at the interface. A consequence of this approach is that the domain is divided into two completely independent regions that separately satisfy the Navier-Stokes equations. Typically only the flow in one of these regions is physically relevant but both must be determined. Two complications arise when this is implemented in practice. The first difficulty we encounter is that the presence of two independent flow regions implies the existence of two independent arbitrary pressure constants. We determine both flows on a single grid so the procedure described above eliminates one of the pressure constants, but that still leaves one unspecified. Second, the specification of the fluid velocity on the interface is non-trivial. The problem is conservation of mass. Because our simulations are performed on a periodic domain with zero velocity specified on the upper and lower edges, the exact mass conservation constraint on the fluid velocity at the interface, $\mathbf{U}(q,t)$, is

$$\int_0^1 (\mathbf{U}\cdot\mathbf{n})\left(\frac{ds}{dq}\right) dq = 0. \tag{5.41}$$

Unfortunately, we can only determine the velocity on the interface in our simulations by interpolating it off the computational grid. The process of interpolation introduces errors. Even if we interpolate the interface velocity off of a divergence free flow field, the resulting values would not satisfy (5.41). Instead, the effective constraint on the interface velocity is

$$\int_0^1 (\mathbf{U}\cdot\mathbf{n})\left(\frac{ds}{dq}\right) dq = A \tag{5.42}$$

where the value of A depends on the interpolation scheme and the relative position of the interface control points and grid nodes. We use a second order accurate interpolation scheme so, in our simulations, $A = O\left(\Delta x^2\right)$. Unfortunately this implies that, in general, the numerical method can not produce a flow field that satisfies $\mathbf{U}$ on the interface, even though analytically a solution exists. It is possible, however, for the numerical method to produce a slightly perturbed flow field that has a velocity of $\mathbf{U} + A\mathbf{n}$ on the interface. Put another way, the numerical method is only capable of specifying the fluid velocity on the interface up to a constant. This constant is not arbitrary. As we shall see, it is set automatically by scheme. Before we can do this, however, it is necessary to discuss the system that must be solved to determine the interface forcing.

The singular interface forcing modifies the flow field through the application of the spatial corrections discussed above. Although the dependence is convoluted, careful inspection of the form that these corrections take reveals that there is a linear relationship between the interface forcing (f_τ, f_n) and the flow field. This implies that there is a linear relationship between the interface forcing and the velocity on the interface (u_τ, u_n). The velocity on the interface is interpolated off the grid using the averaged Taylor series approach discussed earlier in Chapter 4. The dependence of (u_τ, u_n) on (f_τ, f_n) is far too complicated to determine analytically but, because we know that the system is linear, we can apply one of the many iterative solvers which only require the system residual. In this work we use the preconditioned BiConjugate Gradient Stabilized method (see [5]).

Before we define the precise system residual, it is instructive to discuss the most obvious, but not quite correct, definition of the system residual. At each of the interface control points, $0 \leq i < N_c$, we interpolate the flow velocity (u, v) off the staggered grid and use these to calculate the tangential and normal velocities at the interface control points

$$u_\tau(i) = \tau_x(i)\, u(i) + \tau_y(i)\, v(i)\,, \tag{5.43}$$

$$u_n(i) = n_x(i)\, u(i) + n_y(i)\, v(i)\,. \tag{5.44}$$

The residual associated with the tangential forcing at the control points is calculated using

$$r(i) = u_\tau(i) - U_\tau(i)\,, \tag{5.45}$$

where $U_\tau(i)$ is the desired tangential velocity on the interface at the i^{th} control point. The residual associated with the normal forcing at the control points is calculated using

$$r(N_c + i) = u_n(i) - U_n(i)\,, \tag{5.46}$$

where $U_n(i)$ is the desired normal velocity on the interface at the i^{th} control point. As we mentioned above, however, it is not possible to completely specify the normal velocity on the interface. We must leave a constant that is implicitly determined by conservation of mass. Using

$$r(N_c + i) = [u_n(i) - U_n(i)] - [u_n(i+1) - U_n(i+1)]\,, \tag{5.47}$$

to compute the residual for $0 \leq i < N_c - 1$ forces the difference between the desired normal velocity and computed normal velocity to be a constant, as we require. The above leaves the residual at $i = N_c - 1$ unspecified but this last equation is needed to remove the arbitrary additive constant associated with pressure. In Appendix C we found that the jump in the pressure is given by the normal force acting at the interface

$$[p] = f_n. \tag{5.48}$$

Any constant value added to the normal forcing at all the control points simply acts to change the jump in the pressure. Because the flows on each side of the interface are independent, however, a uniform pressure jump has no dynamic consequence and simply serves to change the additive pressure constant. This null vector can be eliminated by setting the average value of the pressure jump to $[p]_{avg}$, using

$$r\left(2N_c - 1\right) = \frac{1}{N_c} \sum_{i=0}^{N_c - 1} f_n\left(i\right) - [p]_{avg}\,. \tag{5.49}$$

The value of $[p]_{avg}$ simply specifies a relationship between the additive pressure constants in each independent flow region. It is arbitrary and can be set to zero without any loss of generality. We will sometimes use a non-zero value of $[p]_{avg}$ to aid with the visualization of the pressure field. Altogether then, the residual for the tangential forcing at the control points, $0 \leq i < N_c$, is calculated using

$$r\left(i\right) = u_\tau\left(i\right) - U_\tau\left(i\right), \tag{5.50}$$

and the residual for the normal forcing at the control points, $0 \leq i < N_c - 1$, is calculated using

$$r\left(N_c + i\right) = \left[u_n\left(i\right) - U_n\left(i\right)\right] - \left[u_n\left(i+1\right) - U_n\left(i+1\right)\right], \tag{5.51}$$

and at the final control point using

$$r\left(2N_c - 1\right) = \frac{1}{N_c} \sum_{i=0}^{N_c-1} f_n\left(i\right) - \left[p\right]_{avg}. \tag{5.52}$$

Using these residuals, we have a well defined, non-singular system relating the interface forcing to the interface velocity. As far as we know, this is the first time that the difficulties associated with specifying an embedded boundary velocity in a fixed grid method have been identified and resolved. It is possible that a similar approach could be useful in immersed boundary methods that typically have substantial problems determining the appropriate interface forcing (see [22]).

It is possible to solve the above system for the interface forcing using an unconditioned iterative method such as the BiConjugate Gradient Stabilized method. It is quite easy to calculate a preconditioner, however, and we have found that it significantly reduces the number of iterations that must be performed during each time step. The preconditioner that we use is an exact inverse of the system at a previous time step. Introducing some notation, let

$$\mathbf{x}^T = \left[\begin{array}{cccccc} f_\tau\left(0\right) & \cdots & f_\tau\left(N_c-1\right) & f_n\left(0\right) & \cdots & f_n\left(N_c-1\right) \end{array}\right] \tag{5.53}$$

and $\mathbf{A}$ be the system we want to solve so that the residual is given by

$$\mathbf{r} = \mathbf{A}\mathbf{x} - \mathbf{b}. \tag{5.54}$$

We can determine $\mathbf{A}$ by $2N_c + 1$ system evaluations. The vector $\mathbf{b}$ is determine by setting $\mathbf{x} = \mathbf{0}$ and calculating the residual. The i^{th} column of $\mathbf{A}$ is determined by setting $\mathbf{x} = \mathbf{e}_i$, the i^{th} unit vector, evaluating the residual and adding $\mathbf{b}$. Once the system $\mathbf{A}$ is determined, we can store it in factored form and use it as a preconditioner. For fixed interface problems, this preconditioner exactly inverts the system for all time steps. When the interface moves, the system changes slowly over time, but this preconditioner is still quite successful.

5.7 Solution procedure

Now that all the required steps in the algorithm have been discussed, we can fully describe the solution algorithm. The basic idea is that each iteration for the interface forcing requires the solution of the Navier-Stokes equations. Thus each BiConjugate Gradient Stabilized iteration for the interface forcing performs the following steps:

1. Determine $\mathbf{u}^*$ applying corrections evaluated with $(f_\tau, f_n)^n$.
2. Determine $\hat{\mathbf{u}}$ without corrections.
3. Apply corrections to $\hat{\mathbf{u}}$ using $(f_\tau, f_n)^n$ and $(f_\tau, f_n)^{n+1}$ where appropriate.
4. Determine p^{n+1} applying spatial and temporal corrections where required.
5. Determine $\mathbf{u}^{n+1}$ applying spatial corrections for the pressure gradients.
6. Interpolate the interface velocity off the grid using the averaged Taylor series method.
7. Calculate the residual associated with the current interface forcing.

In practice, the first two steps are precomputed so that the calculation of the residual is more efficient. The calculation of the preconditioner, described above, is generic and done automatically by the BiConjugate Gradient Stabilized code. For problems with a fixed interface, this process converges to the exact solution in a single iteration. When the interface is allowed to move it is common for two or three iterations to be performed before the infinity norm of the residual is sufficiently reduced.

5.8 Calculation of the stream function

The stream function is useful for the visualization of two-dimensional flows. It is related to the velocity by (see [77])

$$u = +\frac{\partial \Psi}{\partial y}, \tag{5.55}$$

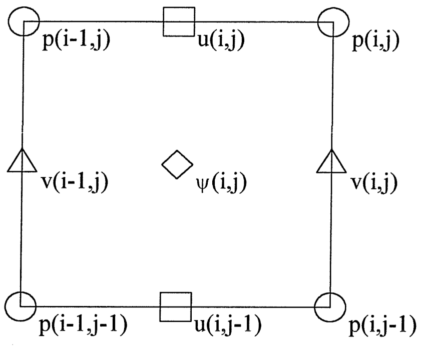

Figure 5.3: Location of the stream function nodes

$$v = -\frac{\partial \Psi}{\partial x}, \tag{5.56}$$

which implies

$$\nabla^2 \Psi = -\omega, \tag{5.57}$$

where the vorticity ω is defined as

$$\omega = \frac{\partial v}{\partial x} - \frac{\partial u}{\partial y}. \tag{5.58}$$

It is convenient to calculate the stream function at nodes that are located at the corners of the pressure cell (see Fig. 5.3). Exploiting the no-slip condition on the boundaries of the computational domain, we set the stream function to

$$\Psi\left(x, y = y_L\right) = 0 \tag{5.59}$$

on the lower boundary and

$$\frac{\partial \Psi}{\partial y}\left(x, y = y_U\right) = 0 \tag{5.60}$$

on the upper boundary. The value of the stream function on the interface, which should be a constant, is determined through the solution of the above system. Recalling that the velocity is continuous across the interface, we see from (5.55) and (5.56) that the stream function and its first derivatives will be continuous even when singular forcing is applied. Using the velocity jumps calculated in Appendix C, however, it is easy to find that the jump in the vorticity across the interface is

$$[\omega] = \frac{f_\tau}{D}. \tag{5.61}$$

This implies that the stream function satisfies a Poisson equation with a discontinuous forcing which, we know from Chapter 3, yields a solution with discontinuous second derivatives. To be exact, we have

$$[\Psi] = 0, \tag{5.62}$$

$$[\Psi_x] = 0, \tag{5.63}$$

$$[\Psi_y] = 0, \tag{5.64}$$

$$[\Psi_{xx}] = -n_x^2 [\omega], \tag{5.65}$$

$$[\Psi_{xy}] = -n_x n_y [\omega] \tag{5.66}$$

and

$$[\Psi_{yy}] = -n_y^2 [\omega]. \tag{5.67}$$

Using these jumps to correct the Laplacian and standard methods to correct the velocity derivatives, the stream function can be easily determined. In general the value of the stream function on the interface will not be exactly constant but does converge to a constant at a second order rate.

5.9 Results

The validation of a Navier-Stokes flow code is complicated by the fact that exact solutions to these equations are rare. The solutions that do exist are typically limited to straight stream lines in a Cartesian or cylindrical geometry. Unfortunately, these solutions essentially reduce to solving a heat equation for the velocity and do not fully test the conservation of mass properties of a numerical scheme. To circumvent this difficulty, we use three validation flows. Two of these flows have essentially straight stream lines and two are general but have no known exact solution. For the final solution, we validate the code by verifying that the velocities and pressure are converging to a solution at the expected second order accurate rate. This can be done using a clever trick (see [39]). Compute three numerical solutions, U_1, U_2 and U_4, where the grid associated with each solution is twice as fine as the previous grid. We then determine the infinity norm of the difference between the solutions evaluated at the grid nodes of the coarsest grid and compute

$$\phi = \frac{\|U_1 - U_4\|_\infty}{\|U_2 - U_4\|_\infty} \approx \frac{C(\Delta x)^q - C\left(\frac{\Delta x}{4}\right)^q}{C\left(\frac{\Delta x}{2}\right)^q - C\left(\frac{\Delta x}{4}\right)^q} = \frac{4^q - 1}{2^q - 1}, \tag{5.68}$$

where q is the order of the numerical method. Solving the above for q, we find

$$q = \log_2(\phi - 1). \tag{5.69}$$

There is a slight complication for our particular problem due to the position of the nodes on the staggered grid. To use the above, the infinity norm must be calculated at nodes that all lie at the same spatial location. When a staggered grid is refined, the nodes change position and, for instance, none of the pressure nodes on the U_2 grid will coincide with the pressure nodes on the U_1 grid. Since we only expect second order accuracy, however, this problem is easily resolved. Once the solutions are determined on each grid, they are interpolated to the stream function nodes, which do coincide as the mesh is refined. This is done using

$$u_s(i,j) = \frac{1}{2}\left[u(i,j) + u(i,j-1)\right], \tag{5.70}$$

$$v_s(i,j) = \frac{1}{2}\left[v(i,j) + v(i-1,j)\right] \tag{5.71}$$

and

$$p_s(i,j) = \frac{1}{4}\left[p(i,j) + p(i,j-1) + p(i-1,j) + p(i-1,j-1)\right] \tag{5.72}$$

which, of course, must all be corrected, using the standard method, if their interpolating stencil is intersected by the interface. We then use the values of U_{1s}, U_{2s} and U_{4s} to determine the order of the method.

5.9.1 Impulsively started flat plate

There is an exact solution to the Navier-Stokes equations for the problem specified by the initial conditions

$$u\left(y, t=0\right) = 0, \tag{5.73}$$

$$v\left(y, t=0\right) = 0, \tag{5.74}$$

and boundary conditions ($y_1 < y_2$):

$$u\left(y=y_1, t \geq 0\right) = U_1, \tag{5.75}$$

$$v\left(y=y_1, t \geq 0\right) = 0, \tag{5.76}$$

$$u\left(y=y_2, t \geq 0\right) = U_2, \tag{5.77}$$

$$v\left(y=y_2, t \geq 0\right) = 0, \tag{5.78}$$

given by

$$p\left(y, t\right) = \text{constant}, \tag{5.79}$$

$$v\left(y, t\right) = 0, \tag{5.80}$$

$$\begin{aligned} u\left(y, t\right) = {} & U_1\left(\frac{y-y_2}{y_1-y_2}\right) + U_2\left(\frac{y-y_1}{y_2-y_1}\right) \\ & - \sum_{n=1}^{\infty}\left\{\frac{2\left[U_1-\left(-1\right)^n U_2\right]}{\pi n}\exp\left[\frac{-n^2\pi^2 Dt}{\left(y_2-y_1\right)^2}\right]\sin\left[\frac{n\pi\left(y-y_1\right)}{\left(y_2-y_1\right)}\right]\right\}. \end{aligned} \tag{5.81}$$

Note that the above has no dependence on x so it is a trivial example of an x-periodic flow. We will not solve the above problem exactly. Instead, on an x-periodic computational domain ($-\pi \leq x, y \leq \pi$) we solve the problem specified by the initial conditions

$$u\left(x,y,t=0\right) = 0, \tag{5.82}$$

$$v\left(x,y,t=0\right) = 0 \tag{5.83}$$

with no-slip along the $y = \pm\pi$ boundaries and

$$u\left(x = X\left(q\right), y = Y\left(q\right), t \geq 0\right) = 1 \tag{5.84}$$

$$v\left(x = X\left(q\right), y = Y\left(q\right), t \geq 0\right) = 0 \tag{5.85}$$

where $q \in [0,1]$ is the parametrization variable and the interface is a periodic cubic spline interpolation of

$$X\left(q\right) = -\pi + 2\pi q, \tag{5.86}$$

$$Y\left(q\right) = -1, \tag{5.87}$$

using N_c control points (q_i for $0 \leq i < N_c$) placed approximately a distance of $2\Delta x$ apart. In terms of the above analytic solution, this problem corresponds to

$$U_1 = 0, \quad y_1 = -\pi, \tag{5.88}$$

$$U_2 = 1, \quad y_2 = -1, \tag{5.89}$$

in the region below the interface, $-\pi \leq y \leq -1$, and

$$U_1 = 1, \quad y_1 = -1, \tag{5.90}$$

$$U_2 = 0, \quad y_2 = \pi, \tag{5.91}$$

in the region above the interface, $-1 \leq y \leq -\pi$. We have computed the solution using $\Delta t = 0.2\Delta x^2$ and $D = 1$, until $t = 0.2$ on three different grids with $N_x = N_y = 32$, 64 and 128. The horizontal velocity at $t = 0.2$ on the 32x32 grid is shown in Fig. 5.4. The difference between the exact solution and the computed solution, even on the

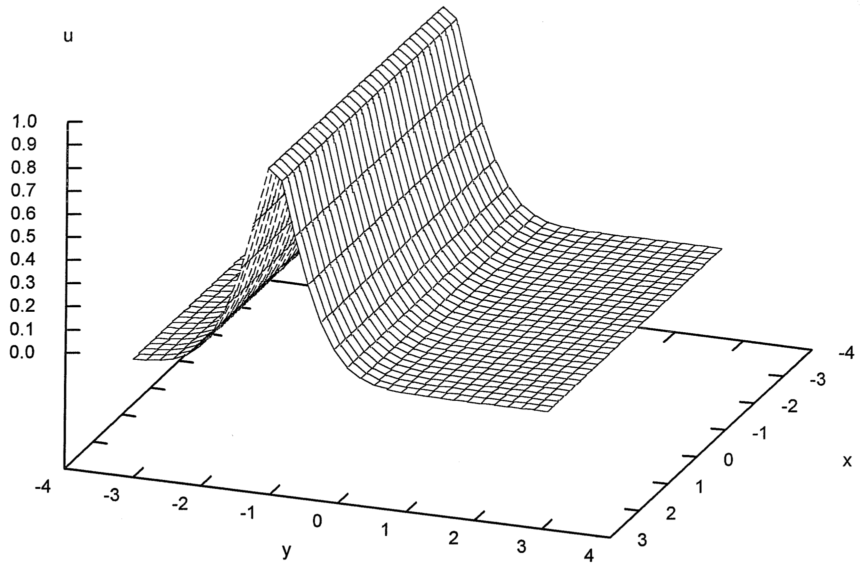

Figure 5.4: Horizontal velocity induced by an impulsively started flat plate at y=-1

$N_x = N_y$	$\|u_N - u\|_\infty$	Ratio
32	0.0120379	
64	0.0032118	3.74794
128	0.0009318	3.44703
256	0.0002114	4.40827

Table 5.1: Resolution study for impulsively started flat plate problem

relatively coarse grid depicted in Fig. 5.4, is imperceptible. We note that even though the velocity at the interface is $u = 1$, all of the computational nodes have velocities less than one. This is because the interface lies between the computational nodes on the grid. The position of the interface is also responsible for the appearance of a blunt region between the nodes adjacent to the interface. There is, in fact, no blunt region near the interface. The appearance of one in Fig. 5.4 is due to smoothness assumptions made by the plotting software. We have confirmed the accuracy of the computed flow field by comparing the numerical and exact horizontal velocities in Table 5.1, where the numerical solution is denoted by u_N and the exact solution is denoted by u. It is also possible to compute the exact stream function associated with

$N_x = N_y$	$\|\|\psi_N - \psi\|\|_\infty$	Ratio
32	0.00677286	
64	0.00193605	3.49829
128	0.00073599	2.63053
256	0.00013965	5.27010

Table 5.2: Resolution study for stream function associated with the impulsively started flat plate

$N_x = N_y$	$\|\|f_{\tau,N} - f_\tau\|\|_\infty$	Ratio
32	0.0148112	
64	0.00338142	4.38017
128	0.00071172	4.75105
256	0.00019931	3.57091

Table 5.3: Resolution study for the tangential forcing associated with the impulsively started flat plate

the flow by integrating the above series. The stream function, computed on a 32x32 grid, is shown in Fig. 5.6. We have verified the convergence of our computed stream function at several different resolutions in Table 5.2. The results indicate second order accuracy. Finally, for this problem we can compare the exact and the computed interface forcing which is required to accelerate the fluid. From Appendix C, we see that

$$\left[\frac{\partial u}{\partial y}\right] = -\frac{f_\tau}{D} \tag{5.92}$$

and the jump in the derivative can be computed from the known exact solution. The normal force for this problem is zero since pressure is not needed to maintain continuity. The tangential forcing is constant along the entire interface due to the one-dimensional nature of the problem. Thus, it is sufficient to plot the value of f_τ at $x = 0$ which we do for several different resolutions in Fig. 5.5. The exact tangential forcing required at $t = 0$ is infinite so, instead of plotting the entire time range we plot all solutions for $t \geq \Delta t_{32}$, where Δt_{32} is the time step associated with the 32x32 grid. Figure 5.5 indicates that the tangential forcing converges quite quickly to the exact interface forcing. Indeed, the error at $t = 0.2$ is given in Table 5.3 and the results indicate that the tangential forcing converges at the expected second order rate. For this simple problem we were able to determine exact solutions for every computed

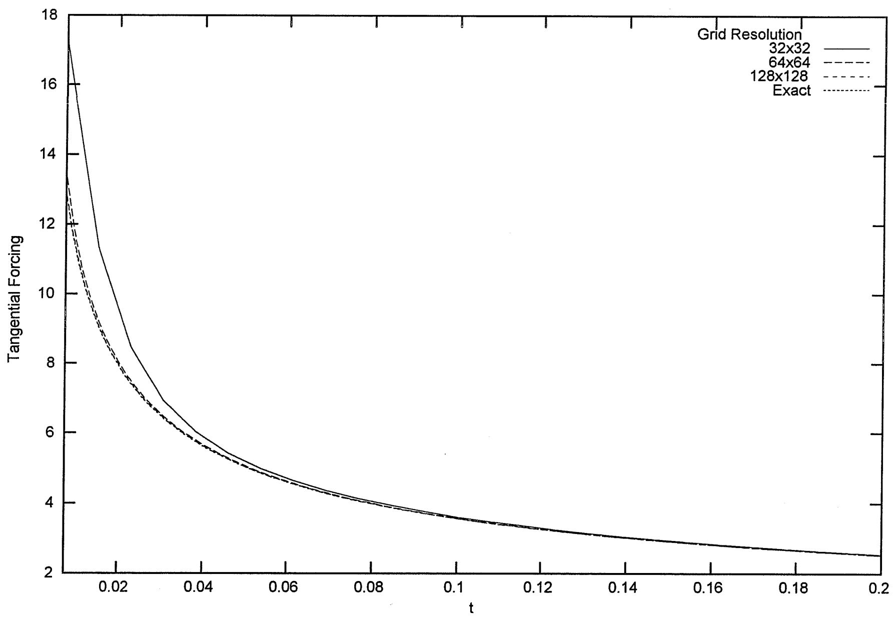

Figure 5.5: Tangential forcing applied at the interface as a function of time for the impulsively started flat plate problem

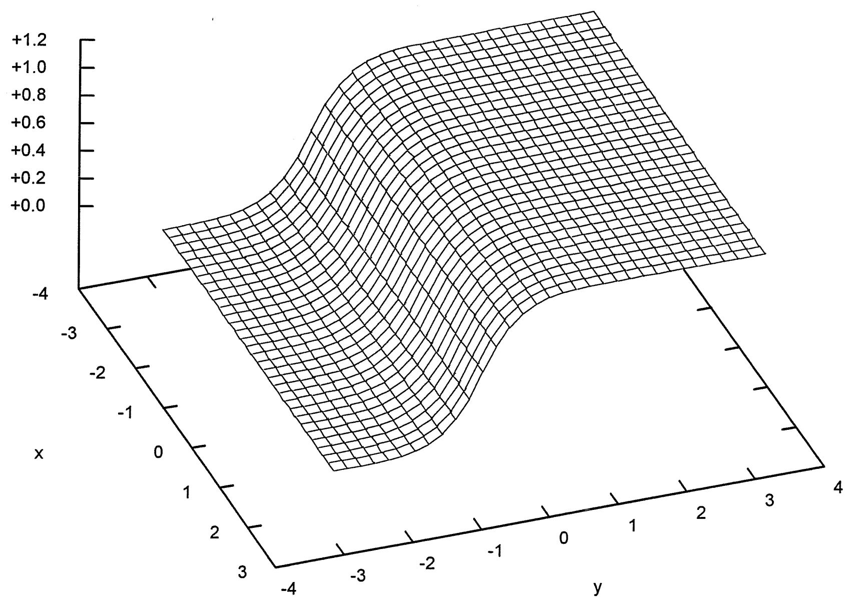

Figure 5.6: Stream function associated with the impulsively started flat plate problem

quantity and uniformly demonstrate second order accuracy. Although we are not able to be quite as thorough in the remaining examples, they will also demonstrate the accuracy of the computed flow field.

5.9.2 Impulsively started cylinder

There is an exact solution to the Navier-Stokes equations for the problem specified by the radially symmetric initial and boundary conditions

$$\begin{aligned} u_\theta(r=a,t) &= \Omega a, && (5.93) \\ u_\theta(r,0) &= 0, \qquad r<a && (5.94) \end{aligned}$$

given in terms of Bessel functions (see [1]) by

$$u_\theta = \Omega r + \sum_{n=0}^{\infty} \frac{2\Omega a}{\lambda_n J_0(\lambda_n)} \exp\left[\frac{-D\lambda_n^2 t}{a^2}\right] J_1\left(\frac{\lambda_n r}{a}\right) \tag{5.95}$$

where u_θ is the velocity in the θ-direction, a is the radius of the impulsively started cylinder and λ_n are the solutions to

$$J_1(\lambda_n) = 0. \tag{5.96}$$

Using the above, we can determine the horizontal and vertical velocities inside the cylinder with

$$u = \left(\frac{-y}{\sqrt{x^2+y^2}}\right) u_\theta, \tag{5.97}$$

and

$$v = \left(\frac{x}{\sqrt{x^2+y^2}}\right) u_\theta. \tag{5.98}$$

We solve this problem numerically on an x-periodic computational domain ($-\pi \leq x, y \leq \pi$) with initial conditions

$$u(x,y,t=0) = 0, \tag{5.99}$$

$$v(x,y,t=0) = 0, \tag{5.100}$$

no-slip boundaries at $y = \pm\pi$ and with the circular interface maintained at

$$u(x = X(q), y = Y(q), t \geq 0) = -\Omega y, \tag{5.101}$$

$$v(x = X(q), y = Y(q), t \geq 0) = +\Omega x. \tag{5.102}$$

Note that $q \in [0,1]$ is the parametrization variable for the interface. In this case, the interface is a periodic cubic spline interpolation of

$$X(q) = +a\cos(2\pi q), \tag{5.103}$$

$$Y(q) = -a\sin(2\pi q), \tag{5.104}$$

using N_c control points (q_i for $0 \leq i < N_c$) placed approximately a distance of $2\Delta x$ apart. The numerical solution will converge to the above analytic one for $r \leq a$. Outside the cylinder, the numerical solution will converge to the flow associated with a periodic array of impulsively started cylinders positioned between two infinite flat plates. While there is no exact solution available to validate the flow on the entire domain, we will verify that it does converge to a solution at a second order rate.

All the simulations are run using $\Delta t = 0.2\Delta x^2$ until $t = 0.2$ with $D = 1$, $a = 2$ and $\Omega = 1$. The numerically determined stream function, horizontal and vertical velocities and pressure are plotted in Figures 5.7, 5.8, 5.9 and 5.10, respectively. The flat region in the center of the cylinder for the stream function and velocity fields indicates that the solution has not reached steady state at $t = 0.2$. The position of the interface is clear from the velocity fields due to the pronounced jumps in the normal derivatives. Note that the appearance of a blunt region, particularly in Fig. 5.9, near the interface is an artifact of the plotting software. The derivatives on both sides of the interface are sharp and jump discontinuously across it. We have computed using $J_{p,avg} = -0.5$ to ease the visualization of the pressure (Fig. 5.10). Note the

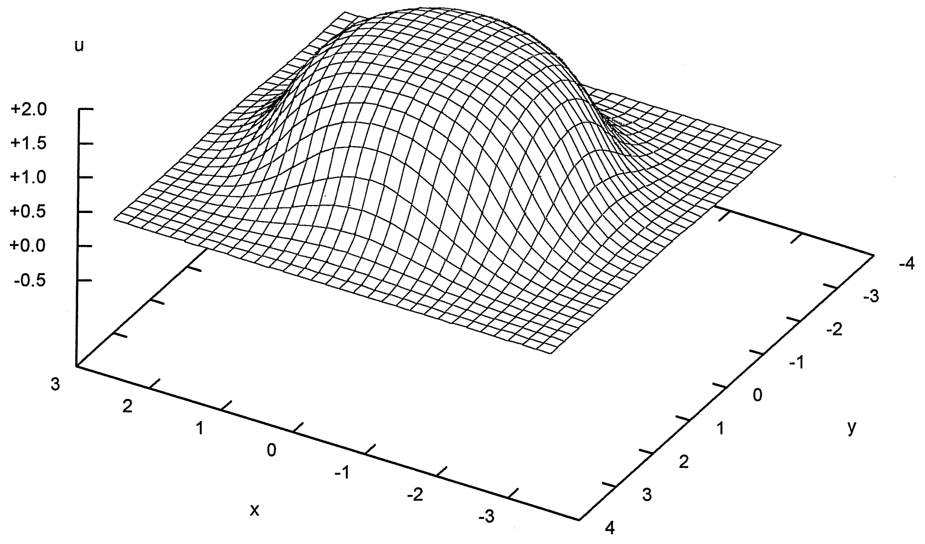

Figure 5.7: Stream function for impulsively started cylinder problem

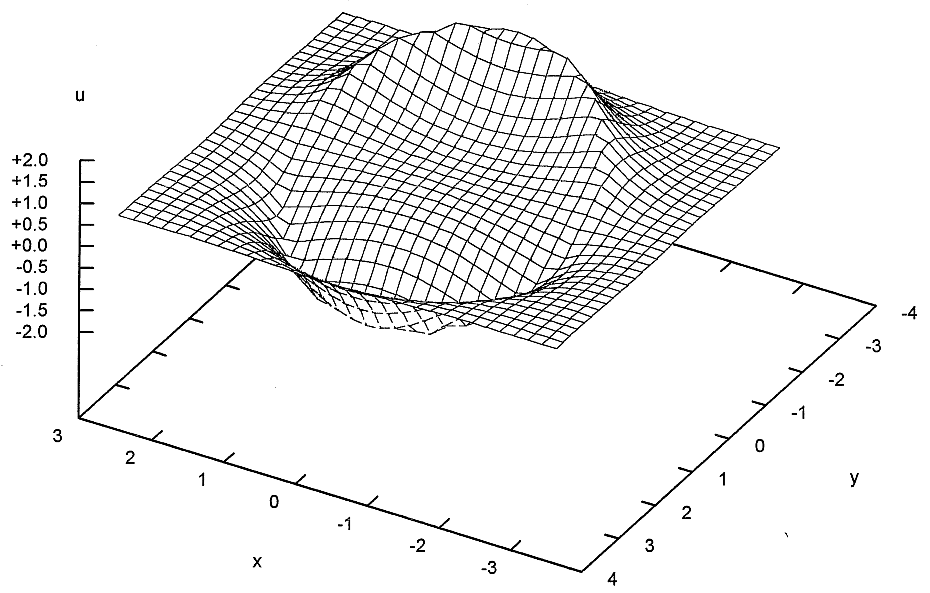

Figure 5.8: Horizontal velocity for the impulsively started cylinder problem

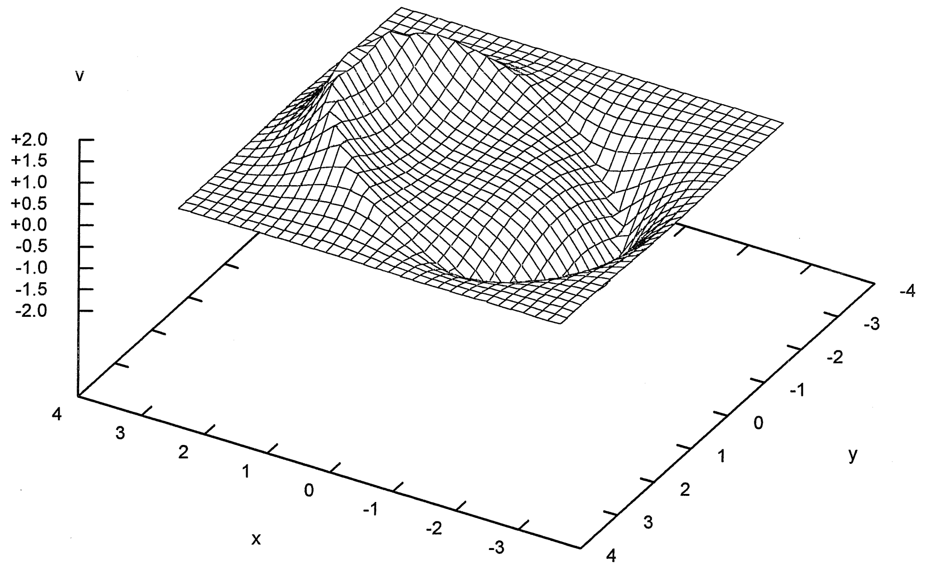

Figure 5.9: Vertical velocity for the impulsely started cylinder problem

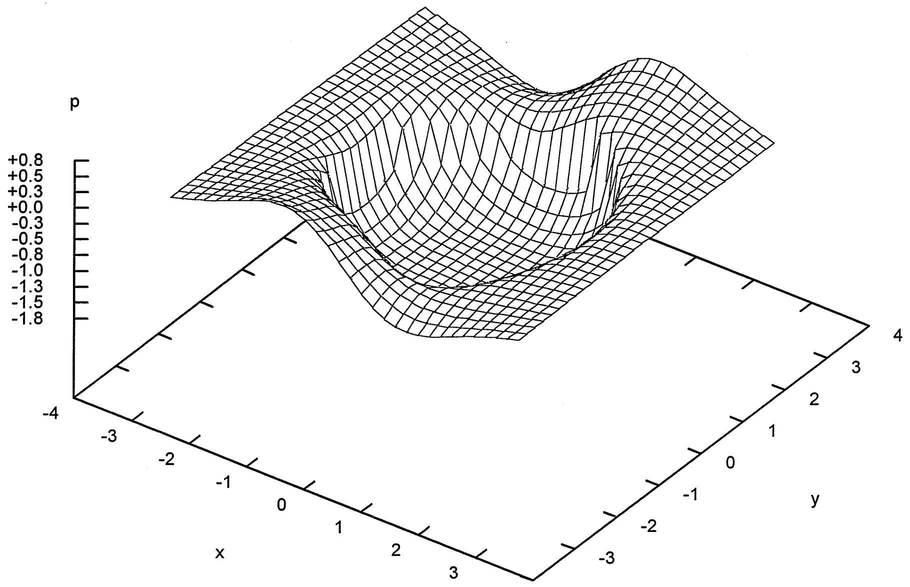

Figure 5.10: Pressure for the impulsively started cylinder problem

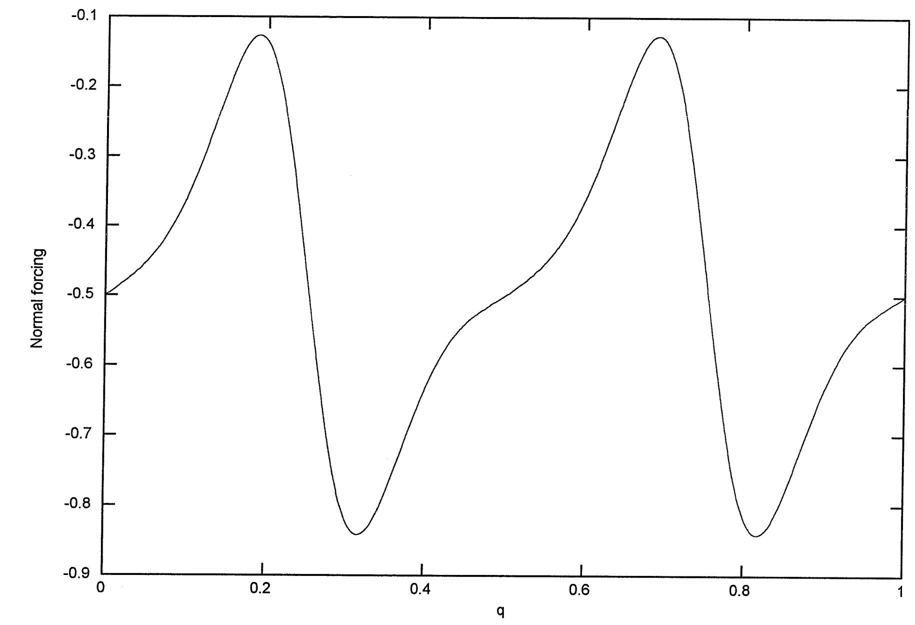

Figure 5.11: Interface forcing in the normal direction for the impulsively started cylinder problem

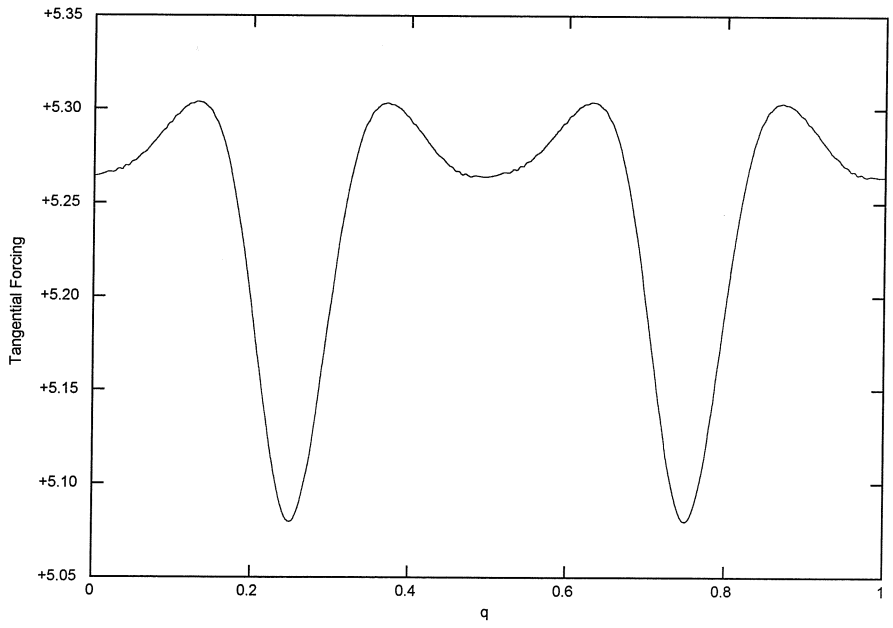

Figure 5.12: Interface forcing in the tangential direction for the impulsively started cylinder problem

$N_x = N_y$	$\|\|u_N - u\|\|_\infty$	Ratio
32	0.01075240	
64	0.00284287	3.78223
128	0.00074858	3.79769
256	0.00018295	4.09175

Table 5.4: Resolution study of the horizontal velocity inside the cylinder

$N_x = N_y$	$\|\|v_N - v\|\|_\infty$	Ratio
32	0.01077110	
64	0.00282913	3.80721
128	0.00074599	3.79244
256	0.00018269	4.08337

Table 5.5: Resolution study of the vertical velocity inside the cylinder

oscillation of the pressure near the domain boundary is required to force the fluid between the no-slip computational boundary and the cylinder. All the flow variables are completely smooth both inside and outside the cylinder. The forcing along the interface is also smooth, as indicated in Figures 5.11 and 5.12.

Inside the cylinder, the numerical and exact solutions for the velocity can be compared. A resolution study of the horizontal and vertical velocities inside the cylinder is provided in Tables 5.4 and 5.5, respectively. The results indicate that the numerical method is second order accurate.

Although it is not possible to determine the exact flow everywhere in the computational domain, we can verify that the numerical solution is indeed converging to a solution everywhere. Using the approach outlined above, the computed convergence rate for all the flow variables is given in Table 5.6. As expected, second order accuracy is indicated. We can also compute the divergence of the flow field on the entire grid. As mentioned above, the discrete divergence of the flow will not, in general, be zero

Variable	Order of Convergence
u	2.02929
v	2.04117
p	1.94944
ψ	2.20631

Table 5.6: Computed rate of convergence for the impulsively started cylinder problem

$N_x = N_y$	$\nabla \cdot \mathbf{u}$	Ratio
32	1.1385×10^{-6}	
64	2.5105×10^{-7}	4.53520
128	8.2173×10^{-8}	3.05497
256	2.0011×10^{-8}	4.10643

Table 5.7: Resolution study of the flow field divergence for the impulsively started cylinder problem

$N_x = N_y$	$\|\|u_{n,N} - u_n\|\|_\infty$	Ratio
32	3.6360×10^{-5}	
64	1.0881×10^{-5}	3.34164
128	1.6498×10^{-7}	65.9518
256	7.7136×10^{-8}	2.13888

Table 5.8: Resolution study of the error in the normal velocity for the impulsively started cylinder problem

(to machine precision) when interface forcing is applied. Instead, the divergence will have a spatially uniform value that converges to zero with the mesh spacing. For this problem the value of the divergence at $t = 0.2$ for the different grid resolutions is listed in Table 5.7. Note that the flow field divergence is quite small and converges to zero as the mesh is refined. Finally, we noted earlier that the normal velocity of the interface can only be specified up to a constant which is determined automatically by the scheme. In this problem, the normal velocity we wish to specify satisfies the conservation of mass so the differences between the exact and computed normal velocity on the interface should vanish as the grid is refined. The error of the normal velocity is listed in Table 5.8. Note that although the error decreases somewhat erratically, comparing the error on the 32x32 and 256x256 grids indicates at least a second order rate of convergence.

5.9.3 Pressure gradient driven flow around cylinders

The above solutions are useful because they allow the numerical results to be compared with exact analytic solutions. They do not fully test the capabilities of the numerical code, however, because the pressure simply cancels against the convective terms. Here we solve a more general problem in which the ability of the pressure to conserve mass is more rigorously tested. We again consider an x-periodic domain $(-\pi \leq x, y \leq \pi)$ with initial conditions

$$\begin{aligned} u\left(x, y, t=0\right) &= 0, & (5.105)\\ v\left(x, y, t=0\right) &= 0, & (5.106)\end{aligned}$$

no-slip boundary conditions on $y = \pm\pi$ and on the interface

$$\begin{aligned} u\left(x = X\left(q\right), y = Y\left(q\right), t \geq 0\right) &= 0, & (5.107)\\ v\left(x = X\left(q\right), y = Y\left(q\right), t \geq 0\right) &= 0. & (5.108)\end{aligned}$$

Note that $q \in [0, 1]$ is the parametrization variable for the interface. In this case the interface is a periodic cubic spline interpolation of the unit circle,

$$\begin{aligned} X\left(q\right) &= +\cos\left(2\pi q\right), & (5.109)\\ Y\left(q\right) &= -\sin\left(2\pi q\right), & (5.110)\end{aligned}$$

using N_c control points (q_i for $0 \leq i < N_c$) placed approximately a distance of $2\Delta x$ apart. In this problem, the flow is driven by a body force (pressure gradient) in the periodic (horizontal) direction,

$$\mathbf{b} = (10, 0), \qquad (5.111)$$

which is applied at all points in the computational domain. Although no exact solution exists for this problem, we will verify that it does converge to a solution at a second order rate.

All the simulations are run using $\Delta t = 0.2\Delta x^2$ until $t = 0.2$ with $D = 1$. The numerically determined stream function, horizontal and vertical velocities and pressure are plotted in Figures 5.13, 5.14, 5.15 and 5.16, respectively. We consider the region inside the cylinder first. Comparing the stream function and velocity fields we see that there is no flow inside the cylinder. Indeed, it is clearly evident from Fig. 5.16 that in this region there is a hydrostatic balance between the computed

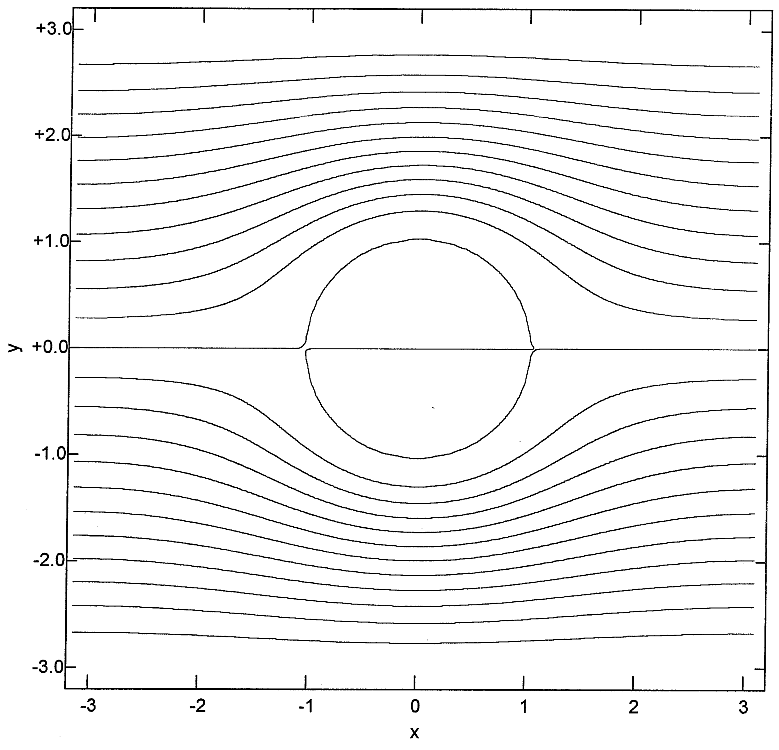

Figure 5.13: Stream lines for pressure driven flow around cylinders

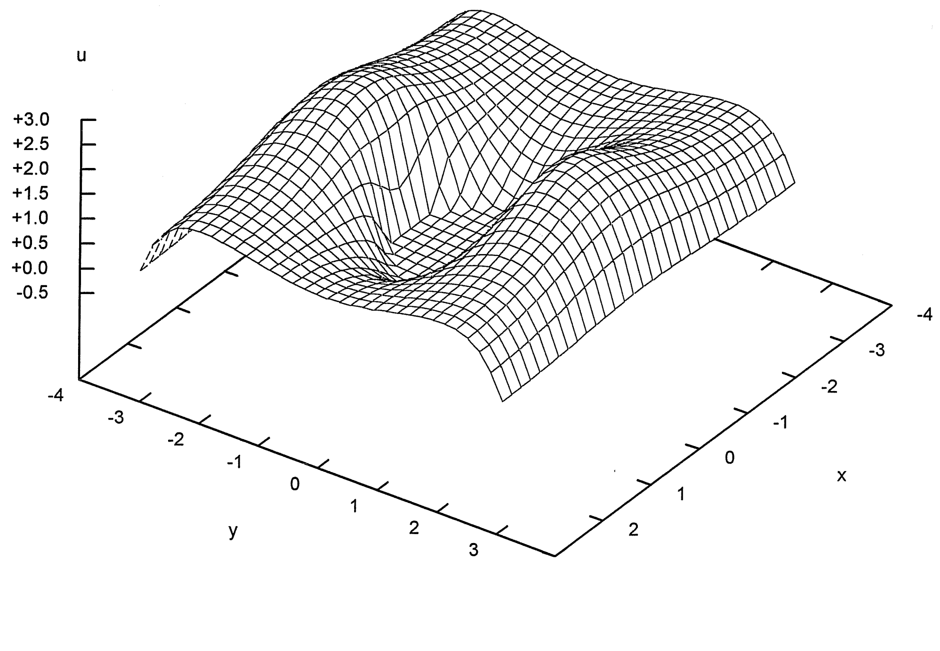

Figure 5.14: Horizontal velocity for pressure driven flow around cylinders

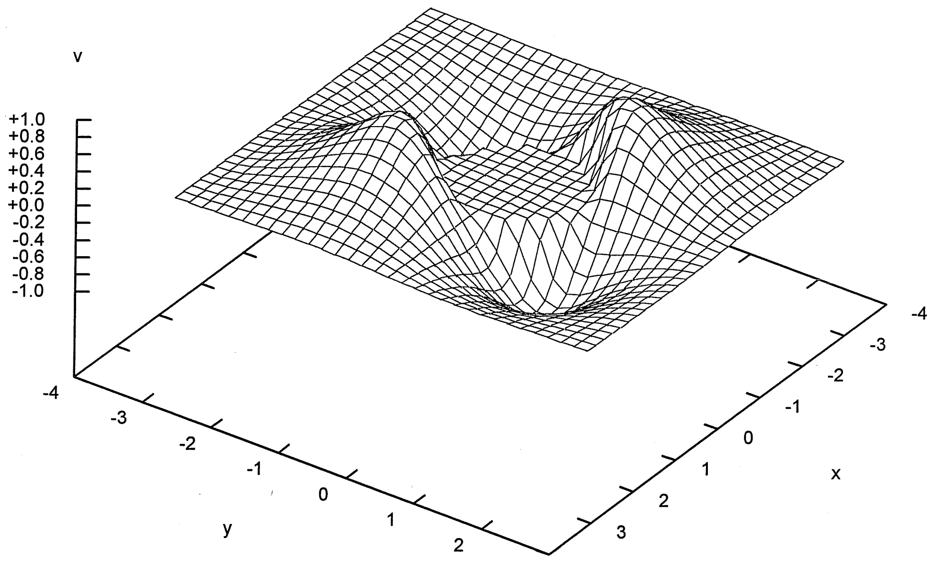

Figure 5.15: Vertical velocity for pressure driven flow around cylinders

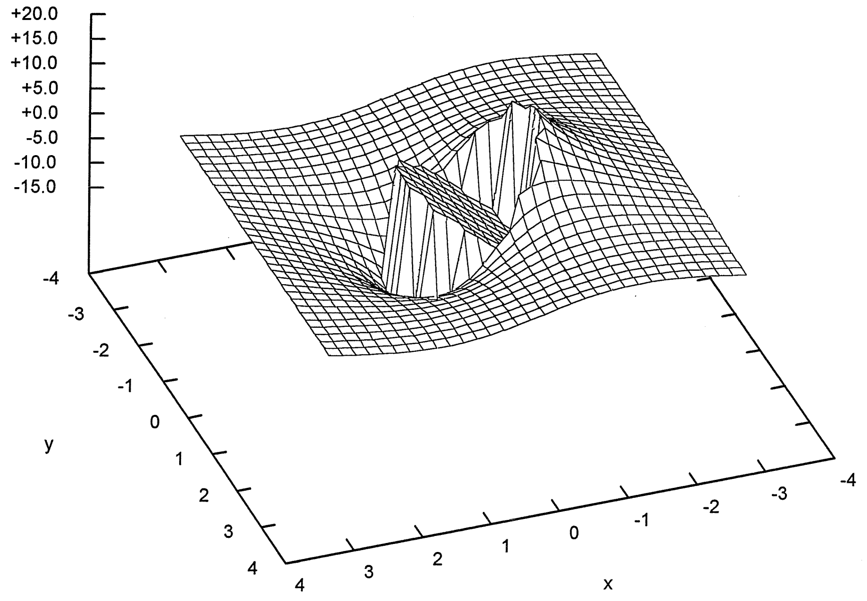

Figure 5.16: Pressure for pressure driven flow around cylinders

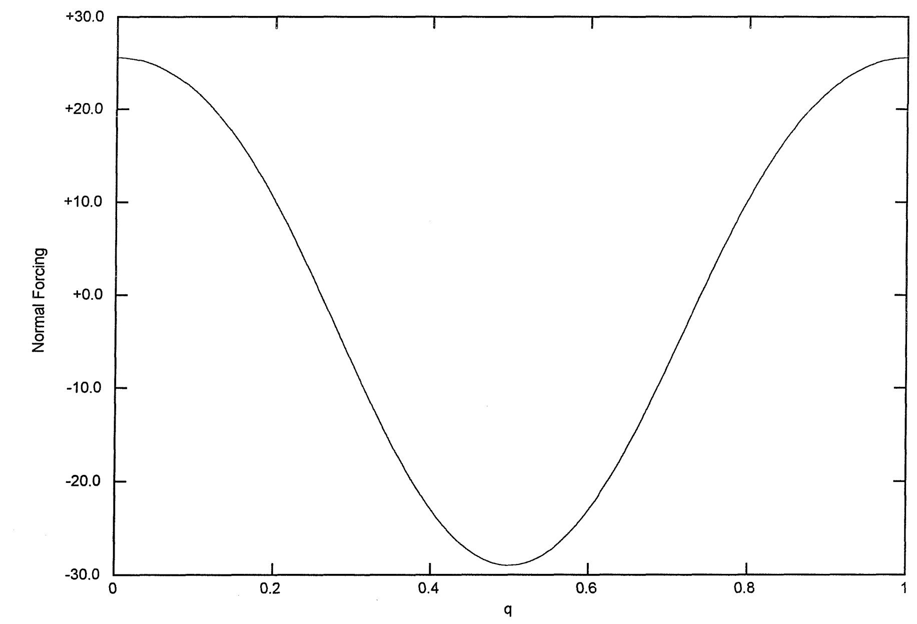

Figure 5.17: Interface forcing in the normal direction for pressure driven flow around cylinders

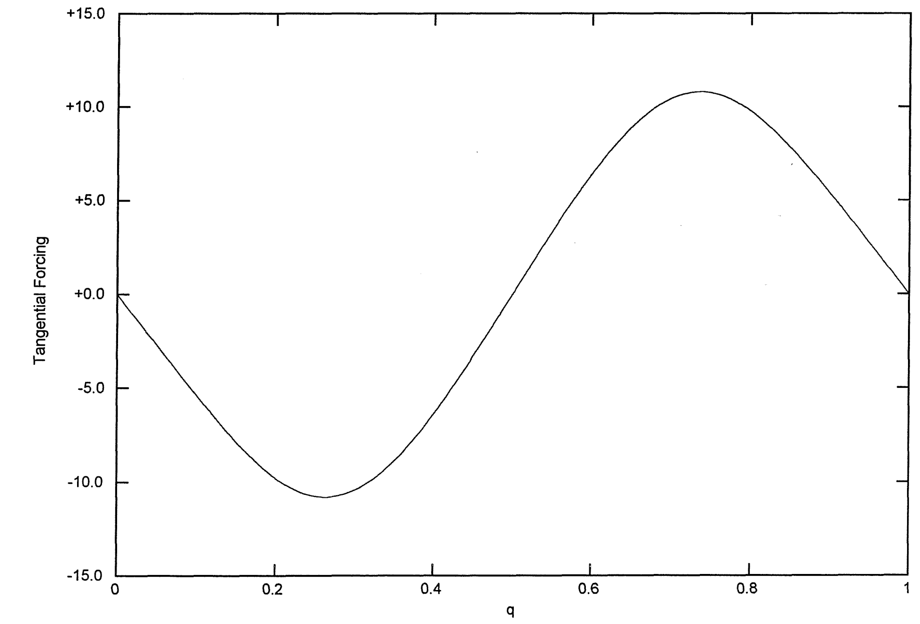

Figure 5.18: Interface forcing in the tangential direction for pressure driven flow around cylinders

$N_x = N_y$	$\nabla \cdot \mathbf{u}$	Ratio
32	3.4235×10^{-6}	
64	1.6658×10^{-7}	20.5513
128	4.7673×10^{-9}	34.9426
256	1.3485×10^{-8}	.35351

Table 5.9: Resolution study of the flow field divergence for the pressure driven problem

pressure ($[p]_{avg} = 0$) and the supplied body force. The flow outside the cylinder is quite different. Here, the fluid is not enclosed and is free to accelerate, as indicated in Figures 5.14 and 5.15. The complete independence of the solutions on each side of the interface is clearly demonstrated in this problem. They are both solutions to the Navier-Stokes equations and each have an arbitrary constant associated with their pressure. We can adjust the relative values of the pressure inside and outside the cylinder anyway we wish, by choosing different values for $J_{p,avg}$, without changing the velocity fields. Even though the flow fields on each side of the interface are physically independent, they are numerically coupled through the determination of the normal, Fig. 5.17, and tangential, Fig. 5.18, interface forcing. Note that the forcing is a smooth function despite the large jumps in the pressure and velocity derivatives evident above. Although we have no exact solution available for this problem, there are several checks that we can perform to verify that the code is working properly.

The first thing we can check is the divergence of numerical solution. As mentioned previously, the continuity condition can not be satisfied exactly when forcing is applied at the interface. Instead the divergence of the flow field is uniformly equal to a small value that converges to zero as the grid is refined. The value of the flow divergence for various resolutions is listed in Table 5.9. The divergence is clearly small and, although it decreases somewhat erratically, comparing the error on the 32x32 and 256x256 grids indicates at least a second order rate of convergence. A quantity related to the flow field divergence that is also easy to calculate is the normal velocity error at the interface.

We mentioned previously that it is not possible to completely specify the normal velocity of the interface. Thus, we only specify it up to a constant which is auto-

$N_x = N_y$	$\|\|u_{n,N} - u_n\|\|_\infty$	Ratio
32	3.1286×10^{-5}	
64	1.1314×10^{-5}	2.76533
128	2.8136×10^{-7}	40.2421
256	2.9561×10^{-8}	9.51036

Table 5.10: Resolution study of the error in the normal velocity for the pressure driven problem

Variable	Order of Convergence
u	1.94978
v	2.14038
p	2.61605
ψ	1.92732

Table 5.11: Computed rate of convergence for the pressure driven problem

matically set by the numerical scheme. The convergence of the numerical normal velocity to the exact normal velocity is another measure of the solution error. The normal velocity error for this problem is listed in Table 5.10 for various grid resolutions. The error clearly decreases with grid resolution and the results indicate a rate of convergence that is at least second order.

Finally, comparing the numerical solutions for three different resolutions (64x64, 128x128 and 256x256 in this case), it is possible to directly approximate the convergence of the computed flow variables. In Table 5.11 we list the computed convergence rates for each variable. Second order accuracy is confirmed for each.

Chapter 6 Solidification with natural convection

6.1 Introduction

In this chapter, we will combine all of the techniques of the preceding chapters and develop a scheme capable of simulating dendritic solidification in the presence of natural convection. Due to the non-uniform temperature profile in systems undergoing dendritic solidification, natural convection is ubiquitous. The solution of the fully coupled fluid and thermal system, however, is a formidable task. Although methods theoretically capable of simulating systems of this type have been developed, simulation of the unstable problem has remained almost virgin territory.

Simulations of melting and stable solidification feature a stable, isothermal interface. In these problems, the domain in which fluid flow must be computed tends to be fairly regular and many different schemes have been used (see below). The unstable solidification problem, however, is much more difficult. The interface in the unstable problem tends to become significantly deformed. This implies that fluid flow must be simulated in an irregular deforming region which can cause many schemes trouble. In fact, the only simulations of unstable dendritic solidification we are aware of impose a fixed shape on the dendrite, significantly simplifying the geometry of the problem. These studies examine the effect of convection on the local thermal/concentration gradients but do not allow the geometry of the interface to respond to such changes (see [10] and [52]). We demonstrate in this chapter that our approach yields a viable method for the simulation of the fully coupled unstable solidification problem with a growing, deforming interface.

6.2 Existing numerical methods

As noted above, we are not aware of any full simulations of dendritic solidification in the presence of natural convection. There is, however, a large body of literature associated with the simulation of stable melting/solidification in the presence of natural convection. Fixed grid methods are quite popular for the stable problem. Fixed grid approaches typically use the enthalpy method (discussed in Chapter 4) to implicitly track the interface and account for latent heat release. The no-slip condition on the solid-liquid interface is commonly modeled using an artificial viscosity or porous media friction term in the solid region (see Chapter 5). Examples of this approach can be found in [26], [12], [54] and [72]. The smooth regular interfaces that are formed during many stable solidification processes lend themselves quite naturally to domain mapping techniques. In this approach, the time dependent physical domain is mapped into a fixed computational domain. Although it can be difficult to determine a mapping that will do this for general problems, in this case the interface is typically assumed to be given by

$$y = F(x,t) \tag{6.1}$$

and the mapped coordinates are given by

$$\begin{aligned} \xi &= x, && (6.2) \\ \eta &= y/F(x,t). && (6.3) \end{aligned}$$

The governing equations are then discretized and solved in the (ξ, η) domain. The position of the interface, $F(\xi, t)$, is not known and appears in the governing equations in (ξ, η) space in a non-linear fashion requiring an iterative scheme be employed to solve the system. An example of this approach applied to a melting problem can be found in [6]. Cut cell methods (see Chapter 5) have been used effectively to a stable solidification problem in the presence of natural convection (see [71] and [60]). Finally, deforming finite element methods (see Chapters 4 and 5) are a popular and accurate technique for simulating the stable problem (see [79] and [47]).

6.3 Governing equations

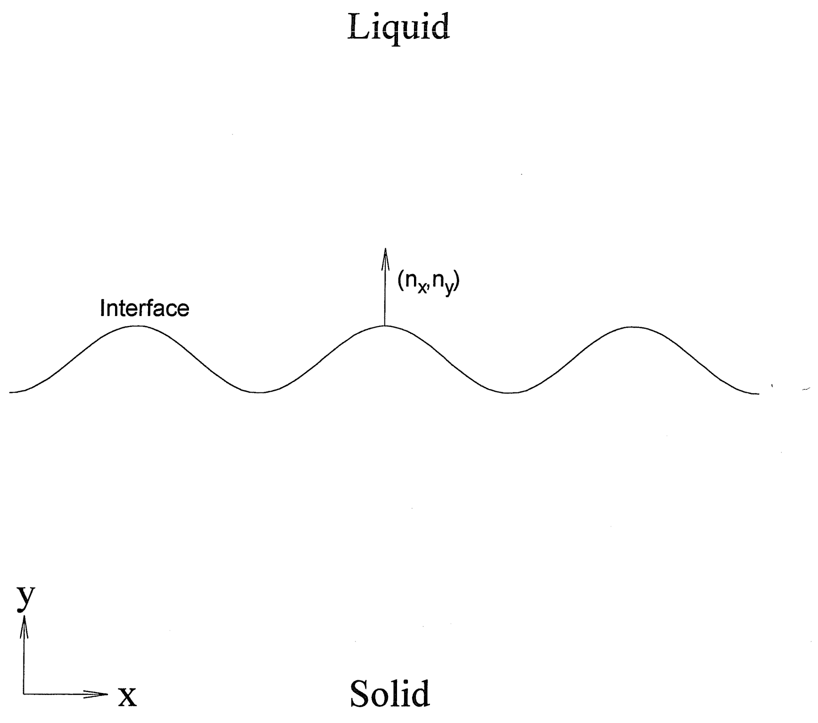

Figure 6.1: Location of the solid and liquid phases relative the interface

The problem we are solving is periodic in the horizontal direction with a periodic interface dividing the domain into two distinct parts, as shown in Fig. 6.1. The solid phase occupies the region below the interface and the liquid occupies the region above it. In the solid phase heat transfer occurs solely by diffusion. Thus, in the solid the temperature satisfies

$$\frac{\partial \theta_S}{\partial t} = H\nabla^2\theta_S. \tag{6.4}$$

In the liquid phase, heat transfer can occur due to diffusion or the bulk transport of material via convection. In the liquid, the temperature satisfies

$$\frac{\partial \theta_L}{\partial t} + \mathbf{u}\cdot\boldsymbol{\nabla}\theta_L = H\nabla^2\theta_L. \tag{6.5}$$

Note that we have assumed symmetric material properties ($H_S = H_L = H$) in (6.4) and (6.5). In this Chapter, we will investigate the symmetric case exclusively. As mentioned previously, material symmetry is not a fundamental limitation of our approach, but is imposed so that the resulting discrete systems can be solved efficiently. The dependence of the temperature on the liquid velocity necessitates that equations governing the fluid motion be solved simultaneously with (6.4) and (6.5). The flow is driven by the buoyant forcing that occurs due to the small but important dependence of the liquid density on temperature. We use the Boussinesq approximation which models this effect as a forcing term in the incompressible Navier-Stokes equations. The equations governing the motion of the liquid are the continuity equation,

$$\boldsymbol{\nabla} \cdot \mathbf{u} = 0, \tag{6.6}$$

and the conservation of momentum,

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \boldsymbol{\nabla} \mathbf{u} = -\nabla p + D\nabla^2 \mathbf{u} + \mathbf{B}\theta_L. \tag{6.7}$$

The conditions on the temperature at the interface are unchanged from the purely diffusive case. On the interface, (X, Y), the temperature must satisfy the Gibbs-Thomson condition

$$\theta_S(X, Y, t) = \theta_L(X, Y, t) = -\varepsilon_c(\mathbf{n})\,\kappa, \tag{6.8}$$

and conservation of energy,

$$\mathbf{n} \cdot (\nabla\theta_S(X, Y, t) - \nabla\theta_L(X, Y, t)) = \frac{1}{h}\left(n_x \frac{\partial X}{\partial t} + n_y \frac{\partial Y}{\partial t}\right). \tag{6.9}$$

The symmetric model implies $\rho_L = \rho_S$, so the fluid satisfies the no-slip condition on the interface,

$$\mathbf{u}(X, Y, t) = \mathbf{0}. \tag{6.10}$$

The above set of equations form a highly non-linear system. The motion of the interface and, hence, the domain occupied by fluid, must be determined simultaneously with the velocity and temperature fields. We now demonstrate that the methods we have developed in the previous chapters can be combined to produce an efficient scheme for simulating this system.

6.4 Discretization

Most of the material in this section is merely a summary of the material from the previous chapters. We will start by discussing the representation of the interface and the determination of the phase of each computational node. Once the phases of the nodes are known, it is possible to discretize the governing equations. We will then discuss the correction of the spatial and temporal stencils that appear in the discrete governing equations. Finally, we review the interpolation scheme used to determine the temperature and fluid velocity on the interface.

6.4.1 Representation of the interface

The interface is explicitly tracked by a series of discrete marker particles. These markers are embedded in the computational domain and allowed to move freely. There is no restriction placed on their position relative to the stencil nodes associated with the temperature or flow variables. In addition to the marker position, (X, Y), we also store the values of the normal velocity, V_N, of the interface as well as the normal, f_N, and tangential, f_τ, components of the force acting on the fluid at each marker particle location. The location of the interface between the markers is determined by a periodic cubic spline interpolant that passes through all the marker particles. The values of V_N, f_τ and f_N between the markers are also determined by periodic cubic spline interpolation.

The marker particles are advanced forward using a first order accurate time marching scheme. The new position of the marker particles is calculated using

$$X^{n+1} = X^n + \Delta t \left(n_x^n V_N^{n+1} \right), \tag{6.11}$$

$$Y^{n+1} = Y^n + \Delta t \left(n_y^n V_N^{n+1} \right), \tag{6.12}$$

where the $n+1$ and n superscripts imply that the quantity they are attached to is evaluated at time t_{n+1} and $t_n = t_0 + n\Delta t$, respectively. Note that even though the above is only first order accurate, the system solution will still improve by a factor of four when the mesh size is halved since $\Delta t = O\left(\Delta x^2\right)$.

6.4.2 Determination of the phase of each node

The determination of which side of the interface (solid or liquid) each computation node resides is handled using the algorithm described in Chapter 4. This algorithm also determines the time at which a node changes phase for all nodes crossed by the interface during the current time step.

6.4.3 The system

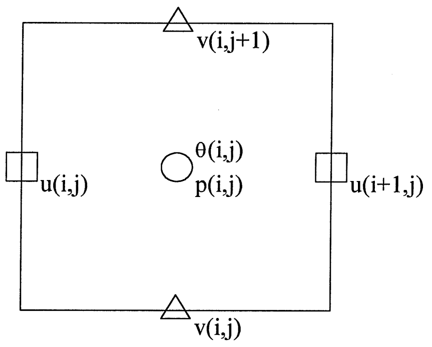

Figure 6.2: Single cell in the staggered grid

The temperature and flow equations are discretized on a staggered grid. As shown in Fig. 6.2, the pressure and temperature nodes are jointly located at the center of the computational cells and the horizontal and vertical components of the velocity are located on the horizontal and vertical sides of the cells, respectively. We will not explicitly write out the discrete form of the spatial derivatives in this section. The stencils that we use are exactly the same as those discussed in Chapters 4 and 5. Detailed expressions for all required spatial stencils can be found in Appendix B.

The time derivatives, on the other hand, will be written out in discrete form and the required corrections for the time derivatives will be derived later in this chapter.

The equations governing the evolution of the temperature, (6.4) and (6.5), can be combined into a single equation. Our discrete equation for the temperature, which applies everywhere in the computational domain, is

$$\theta^{n+1} = \theta^n + \Delta t \left(H \nabla^2 \theta^{n+1} - G^n \right) + E_\theta. \tag{6.13}$$

The value of E_θ is selected to correct the time derivative (discussed later) and the effects of convection are included in G which is defined by

$$G^n = \begin{cases} 0 & \text{if } \theta^n\left(i,j\right) \text{ is a solid node} \\ \mathbf{u}^n \cdot \boldsymbol{\nabla} \theta^n & \text{if } \theta^n\left(i,j\right) \text{ is a liquid node} \end{cases}. \tag{6.14}$$

Although there is only liquid in part of the computational domain, we follow the approach discussed in Chapter 5 and assume that the entire domain is filled with fluid. The resulting flow field will include both the required velocities in the liquid phase and physically irrelevant velocities in the solid domain. The flow calculated in the solid region does not enter into the temperature equation and, hence, does not influence the computed solution in any way. The discrete equations for the flow variables, which apply everywhere in the computational domain, are

$$\mathbf{u}^* = \mathbf{u}^n + \Delta t \left(D \nabla^2 \mathbf{u}^n - \mathbf{u}^n \cdot \boldsymbol{\nabla} \mathbf{u}^n + \mathbf{B} \theta^n - \boldsymbol{\nabla} p^n \right) + \mathbf{E}^*, \tag{6.15}$$

$$\hat{\mathbf{u}} = \mathbf{u}^n + \Delta t \left[\frac{1}{2} D \left(\nabla^2 \mathbf{u}^* + \nabla^2 \mathbf{u}^n \right) - \mathbf{u}^n \cdot \boldsymbol{\nabla} \mathbf{u}^n + \mathbf{B} \theta^{n+1} \right] + \hat{\mathbf{E}}, \tag{6.16}$$

$$\nabla^2 p^{n+1} = \frac{1}{\Delta t} \left(\boldsymbol{\nabla} \cdot \hat{\mathbf{u}} \right), \tag{6.17}$$

$$\mathbf{u}^{n+1} = \hat{\mathbf{u}} - \Delta t \left(\boldsymbol{\nabla} p^{n+1} \right), \tag{6.18}$$

where $\mathbf{B} = (B_x, B_y)$ is a constant and $\mathbf{E}^*$ and $\hat{\mathbf{E}}$ are corrections to the time derivative (discussed later).

6.4.4 Calculation of the jump conditions

The Navier-Stokes equations

Recall in Chapter 5, we developed discretizations for the spatial derivatives of solutions to the incompressible Navier-Stokes equations,

$$\nabla \cdot \mathbf{u} = 0 \tag{6.19}$$

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \boldsymbol{\nabla} \mathbf{u} = -\boldsymbol{\nabla} p + D\nabla^2 \mathbf{u} + \mathbf{b}, \tag{6.20}$$

subject to a continuous body force, $\mathbf{b}$. As noted above, we assume that the entire domain is filled with fluid. Comparing (6.7) and (6.20) we see that we can use

$$\mathbf{b} = \mathbf{B}\theta \tag{6.21}$$

as our body forcing term in each phase (in the liquid, $\theta = \theta_L$ and $\theta = \theta_S$ in the solid). Because the temperature must be continuous at the interface (the Gibbs-Thomson condition), this yields a continuous body force. The above will accurately predict the fluid velocity in the liquid phase provided that the forcing on the interface, (f_τ, f_N), is determined such that the no-slip condition on the interface, (X, Y), is maintained:

$$\mathbf{u}(X, Y, t) = \mathbf{0}. \tag{6.22}$$

There will also be an independent flow calculated in the solid region that has no physical relevance and does not influence the temperature of the solid. The jump conditions for this problem are identical to those presented in Chapter 5. For completeness, we repeat them here. Using (6.22), the jumps in the pressure and its derivatives are given by

$$[p] = f_n, \tag{6.23}$$

$$[p_x] = \tau_x \frac{\partial f_n}{\partial s} + n_x \frac{\partial f_\tau}{\partial s}, \tag{6.24}$$

$$[p_y] = \tau_y \frac{\partial f_n}{\partial s} + n_y \frac{\partial f_\tau}{\partial s}, \tag{6.25}$$

$$[p_{xx}] = +\tau_x \frac{\partial [p_x]}{\partial s} - \tau_y \frac{\partial [p_y]}{\partial s} + \tau_y^2 \left(B_x [\theta_x] + B_y [\theta_y]\right), \tag{6.26}$$

$$[p_{xy}] = +\tau_y \frac{\partial [p_x]}{\partial s} + \tau_x \frac{\partial [p_y]}{\partial s} - \tau_x \tau_y \left(B_x [\theta_x] + B_y [\theta_y]\right), \tag{6.27}$$

$$[p_{yy}] = -\tau_x \frac{\partial [p_x]}{\partial s} + \tau_y \frac{\partial [p_y]}{\partial s} + \tau_x^2 \left(B_x [\theta_x] + B_y [\theta_y]\right). \tag{6.28}$$

The jumps in the horizontal velocity and its derivatives are given by

$$[u] = 0, \tag{6.29}$$

$$[u_x] = -n_x \tau_x \frac{f_\tau}{D}, \tag{6.30}$$

$$[u_y] = -n_y \tau_x \frac{f_\tau}{D}, \tag{6.31}$$

$$[u_t] = -\left(\frac{\partial X}{\partial t} [u_x] + \frac{\partial Y}{\partial t} [u_y]\right), \tag{6.32}$$

$$[u_{xx}] = +\tau_x \frac{\partial [u_x]}{\partial s} - \tau_y \frac{\partial [u_y]}{\partial s} + \frac{\tau_y^2}{D} \left([u_t] + [p_x]\right), \tag{6.33}$$

$$[u_{xy}] = +\tau_y \frac{\partial [u_x]}{\partial s} + \tau_x \frac{\partial [u_y]}{\partial s} - \frac{\tau_x \tau_y}{D} \left([u_t] + [p_x]\right), \tag{6.34}$$

$$[u_{yy}] = -\tau_x \frac{\partial [u_x]}{\partial s} + \tau_y \frac{\partial [u_y]}{\partial s} + \frac{\tau_x^2}{D} \left([u_t] + [p_x]\right). \tag{6.35}$$

The jumps in the vertical velocity and its derivatives are given by

$$[v] = 0, \tag{6.36}$$

$$[v_x] = -n_x \tau_y \frac{f_\tau}{D}, \tag{6.37}$$

$$[v_y] = -n_y \tau_y \frac{f_\tau}{D}, \tag{6.38}$$

$$[v_t] = -\left(\frac{\partial X}{\partial t} [v_x] + \frac{\partial Y}{\partial t} [v_y]\right), \tag{6.39}$$

$$[v_{xx}] = +\tau_x \frac{\partial [v_x]}{\partial s} - \tau_y \frac{\partial [v_y]}{\partial s} + \frac{\tau_y^2}{D}([v_t] + [p_y]), \tag{6.40}$$

$$[v_{xy}] = +\tau_y \frac{\partial [v_x]}{\partial s} + \tau_x \frac{\partial [v_y]}{\partial s} - \frac{\tau_x \tau_y}{D}([v_t] + [p_y]), \tag{6.41}$$

$$[v_{yy}] = -\tau_x \frac{\partial [v_x]}{\partial s} + \tau_y \frac{\partial [v_y]}{\partial s} + \frac{\tau_x^2}{D}([v_t] + [p_y]). \tag{6.42}$$

The heat equation

In Chapter 4 we calculated the jump conditions appropriate for the solidification problem in the absence of convection. We demonstrate below that the inclusion of the convection terms in the liquid phase does not modify the jump conditions on the temperature, at least through the second order derivatives.

The conditions on the temperature at the interface are unchanged by the inclusion of convection in the liquid phase. Thus the jump condition on the temperature and its normal derivative are identical to those found in Chapter 4. This implies that the jump conditions on the first derivatives are also unchanged,

$$[\theta_x] = n_x [\theta_n], \tag{6.43}$$

$$[\theta_y] = n_y [\theta_n], \tag{6.44}$$

$$[\theta_t] = -\left(\frac{\partial X}{\partial t}[\theta_x] + \frac{\partial Y}{\partial t}[\theta_y]\right) = -h[\theta_n]^2. \tag{6.45}$$

The value of $[\theta_n]$ is related to the normal velocity of the interface by (6.9) which implies

$$[\theta_n] = \frac{V_N}{h}. \tag{6.46}$$

Recall that the normal velocity is one of the quantities that is stored at the marker particles and which must be determined as part of the solution. To determine the jump in the second derivatives, we will need to examine the equations governing the evolution of the temperature. The equations governing the temperature in the solid and liquid phases can be combined, formally at least, as a single Poisson equation subject to a forcing that is discontinuous at the interface,

$$\nabla^2 \theta = f, \tag{6.47}$$

where

$$f = \begin{cases} \dfrac{1}{H}\dfrac{\partial \theta}{\partial t} & \text{In the solid} \\ \dfrac{1}{H}\left(\dfrac{\partial \theta}{\partial t} + \mathbf{u}\cdot\boldsymbol{\nabla}\theta\right) & \text{In the liquid} \end{cases} . \tag{6.48}$$

From (6.22) we see that the jump in the effective forcing,

$$[f] = \frac{1}{H}\left[\theta_t\right], \tag{6.49}$$

is exactly the same as it was for the pure diffusion case implying that the jumps in the second derivatives will remain the same as well. They are given by

$$[\theta_{xx}] = +\tau_x \frac{\partial\left[\theta_x\right]}{\partial s} - \tau_y \frac{\partial\left[\theta_y\right]}{\partial s} - \tau_y^2 \frac{h}{H}\left[\theta_n\right]^2, \tag{6.50}$$

$$[\theta_{xy}] = +\tau_y \frac{\partial\left[\theta_x\right]}{\partial s} + \tau_x \frac{\partial\left[\theta_y\right]}{\partial s} + \tau_x\tau_y \frac{h}{H}\left[\theta_n\right]^2, \tag{6.51}$$

$$[\theta_{yy}] = -\tau_x \frac{\partial\left[\theta_x\right]}{\partial s} + \tau_y \frac{\partial\left[\theta_y\right]}{\partial s} - \tau_x^2 \frac{h}{H}\left[\theta_n\right]^2. \tag{6.52}$$

The jumps in the third derivatives differ from the purely diffusive case because the jump in the normal derivative of the effective forcing, $[f_n]$, picks up a non-zero contribution from the convective term. This contribution would have to be explicitly interpolated from the nodal values, however, so we will only correct through the second order derivatives for these simulations.

6.4.5 Correction of spatial stencils

The correction of the spatial stencils follows the standard approach discussed in Chapters 3 through 5. The problem we are solving satisfies symmetric jump conditions so we use standard finite difference stencils with additive corrections. It is important to note that any quantity that is calculated through the use of a spatial stencil must be corrected if one of its stencil legs is intersected by the interface. This is not just limited to derivatives but includes the averages that must be calculated due to our use of a staggered grid. For instance, the body forcing term, $B_y\theta$, in the vertical momentum equation requires that the value of the temperature at the $v\left(i,j\right)$ node be interpolated using

$$\theta = \frac{1}{2}\left(\theta\left(i,j\right) + \theta\left(i,j-1\right)\right). \tag{6.53}$$

If the interface cuts the stencil leg connecting the $v\left(i,j\right)$ node to the $\theta\left(i,j\right)$ node, then we substitute

$$\theta\left(i,j\right) := \theta\left(i,j\right) - C \tag{6.54}$$

or

$$\theta\left(i,j\right) := \theta\left(i,j\right) + C \tag{6.55}$$

into (6.53) if the $v\left(i,j\right)$ node is a solid or liquid node, respectively. The correction is given by

$$C = \left(y - Y\right)\left[\theta_y\right] + \frac{1}{2!}\left(y - Y\right)^2\left[\theta_{yy}\right], \tag{6.56}$$

where (x, y) is the position of the $\theta\left(i,j\right)$ node and (X, Y) is the point on the interface that intersects the stencil leg (note, in this case $x = X$). The values of the jumps are all evaluated at the point of intersection between the stencil leg and the interface. All other spatial stencils can be corrected in a similar manner.

6.4.6 Correction of time derivatives

Following the approach outlined in Chapter 3, we utilize the values of $[\theta_t]$, $[u_t]$ and $[v_t]$ to correct the time derivatives at nodes which undergo a phase change. To calculate the correction to the time derivative of the temperature, we use

$$\theta^{n+1} = \theta^n + \int_{t_n}^{t_{n+1}} \frac{\partial\theta}{\partial t}dt. \tag{6.57}$$

It is important to take care when translating the formulas developed in Chapter 3. Recall that, in Chapter 3, we defined

$$[\theta_t] = \theta_t\left(t_n + t_I^+\right) - \theta_t\left(t_n + t_I^-\right) = \theta_t^+ - \theta_t^- \tag{6.58}$$

which means that $[\theta_t]$ is the value of the time derivative after the discontinuity minus the value of the time derivative before the discontinuity. The definition of $[\theta_t]$ in this chapter, however, is

$$[\theta_t] = \lim_{\varepsilon\to 0^+} \left\{\theta_t\left(X + \varepsilon n_x, Y + \varepsilon n_y, t\right) - \theta_t\left(X - \varepsilon n_x, Y - \varepsilon n_y, t\right)\right\} = \theta_t^+ - \theta_t^-, \tag{6.59}$$

where (X, Y) is the location of the interface and (n_x, n_y) is the interface normal pointing away from the solid. From the definition of the normal, the liquid phase is the "+" side of the interface and the solid is the "-" side of the interface. Thus, formulas developed in Chapter 3 can be used directly provided we note that they apply at a node which starts out solid (i.e., $\theta_t^n \approx \theta_t^-$) and changes phase to liquid (i.e., $\theta_t^{n+1} \approx \theta_t^+$) during the time step. If we are interested in the more typical case of a node changing from a liquid to a solid, we simply make the substitution

$$[\theta_t] \to -[\theta_t] \tag{6.60}$$

in the correction term. We will derive the appropriate corrections for the solid to liquid transition case and give the general form for the correction terms at the end. Thus, using (6.13), (6.57) and Chapter 3 implies

$$\theta^{n+1} = \theta^n + \Delta t\left(H\nabla^2\theta^{n+1} - G^n\right) - t_I\left[H\nabla^2\theta\right] - \left(\Delta t - t_I\right)[G], \tag{6.61}$$

where the above assumes that the $\theta^n(i, j)$ node is solid at t_n and liquid at t_{n+1} with the phase change occurring at $t_n + t_I$. Using the fact that (6.4) and (6.5) can be combined as

$$\theta_t + G = H\nabla^2\theta, \tag{6.62}$$

we see that

$$[\theta_t] + [G] = \left[H\nabla^2\theta\right]. \tag{6.63}$$

Substituting (6.63) into (6.61) yields

$$\theta^{n+1} = \theta^n + \Delta t\left(H\nabla^2\theta^{n+1} - G^n\right) - t_I\,[\theta_t] - \Delta t\,[G]. \tag{6.64}$$

For our problem (6.10) implies $[G] = 0$. Thus we have

$$E_\theta = \begin{cases} 0 & \text{if } \theta^n\,(i,j) \text{ does not change phase} \\ -t_I\,[\theta_t] & \text{if } \theta^n\,(i,j) \text{ is solid at } t_n \\ +t_I\,[\theta_t] & \text{if } \theta^n\,(i,j) \text{ is liquid at } t_n \end{cases}. \tag{6.65}$$

The determination of the fluid velocity requires two evolution equations be solved for each flow component. The first is used to obtain an explicit prediction for the flow at t_{n+1}. Casting the equation in terms of an integration we have

$$\mathbf{u}^* = \mathbf{u}^n + \int_{t_n}^{t_{n+1}} \frac{\partial \mathbf{u}}{\partial t}dt. \tag{6.66}$$

In this case, all the terms making up the time derivative are evaluated at t_n so we can use the standard correction derived in Chapter 3,

$$\mathbf{u}^* = \mathbf{u}^n + \Delta t\left(D\nabla^2\mathbf{u}^n - \mathbf{u}^n\cdot\boldsymbol{\nabla}\mathbf{u}^n + \mathbf{B}\theta^n - \boldsymbol{\nabla}p^n\right) + (\Delta t - t_I)\,[\mathbf{u}_t], \tag{6.67}$$

where we assume that the $u^n\,(i,j)$ or $v^n\,(i,j)$ node is solid at t_n and liquid at t_{n+1} with the phase change occurring at $t_n + t_I$. Comparing (6.15) with (6.67) and using $\mathbf{E}^* = (E_u^*, E_v^*)$, we find

$$E_u^* = \begin{cases} 0 & \text{if } u^n\,(i,j) \text{ does not change phase} \\ +(\Delta t - t_I)\,[u_t] & \text{if } u^n\,(i,j) \text{ is solid at } t_n \\ -(\Delta t - t_I)\,[u_t] & \text{if } u^n\,(i,j) \text{ is liquid at } t_n \end{cases} \tag{6.68}$$

and

$$E_v^* = \begin{cases} 0 & \text{if } v^n(i,j) \text{ does not change phase} \\ +(\Delta t - t_I)[v_t] & \text{if } v^n(i,j) \text{ is solid at } t_n \\ -(\Delta t - t_I)[v_t] & \text{if } v^n(i,j) \text{ is liquid at } t_n \end{cases} . \tag{6.69}$$

To derive the time correction for the intermediate velocity $\hat{\mathbf{u}}$, we will correct the effective evolution equation for $\mathbf{u}^{n+1}$. We begin by noting that

$$\mathbf{u}^{n+1} = \mathbf{u}^n + \int_{t_n}^{t_{n+1}} \frac{\partial \mathbf{u}}{\partial t} dt, \tag{6.70}$$

and (6.16) implies

$$\begin{aligned} \mathbf{u}^{n+1} &= \mathbf{u}^n + \Delta t \left[\frac{1}{2} D\left(\nabla^2 \mathbf{u}^* + \nabla^2 \mathbf{u}^n\right) - \mathbf{u}^n \cdot \boldsymbol{\nabla} \mathbf{u}^n + \mathbf{B}\theta^{n+1} - \nabla p^{n+1}\right] \qquad (6.71) \\ &+ \left(\frac{1}{2}\Delta t - t_I\right)\left[D\nabla^2 \mathbf{u}\right] - (\Delta t - t_I)\left[\mathbf{u} \cdot \boldsymbol{\nabla} \mathbf{u}\right] - t_I\left([\mathbf{B}\theta] - [\nabla p]\right), \end{aligned}$$

where the corrections again assume that the $u^n(i,j)$ or $v^n(i,j)$ node is changing from solid to liquid. We have also used the fact that $\mathbf{u}^* = \mathbf{u}^{n+1} + O(\Delta t^2)$. The momentum equation implies

$$[\mathbf{u}_t] + [\mathbf{u} \cdot \boldsymbol{\nabla} \mathbf{u}] = \left[D\nabla^2 \mathbf{u}\right] + [\mathbf{B}\theta] - [\boldsymbol{\nabla} p], \tag{6.72}$$

which, when substituted into the above, yields

$$\begin{aligned} \mathbf{u}^{n+1} &= \mathbf{u}^n + \Delta t \left[\frac{1}{2} D\left(\nabla^2 \mathbf{u}^* + \nabla^2 \mathbf{u}^n\right) - \mathbf{u}^n \cdot \boldsymbol{\nabla} \mathbf{u}^n + \mathbf{B}\theta^{n+1} - \boldsymbol{\nabla} p^{n+1}\right] \qquad (6.73) \\ &+ \left(\frac{1}{2}\Delta t - t_I\right)[\mathbf{u}_t] + \frac{1}{2}\Delta t\left([\boldsymbol{\nabla} p] - [\mathbf{u} \cdot \boldsymbol{\nabla} \mathbf{u}] - [\mathbf{B}\theta]\right). \end{aligned}$$

Noting that $[\mathbf{u} \cdot \boldsymbol{\nabla} \mathbf{u}] = \mathbf{0}$ by (6.10) and $[\mathbf{B}\theta] = \mathbf{0}$ due to the continuity of the temperature at the interface, we find

$$\begin{aligned} \mathbf{u}^{n+1} &= \mathbf{u}^n + \Delta t \left[\frac{1}{2} D\left(\nabla^2 \mathbf{u}^* + \nabla^2 \mathbf{u}^n\right) - \mathbf{u}^n \cdot \boldsymbol{\nabla} \mathbf{u}^n + \mathbf{B}\theta^{n+1} - \boldsymbol{\nabla} p^{n+1}\right] \qquad (6.74) \\ &+ \left(\frac{1}{2}\Delta t - t_I\right)[\mathbf{u}_t] + \frac{1}{2}\Delta t\,[\boldsymbol{\nabla} p]. \end{aligned}$$

Using $\hat{\mathbf{u}} = \mathbf{u}^{n+1} + \Delta t \boldsymbol{\nabla} p^{n+1}$ and comparing the above to (6.16) yields

$$\hat{E}_u = \begin{cases} 0 & \text{if } u^n\left(i,j\right) \text{ does not change phase} \\ +\left(\frac{1}{2}\Delta t - t_I\right)\left[u_t\right] + \frac{1}{2}\Delta t\left[p_x\right] & \text{if } u^n\left(i,j\right) \text{ is solid at } t_n \\ -\left(\frac{1}{2}\Delta t - t_I\right)\left[u_t\right] - \frac{1}{2}\Delta t\left[p_x\right] & \text{if } u^n\left(i,j\right) \text{ is liquid at } t_n \end{cases} \tag{6.75}$$

and

$$\hat{E}_v = \begin{cases} 0 & \text{if } v^n\left(i,j\right) \text{ does not change phase} \\ +\left(\frac{1}{2}\Delta t - t_I\right)\left[v_t\right] + \frac{1}{2}\Delta t\left[p_y\right] & \text{if } v^n\left(i,j\right) \text{ is solid at } t_n \\ -\left(\frac{1}{2}\Delta t - t_I\right)\left[v_t\right] - \frac{1}{2}\Delta t\left[p_y\right] & \text{if } v^n\left(i,j\right) \text{ is liquid at } t_n \end{cases} . \tag{6.76}$$

6.4.7 Interpolation of interface values

In addition to evolving the partial differential equations for the temperature and flow variables forward in time, it is necessary to satisfy the conditions at the interface as well. By the construction of our discrete stencils, the jump in the normal derivative of the temperature, (6.9), is automatically satisfied. The values of the interface normal velocity, V_N, and the interfacial fluid forcing, (f_τ, f_N), are determined by the remaining requirements which are that the Gibbs-Thomson condition, (6.8), and no-slip condition, (6.22), be satisfied. To determine whether the Gibbs-Thomson and no-slip conditions are satisfied at the interface marker particles, it is necessary interpolate the values of the temperature and fluid velocity off the computational grid. We use the weighted Taylor series interpolation scheme discussed in Chapter 4. As discussed previously, this provides a smoothly varying estimate from the temperature and fluid velocity on the interface as the marker particles move through the mesh.

6.5 Solution algorithm

The complete solution algorithm is essentially just a combination of the algorithms presented in Chapters 4 and 5. Note from (6.13) that only the values of $\mathbf{u}^n$ are required to determine θ^{n+1}. This means that the convection terms are essentially just an additional forcing that must be added to heat equation. The algorithm for determination of the normal velocity, presented in Chapter 4, is unaffected by the addition of a forcing term so it applies equally well to the fully coupled problem. Similarly, once θ^{n+1} and the new position of the interface is determined, the buoyancy term that couples the Navier-Stokes and heat equations together is essentially just a known forcing. Thus, the algorithm presented in Chapter 5 to determine the new interfacial forcing, $(f_\tau, f_n)^{n+1}$, applies to this problem as well.

For the algorithm presented below, we assume that the values of θ^n, $\mathbf{u}^n$, p^n, V_N^n and $(f_\tau, f_N)^n$ are known. If this is the first time step, these values can be extracted from the initial conditions. The complete algorithm to advance the interface, temperature and flow field forward one time step is

1. Using the algorithm from Chapter 4, determine θ^{n+1}, $(X,Y)^{n+1}$ and V_N^{n+1} by iterating until the error associated with the Gibbs-Thomson condition is sufficiently small, $\|e_i\| < tol$.

2. Use the BiConjugate Gradient Stabilized iteration scheme presented in Chapter 5 to determine the values of $\mathbf{u}^{n+1}$, p^{n+1} and $(f_\tau, f_N)^{n+1}$ such that $\mathbf{u}^{n+1} = \mathbf{0}$ at the new interface location.

3. Determine the distance along the interface, s_i, of each marker particle and the total arc length, $S = s_{N_c}$, associated with the new interface position determined in 1.

4. Assuming there are N_c marker particles, if $S/N_c > 5\Delta x/2$, increase the number of marker particles so that $S/N_c = 2\Delta x$ is satisfied as closely as possible.

5. Using linear interpolation and the values of s_i calculated above, determine the parameter values, q_i^*, associated with the new position of the marker particles such that they are equally spaced along the interface.

6. Regrid the interface quantities X^{n+1}, Y^{n+1}, V_N^{n+1}, f_τ^{n+1} and f_N^{n+1} by interpolating their values at $q = q_i^*$ and use these to establish a new set of evenly spaced marker particles.

N_xxN_y	$\|\|\theta_N - \theta\|\|_\infty$	Ratio
16x64	3.43x10^{-3}	
32x128	8.77x10^{-4}	3.92
64x256	2.17x10^{-4}	4.03

Table 6.1: Resolution study of the temperature error for the $B_y = -20$ case

7. Update all the time dependent quantities and continue with next time step (go back to step 1).

We have used the above algorithm to compute the solution to several different problems. In all cases we select the time step such that

$$\Delta t = \min\left\{\frac{\Delta x^2}{5D}, \Delta t_I\right\}, \tag{6.77}$$

where Δt_I is the largest time step we can take such that the interface travels a distance less than $\Delta x/10$, i.e.,

$$(V_N)_{\max} \Delta t_I \leq \frac{\Delta x}{10}. \tag{6.78}$$

The above choices for Δt ensure that the explicit time stepping used in our Navier-Stokes solver will be stable.

6.6 Results

In this section we will compare the results of our numerical scheme with linear theory and verify that our simulations are indeed second order accurate. We will start the code at $t = 0$ using the linear theory solution as an initial condition with $\varepsilon = 10^{-4}$, $V = 1/2$, $H = D = 1$, $h = 1$, $\delta_0 = 1/100$, $\delta_1 = 0$, $B_y = -20$ and $a = 1$. Recall that this corresponds to the case where linear theory predicted that convection would have a strong impact on the evolution of the system.

Tables 6.1-6.3 compare the difference between the numerical solution and linear theory at $t = 8$. Table 6.1 demonstrates the second order accurate convergence of the temperature while Tables 6.2 and 6.3 demonstrate the second order accuracy of the horizontal and vertical velocities, respectively. Table 6.4 shows the convergence

N_xxN_y	$\|\|u_N - u\|\|_\infty$	Ratio
16x64	7.69x10^{-5}	
32x128	1.47x10^{-5}	5.21
64x256	3.90x10^{-6}	3.78

Table 6.2: Resolution study of the error in the horizontal velocity for the $B_y = -20$ case

N_xxN_y	$\|\|v_N - v\|\|_\infty$	Ratio
16x64	5.99x10^{-5}	
32x128	8.43x10^{-6}	7.10
64x256	2.14x10^{-6}	3.94

Table 6.3: Resolution study of the error in the vertical velocity for the $B_y = -20$ case

N_xxN_y	$\|\|\nabla\cdot\mathbf{u}_N\|\|_\infty$	Ratio
16x64	3.04x10^{-9}	
32x128	1.93x10^{-9}	1.57
64x256	1.57x10^{-11}	123.04

Table 6.4: Resolution study of the error in mass conservation for the $B_y = -20$ case

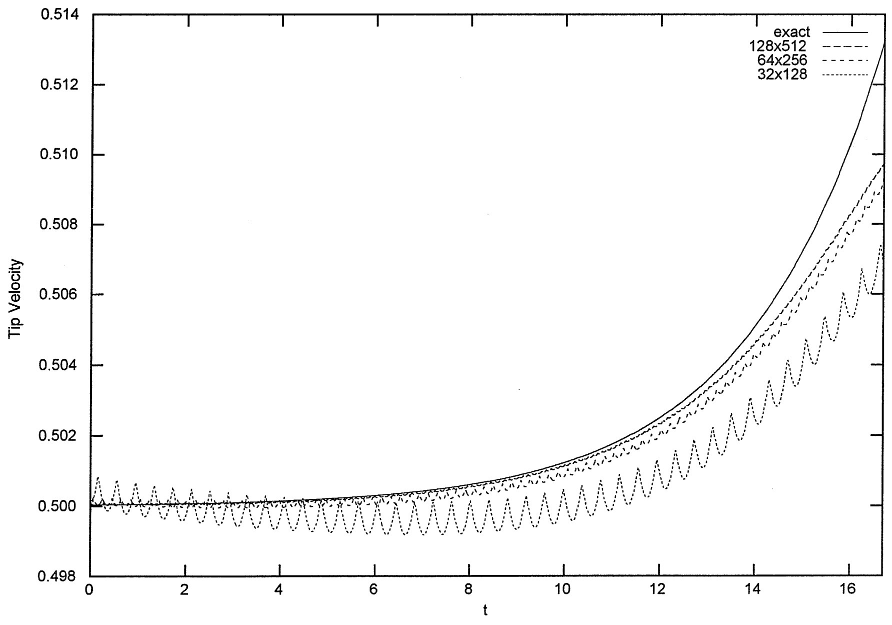

Figure 6.3: Tip velocity comparison between the numerical scheme at three different resolutions and linear theory for the $B_y = -20$ case

N_xxN_y	σ_N	$\left\|\sigma_N - \sigma\right\|_\infty$	Ratio
32x128	0.3516	2.97x10^{-3}	
64x256	0.3539	6.72x10^{-4}	4.43
128x512	0.3544	1.85x10^{-4}	3.63

Table 6.5: Resolution study of the growth rate for the $B_y = -20$ case

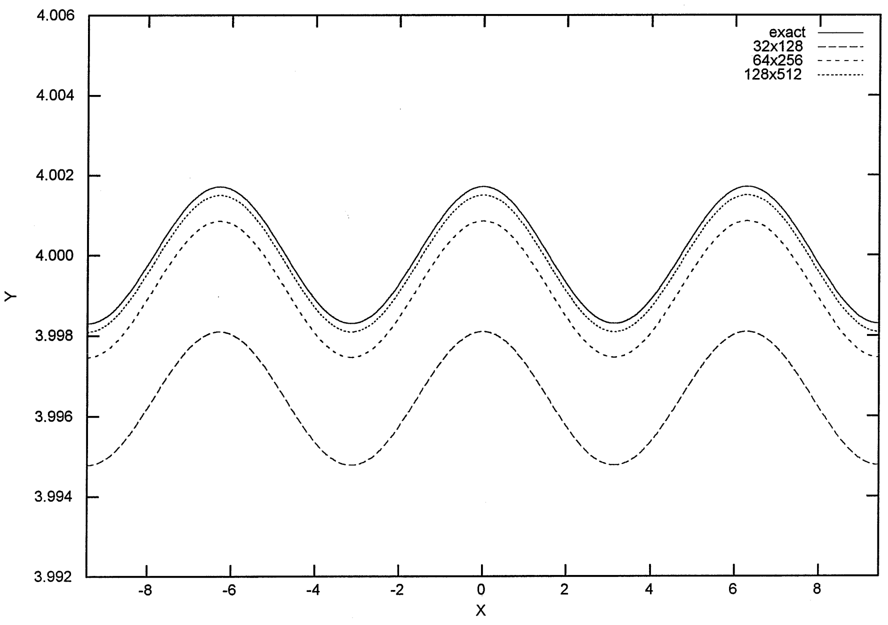

Figure 6.4: Resolution study demonstrating the second order rate of convergence of the interface position for the $B_y = -20$ case

of the divergence of the flow field. The convergence of the tip velocity, which we take to be the maximum of the normal velocity of the interface, is demonstrated in Fig. 6.3. Note that the numerical results converge at a second order rate to the linear theory up to about $t = 12$, after which the two approaches yield different results. Even though there is a discrepancy between linear theory and our results at latter times, the numerical solution is clearly still converging to a solution at a second order rate. We will see shortly that this late time disagreement is due to convection induced tip splitting which our simple linear theory can not predict. While the convergence of the growth rate is all but implied by the convergence of the tip velocity, it can also be measured directly using the procedure discussed in Chapter 4. The results of this measurement, which confirm second order convergence, are listed in Table 6.5. Note that the growth rate in the absence of convection for these parameter values is $\sigma = 0.47$. Thus, the predicted growth rate of $\sigma = 0.35$ for this problem demonstrates that convection can indeed make a significant impact. Finally, Fig. 6.4 demonstrates the convergence of the interface position to linear theory at $t = 8$.

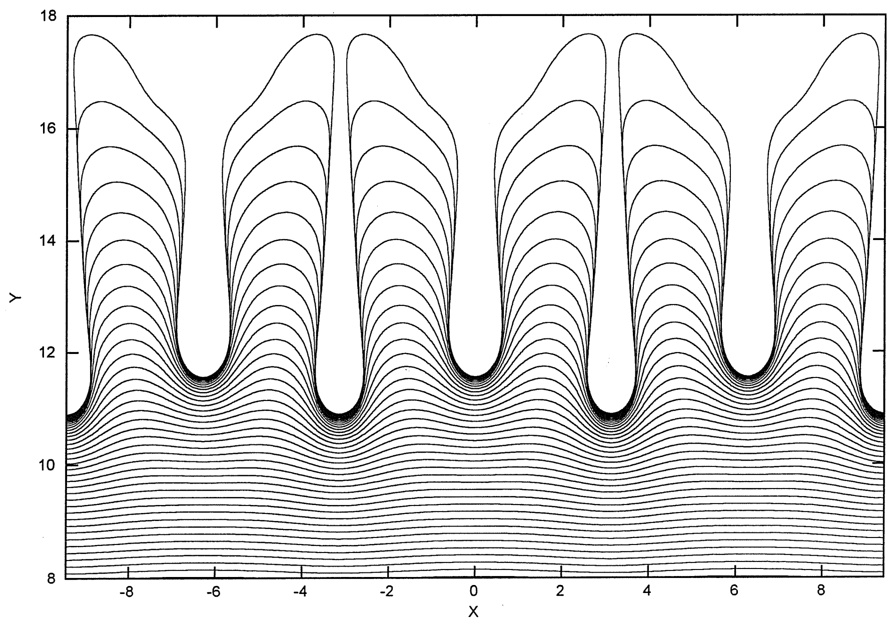

Figure 6.5: Evenly spaced snapshots ($\delta t = 0.25$) showing the evolution of the interface into the strongly non-linear regime for the $B_y = -20$ case

In Fig. 6.5 we have superimposed a series of snapshots of the interface at equally spaced intervals in time ($\delta t = 0.25$). Examining the initial shape of the interface (Fig. 6.4) we see that the dendritic finger undergoes a tip splitting very early in its evolution when the amplitude of the interface perturbation is still quite small. We shall see later that this behavior is a direct result of the bulk transport of energy in the liquid by natural convection.

In Figures 6.6-6.14 we examine the flow field (via the streamlines) near the interface at several different non-evenly spaced points in time. Note that the spacing of the stream lines in these figures is such that the interface, $\psi = 0$, lies halfway between the first positive (solid line) and negative (dashed line) stream line. Thus, although the interface is included for reference in each figure, care must be taken when using the distance between the interface and the closest streamline to infer information about the flow field.

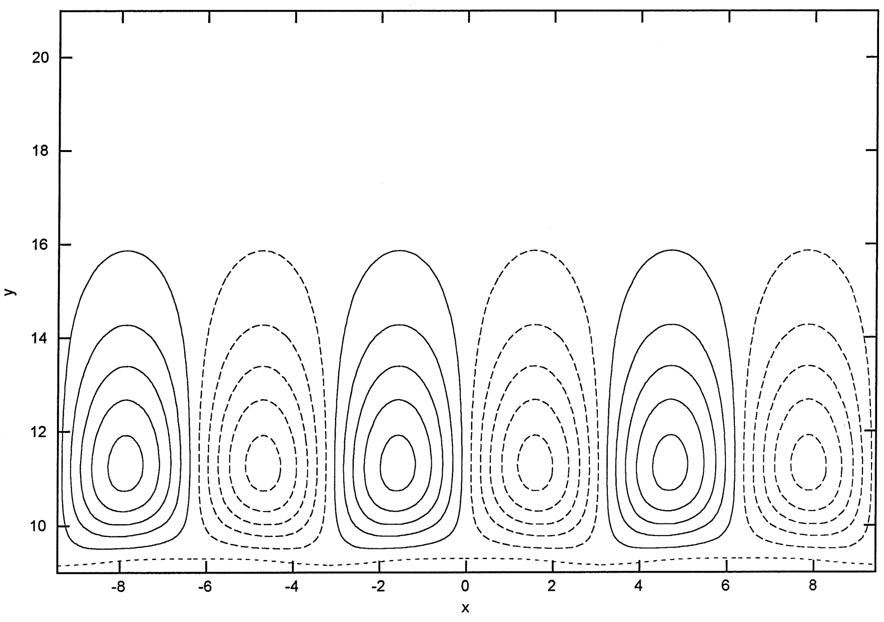

Figure 6.6: Snapshot showing the clockwise (dashed) and anti-clockwise (solid) vorticies above the interface before it becomes significantly perturbed

Figure 6.6 shows the flow field before the interface has been significantly perturbed. Note that the dendritic finger has been flattened in this figure. The solid lines in

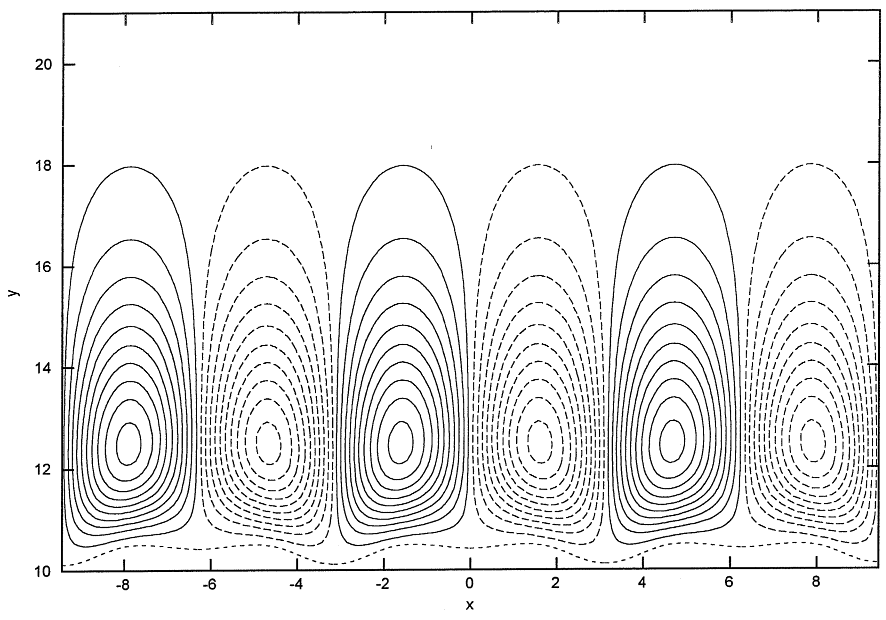

Figure 6.7: Snapshot of the streamlines and interface showing the initial occurrence of tip splitting

Figures 6.6-6.14 correspond to positive values of the stream function, while the dashed lines correspond to negative values. This implies that the flow in Figures 6.6-6.7 sweeps fluid from the valleys of the interface to the tip and then out into the bulk of the liquid. This implies that the fluid which reaches the tip has already been warmed by the interface (recall the bulk liquid is undercooled) before it ever reaches the tip. This decreases the temperature difference between the tip and the liquid immediately surrounding it, slowing its growth and causing the flatting, Fig. 6.6, and splitting, Fig. 6.6, of the tip quite early in its evolution.

Figures 6.8-6.11 show the formation and evolution of a pair of secondary vorticies formed in the valley created by the splitting of the initial dendritic finger. Note the decay of the initial vorticites and growth of the new vortices with time as the no-slip interface extends further out into the bulk liquid. Finally, in Figures 6.12-6.14 we see that the initial vorticies have been eradicated and the new vortex pair rotates in the opposite direction.

We now examine what happens if we "flip" the above problem over. That is, we

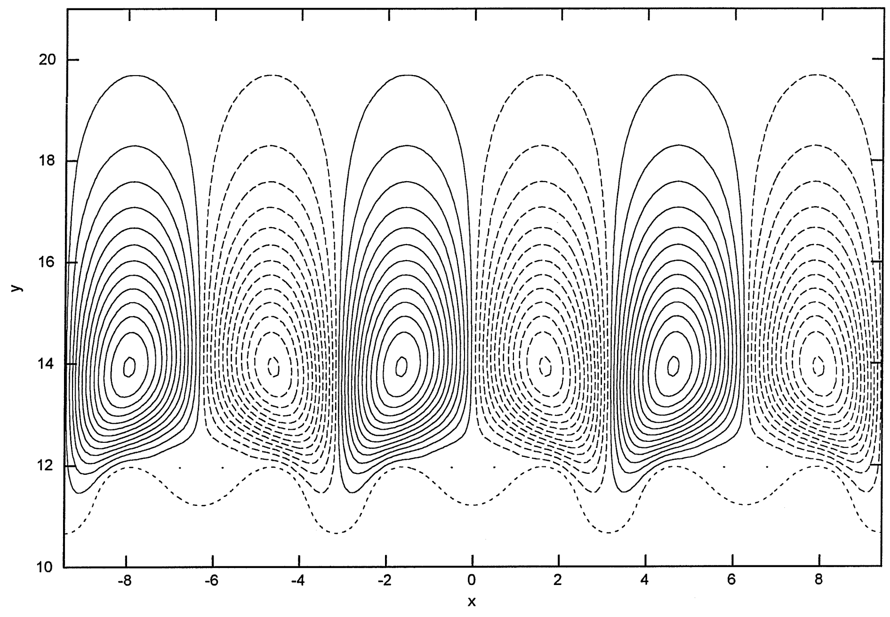

Figure 6.8: Snapshot of the streamlines showing the initial formation of secondary vorticities near the interface

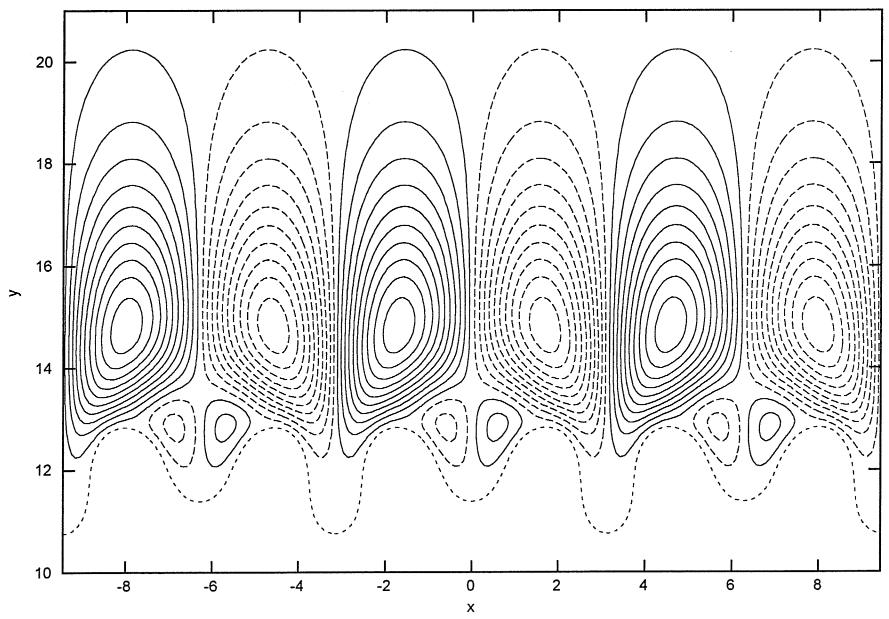

Figure 6.9: Snapshot of the streamlines showing the growth of the secondary vorticities near the interface

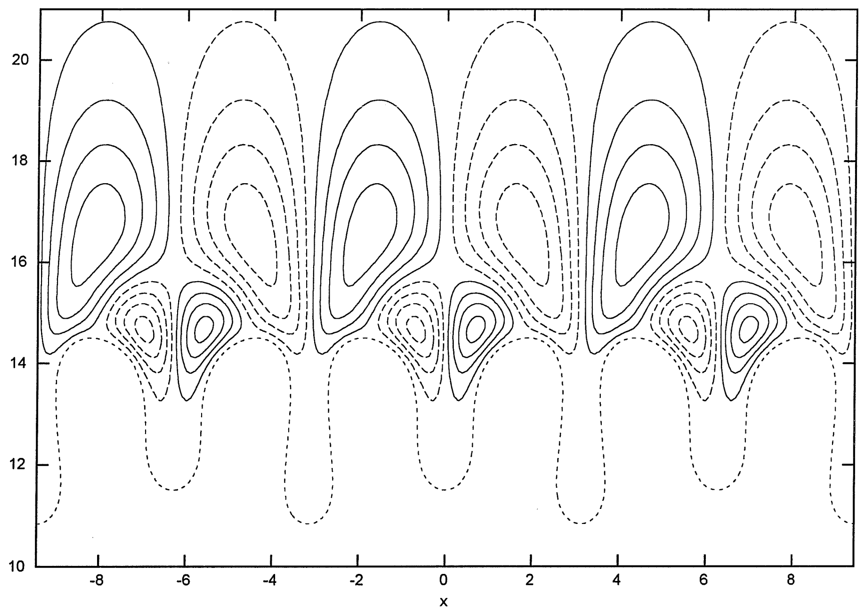

Figure 6.10: Snapshot of the streamlines showing the erosion of the primary vorticies and growth of the secondary vorticities

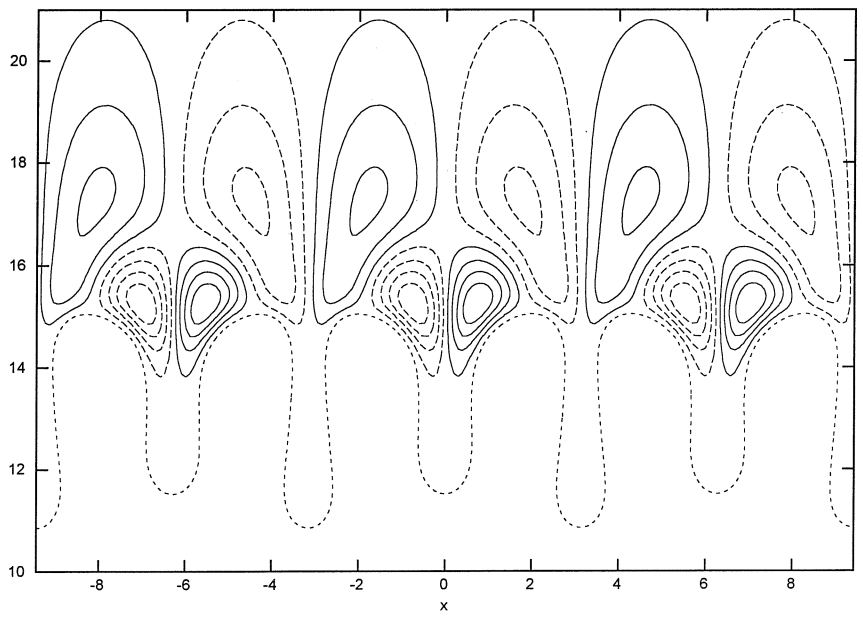

Figure 6.11: Snapshot of the streamlines showing the continued erosion of the primary vorticies and further amplification of the secondary vorticities

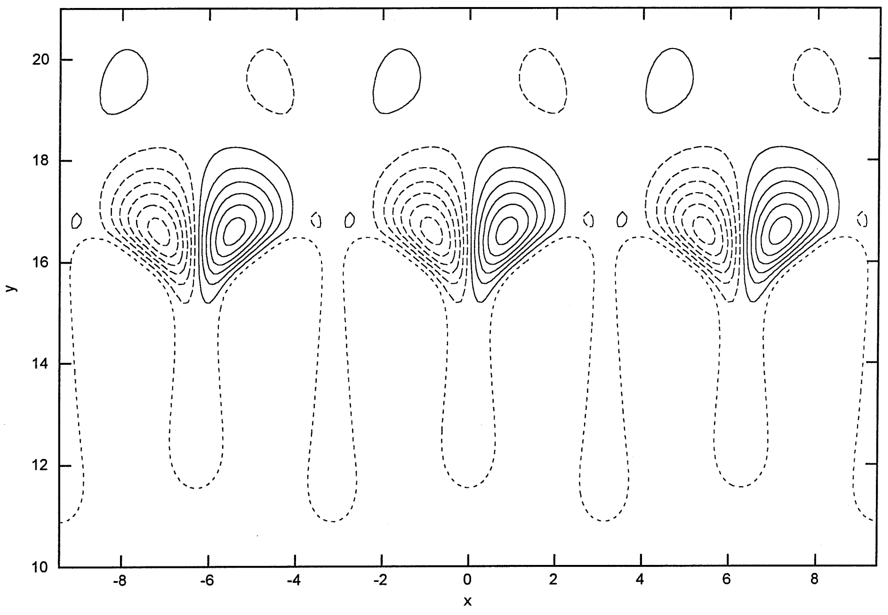

Figure 6.12: Snapshot of the streamlines showing the final elimination of the old primary vortices by the previously small secondary vorticies

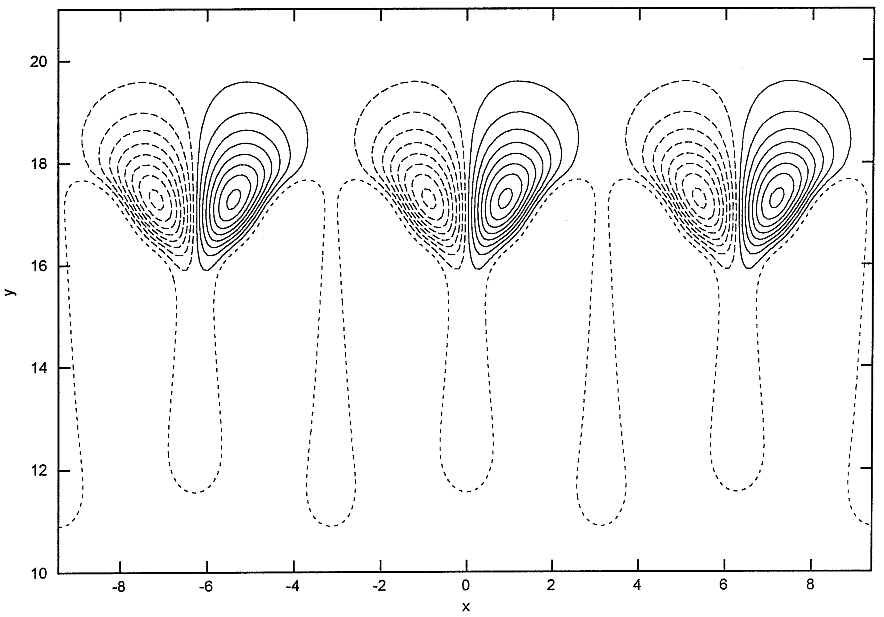

Figure 6.13: Snapshot of the streamlines showing the growth of the new primary vortices

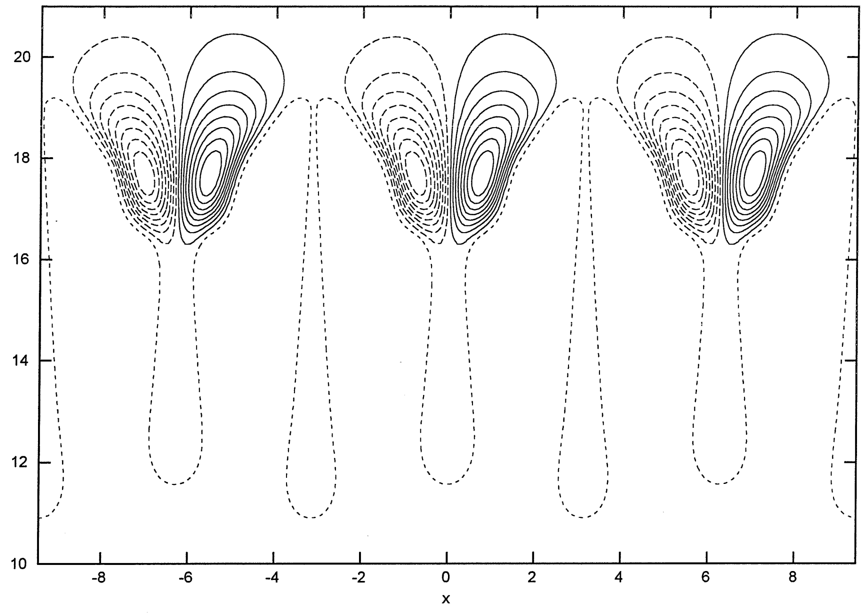

Figure 6.14: Final snapshot of the streamlines showing the continued evolution of the new primary vortices

take $B_y = +20$ which corresponds to having the solid grow in the opposite direction with respect to gravity. This reverses the buoyant forces acting on the system and, as we shall see, has a dramatic influence on the evolution of interface.

We will forego an exhaustive validation for this problem. As we saw in the $B_y = -20$ case, the results indicate a universal second order rate of convergence. It is interesting, however, to compare the convergence of the tip velocity for this case with what we found using $B_y = -20$. Comparing Figures 6.3 and 6.15, we see that there is better agreement between our numerical simulations and linear theory at larger interface perturbations for the $B_y = +20$ case. The interface in the $B_y = -20$ case quickly exhibited non-linear behavior that is not modeled by our simple linear theory. Such non-linear behavior does not occur until later in the evolution of the dendrite for the $B_y = +20$ case, so better agreement with linear theory is to be expected. It is also interesting to compare the influence of convection on the growth rate for the two different cases. Recall that the growth rate for the $B_y = -20$ case was approximately $\sigma = 0.35$. In contrast, the growth rate for the $B_y = +20$ is approximately $\sigma = 0.50$

N_xxN_y	σ_N	$\|\|\sigma_N - \sigma\|\|_\infty$	Ratio
32x128	0.5032	1.71x10^{-3}	
64x256	0.5020	5.10x10^{-4}	3.36
128x512	0.5016	1.33x10^{-4}	3.83

Table 6.6: Resolution study of the growth rate for the $B_y = 20$ case

(see Table 6.6) which is only a minor change from the $\sigma = 0.47$ predicted in the absence of convection.

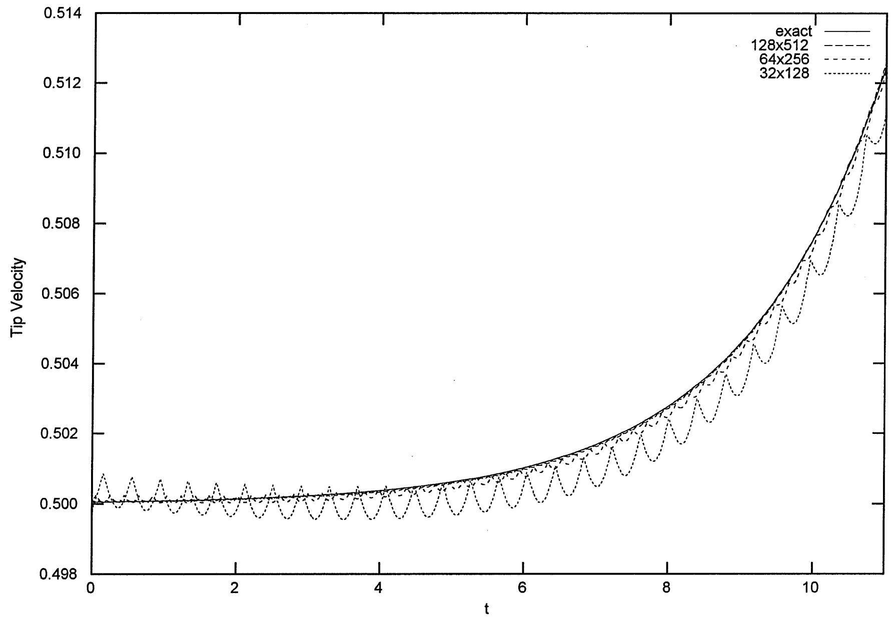

Figure 6.15: Tip velocity comparison between the numerical scheme at three different resolutions and linear theory for the $B_y = +20$ case

In Fig. 6.16 we have superimposed a series of snapshots of the interface at equally spaced intervals in time ($\delta t = 0.05$). Unlike in the previous example, we see that in this case the dendritic finger maintains its basic shape until it is quite deep into the non-linear regime. The difference between the previous case and this one is the initial flow field. In the $B_y = -20$ case, the valleys of the interface were supplied with the "cold" fluid from the bulk liquid. The fluid was then heated against the interface before it arrived at the dendrite tip. As we see in Fig. 6.17, however, the flow in the $B_y = +20$ case directs the "cold" fluid from the bulk liquid directly against the tip.

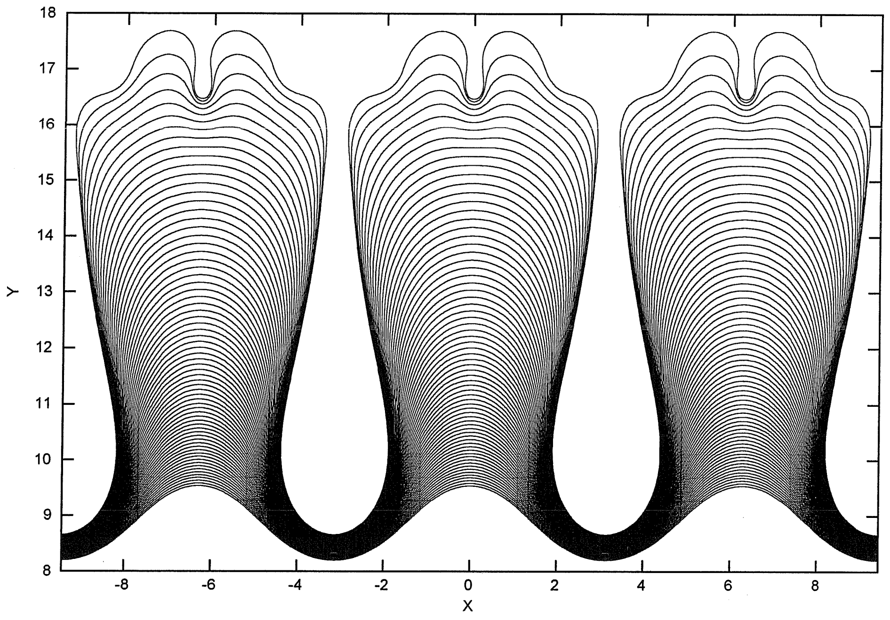

Figure 6.16: Evenly spaced snapshots ($\delta t = 0.05$) showing the evolution of the interface into the strongly non-linear regime for the $B_y = +20$ case

This increases the heat transfer from the tip and acts to counter the surface tension of the interface, hence, increasing the growth rate. Figures 6.17-6.19 demonstrate that this flow is maintained as the dendrite finger grows.

Figures 6.20-6.22 show the continual reduction of both the flow towards the tip and the flow in the valleys as the valley regions become enclosed by the growing solid region. Finally Fig. 6.23 shows that, even in this case, tip splitting does eventually occur. Also in Fig. 6.23 we see that flow in the valley region has been effectively extinguished. In fact, the lack of movement by the portion of the interface forming the valleys indicates that there is, at most, only a slight temperature gradient in that region. Thus, there is no net forcing available to drive the fluid trapped in the valleys, explaining its demise.

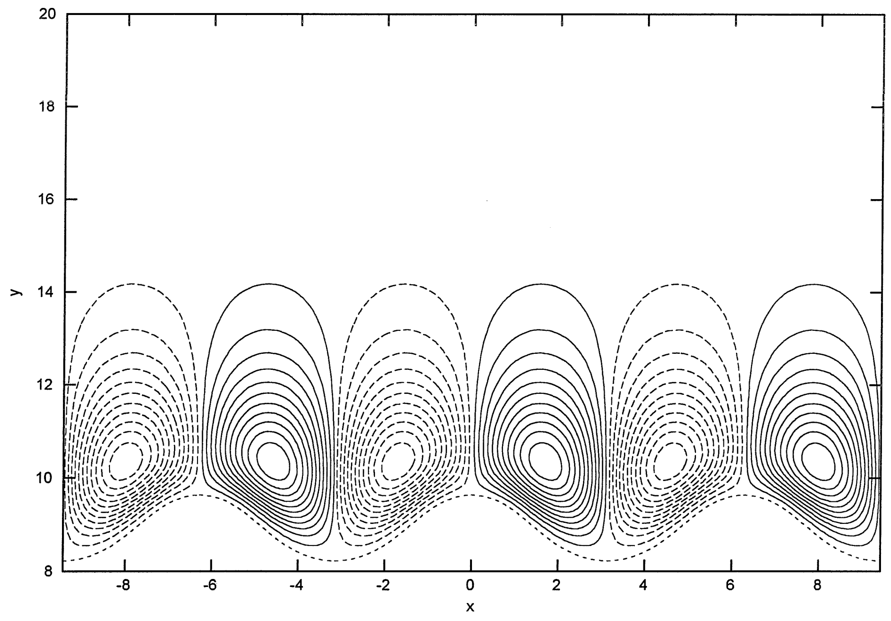

Figure 6.17: Early snapshot of the streamlines around the interface for the $B_y = +20$ case

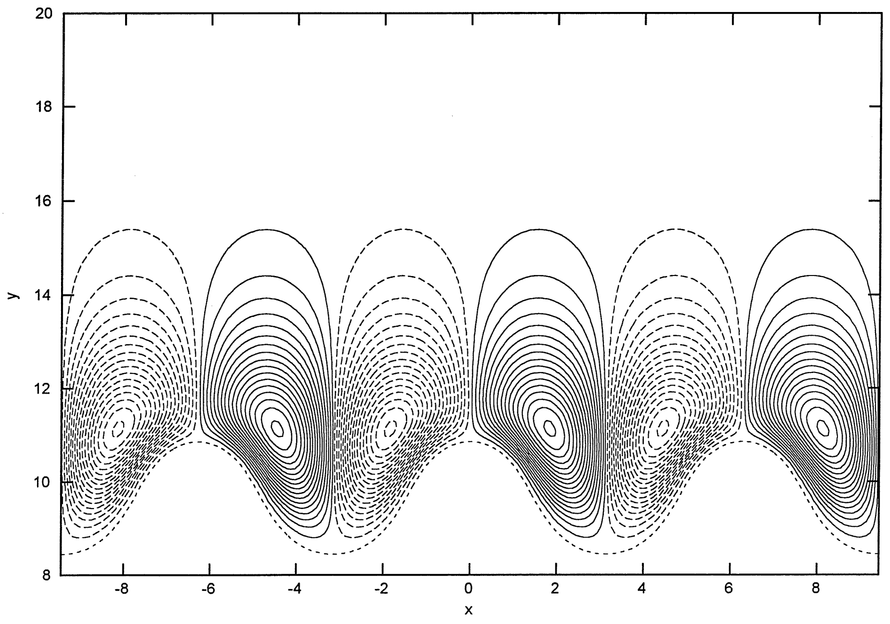

Figure 6.18: Snapshot of the streamlines around the interface for the $B_y = +20$ case

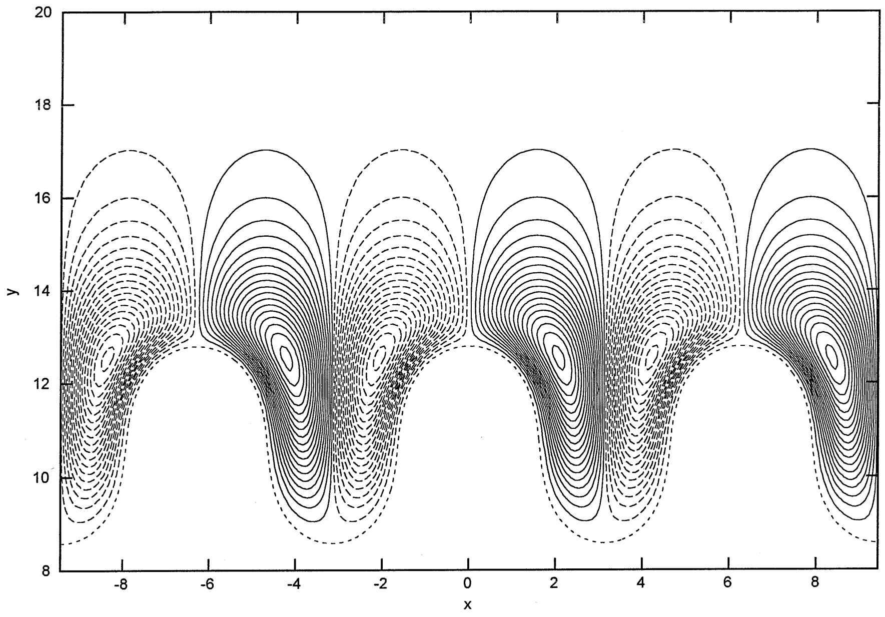

Figure 6.19: Another snapshot of the streamlines around the interface for the $B_y = +20$ case

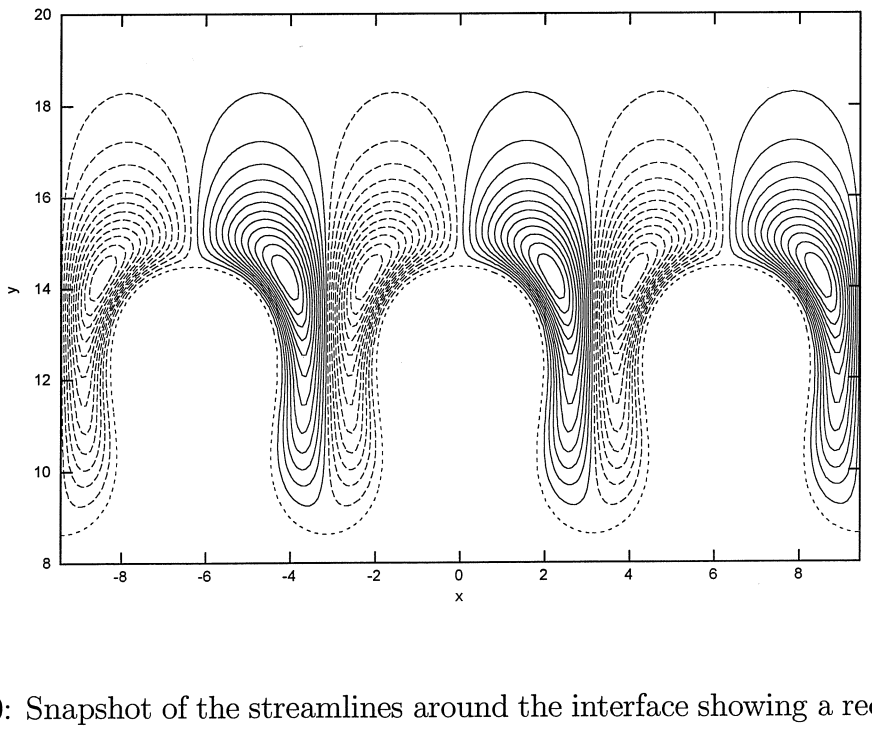

Figure 6.20: Snapshot of the streamlines around the interface showing a reduced flow towards the dendrite tip for the $B_y = +20$ case

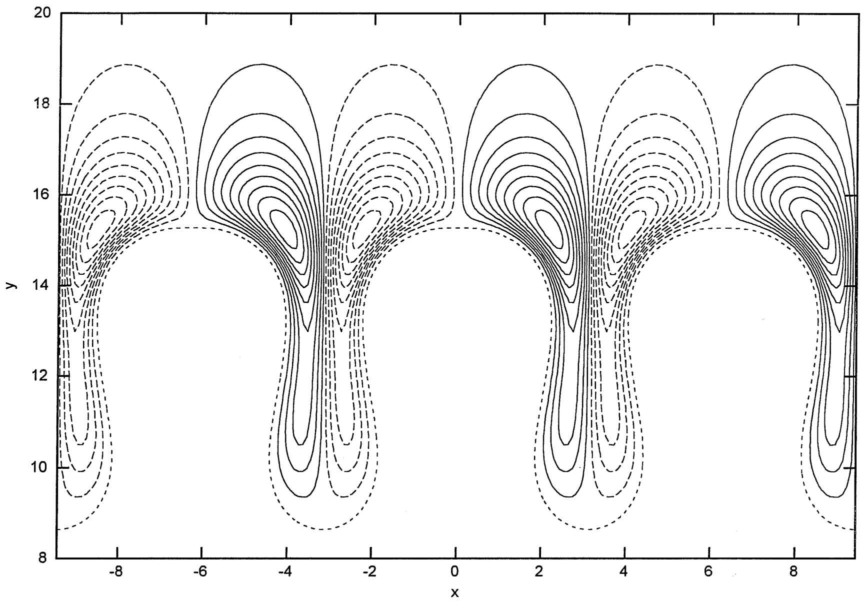

Figure 6.21: Snapshot of the streamlines around the interface showing a further reduced flow towards the dendrite tip for the $B_y = +20$ case

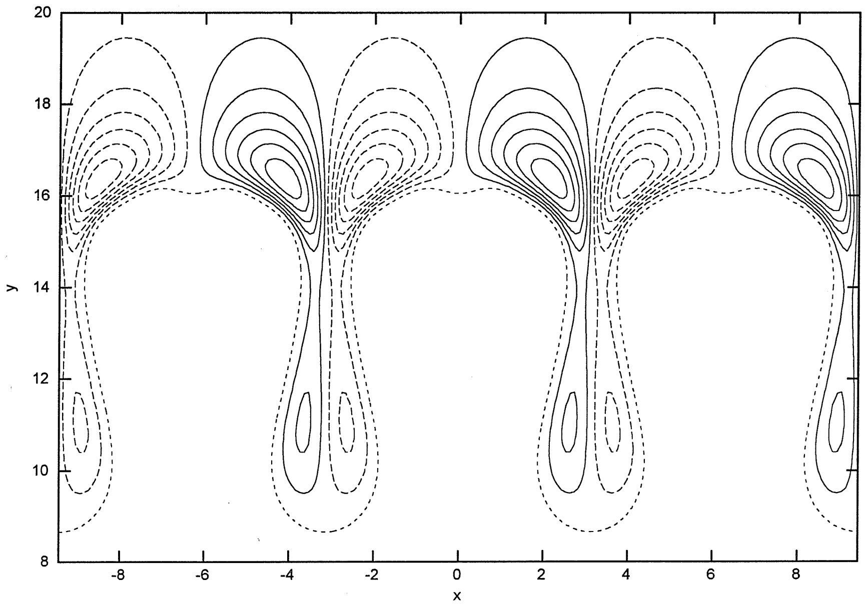

Figure 6.22: Snapshot of the streamlines around the interface showing the initial stages of tip splitting for the $B_y = +20$ case

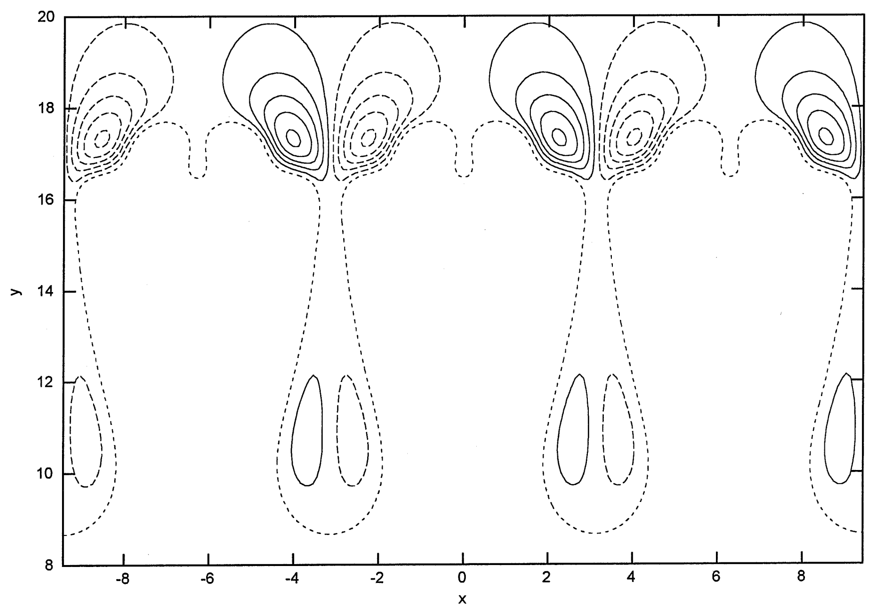

Figure 6.23: Snapshot of the streamlines around the interface showing a significantlly reduced flow in the highly non-linear regime for the $B_y = +20$ case

Chapter 7 Summary

7.1 Present work

The objective of the present work has been to develop a front tracking/fixed grid method capable of simulating dendritic solidification in the presence of natural convection. We have demonstrated that our method is second order accurate and capable of simulating experimentally observed phenomenon such as tip splitting and growth rate variation with system orientation. To our knowledge, this is the first time the fully coupled unsteady dendritic solidification problem has been successfully simulated. This is also the first time that the immersed interface method has been used to discretize the fully non-linear incompressible Navier-Stokes equations. In addition, several less significant contributions to the state of the art have been made. These are

1. The generalization of the immersed interface method to higher order stencils with the added benefit of improved coupling between interface jumps and nodal derivatives.

2. The development of an interpolation scheme that features a smoothly varying discretization error as the location of the interpolant moves through the grid.

3. The identification and removal of a degeneracy in the forcing system associated with the immersed interface/boundary method formulation of incompressible flow in irregular geometries.

4. The demonstration that the jump in the Laplacian of the pressure can be calculated directly in terms of the interfacial forcing for the full Navier-Stokes equations, without resorting to the interpolation of velocity derivatives off the computational mesh.

7.2 Recommendations for future work

The method we have developed is currently restricted to symmetric (equal material properties) problems. While this restriction is easily lifted using the theory developed in Chapter 3, the efficient solution of the resulting discrete systems remains an area of research. One approach would be to use a multigrid method like the one developed by Adams (see [2]). An alternate approach would be to use a different variant of the immersed interface method altogether. Viable options would be the methods described in [27] or [78]. It would be quite interesting to implement the different approaches and compare the results.

It would also be interesting to develop an adaptive mesh formulation of this method. The simulations we have performed have demonstrated that dendritic solidification typically requires the simultaneous resolution of both very large and very small scale features. Such problems are tailor-made for adaptive grid approaches. Our discretization method would allow an adaptive mesh algorithm to subdivide regions based solely on accuracy requirements with no need to distort the computational mesh to locally conform to the interface. An adaptive mesh would also allow the consideration of very large computational domains which would allow longer simulation of problems with low undercooling.

A method that implemented both of the above suggestions would yield a very powerful tool for the investigation of dendritic solidification.

Appendix A Non-dimensionalization of the system

The selection of scales in phase change problems with convection is a difficult issue. There are a wide variety of phenomena occurring on disparate length scales. A careful examination of the different issues involved can be found in [60]. We have followed the main recommendation of [60] and scaled the problem such that the interface velocity is $O(1)$. The dimensional quantities are related to the dimensionless variables by:

$$T_S = T_M + \Delta\theta_S, \tag{A.1}$$

$$T_L = T_M + \Delta\theta_L, \tag{A.2}$$

$$\mathbf{u} = U\tilde{\mathbf{u}}, \tag{A.3}$$

$$p = \rho_L U^2 \tilde{p}, \tag{A.4}$$

$$\kappa = \frac{1}{\lambda}\tilde{\kappa} \tag{A.5}$$

$$x = \lambda\tilde{x}, \tag{A.6}$$

$$y = \lambda\tilde{y}, \tag{A.7}$$

$$X = \lambda\tilde{X}, \tag{A.8}$$

$$Y = \lambda\tilde{Y}, \tag{A.9}$$

$$t = \tau\tilde{t}. \tag{A.10}$$

Dropping the tilda's and using

$$\tau = \frac{\lambda}{U}, \tag{A.11}$$

$$U = \frac{\alpha_L}{\lambda} St, \tag{A.12}$$

$$T_{ref} = T_M, \tag{A.13}$$

where

$$\alpha_S = \frac{K_S}{\rho_S c_S}, \tag{A.14}$$

$$\alpha_L = \frac{K_L}{\rho_L c_L}, \tag{A.15}$$

and the Prandtl number is defined as

$$Pr = \frac{\nu}{\alpha_L}, \tag{A.16}$$

the Stefan number is defined as

$$St = \frac{\rho_L c_L \Delta}{\rho_S L}, \tag{A.17}$$

and the Grashof number of defined as

$$Gr = \frac{\beta g \Delta \lambda^3}{\nu^2}, \tag{A.18}$$

we find the dimensionless system:

$$\frac{\partial \theta_S}{\partial t} = H_S \nabla^2 \theta_S, \tag{A.19}$$

in the solid,

$$\boldsymbol{\nabla} \cdot \mathbf{u} = 0, \tag{A.20}$$

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \boldsymbol{\nabla} \mathbf{u} = -\nabla p + D\nabla^2 \mathbf{u} + \mathbf{B}\theta_L, \tag{A.21}$$

and

$$\frac{\partial \theta_L}{\partial t} + \mathbf{u} \cdot \boldsymbol{\nabla} \theta_L = H_L \nabla^2 \theta_L, \tag{A.22}$$

in the liquid and, on the interface,

$$\mathbf{u} = \mathbf{0}, \tag{A.23}$$

$$\theta_S = \theta_L = -\varepsilon_c (\mathbf{n}) \kappa, \tag{A.24}$$

and

$$(h_S \nabla \theta_S - h_L \nabla \theta_L) \cdot \mathbf{n} = \left(\frac{\partial \mathbf{X}}{\partial t} \cdot \mathbf{n} \right). \tag{A.25}$$

The values of dimensionless material properties are given by

$$D = \left(\frac{Pr}{St} \right) \tag{A.26}$$

$$H_S = \left(\frac{\alpha_S}{\alpha_L} \right) \frac{1}{St}, \tag{A.27}$$

$$H_L = \frac{1}{St}, \tag{A.28}$$

$$h_S = \frac{K_S}{K_L}, \tag{A.29}$$

$$h_L = 1, \tag{A.30}$$

$$\varepsilon_c (\mathbf{n}) = \frac{\gamma (\mathbf{n})}{\lambda \Delta L}, \tag{A.31}$$

$$B_x = \zeta_x \left(\frac{Pr}{St} \right)^2 Gr, \tag{A.32}$$

$$B_y = \zeta_y \left(\frac{Pr}{St} \right)^2 Gr, \tag{A.33}$$

where $\mathbf{B} = (B_x, B_y)$ and $\mathbf{g} = g \left(\zeta_x, \zeta_y \right)$.

Appendix B Spatial discretization on a staggered grid

The placement of the various physical variables at differing points in space in a staggered grid simulation adds an extra layer of complexity to the calculations. In this section we outline the different formulas that must be employed.

B.1 Vertical momentum

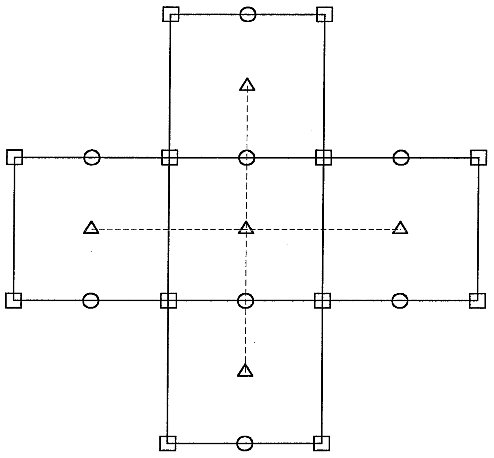

Figure B.1: Centered stencil for the vertical velocity nodes

The computational stencil for the v velocity is shown in Fig. B.1. The relative indices for the u velocity and pressure are shown in Fig. B.2. The scaled equation governing the vertical momentum is

$$\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} = -\frac{\partial p}{\partial y} + D\nabla^2 v + B_y\theta, \tag{B.1}$$

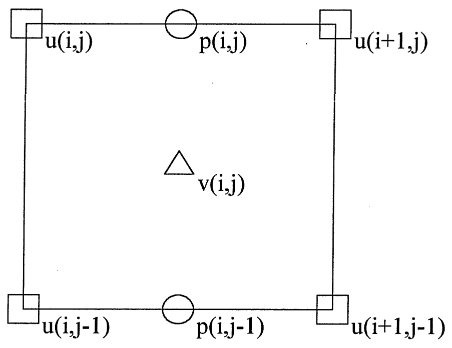

Figure B.2: Vertical velocity cell with relative indices for all variables

where the spatially discrete version (time dependence is implicit) of each term evaluated at the (i,j) v-node location $(1 \leq j \leq N_y)$ is

$$B_y\theta = \frac{B_y}{2}\left[\theta\left(i,j-1\right) + \theta\left(i,j\right)\right], \tag{B.2}$$

$$\frac{\partial p}{\partial y} = \frac{1}{\Delta x}\left[p\left(i,j\right) - p\left(i,j-1\right)\right], \tag{B.3}$$

$$u = \frac{1}{4}\left[u\left(i,j-1\right) + u\left(i,j\right) + u\left(i+1,j-1\right) + u\left(i+1,j\right)\right], \tag{B.4}$$

$$\nabla^2 v = \frac{1}{\Delta x^2}\left[v\left(i+1,j\right) + v\left(i-1,j\right) + v\left(i,j+1\right) + v\left(i,j-1\right) - 4v\left(i,j\right)\right], \tag{B.5}$$

$$\frac{\partial v}{\partial x} = \frac{1}{2\Delta x}\left[v\left(i+1,j\right) - v\left(i-1,j\right)\right], \tag{B.6}$$

$$\frac{\partial v}{\partial y} = \frac{1}{2\Delta x}\left[v\left(i,j+1\right) - v\left(i,j-1\right)\right]. \tag{B.7}$$

Our computational domain is periodic in the horizontal direction and the velocities are specified at the top and bottom of the domain. Thus the values of $v\left(i,0\right)$ and $v\left(i,N_y+1\right)$ are known. Because the position of the upper and lower boundaries coincides with the position of v-nodes, no special stencils are required near the boundaries. The other variables, however, are not evaluated at nodes exactly on the top and bottom of the computational domain. The remaining variables, therefore, will all require modified stencils for the nodes adjacent to the top and bottom boundaries.

B.2 Horizontal momentum

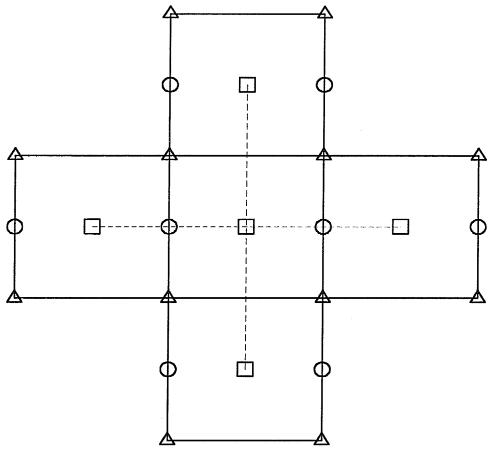

Figure B.3: Centered stencil for the horizontal velocity nodes

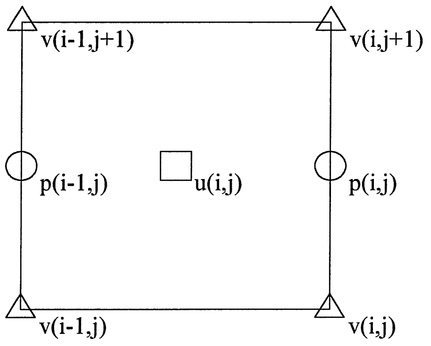

Figure B.4: Horizontal velocity cell with relative indices for all variables

The computational stencil for the u velocity is shown in Fig. B.3. The relative indices for the v velocity and pressure are shown in Fig. B.4. The scaled equation governing the horizontal momentum is

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} = -\frac{\partial p}{\partial x} + D\nabla^2 u + B_x\theta \tag{B.8}$$

where the spatially discrete version (time dependence is implicit) of each term evaluated at an (i,j) interior $(0 < j < N_y)$ u-node location is

$$B_x\theta = \frac{B_x}{2}\left[\theta\left(i-1,j\right)+\theta\left(i,j\right)\right], \tag{B.9}$$

$$\frac{\partial p}{\partial x} = \frac{1}{\Delta x}\left[p\left(i,j\right)-p\left(i-1,j\right)\right], \tag{B.10}$$

$$v = \frac{1}{4}\left[v\left(i-1,j\right)+v\left(i,j\right)+v\left(i-1,j+1\right)+v\left(i,j+1\right)\right], \tag{B.11}$$

$$\nabla^2 u = \frac{1}{\Delta x^2}\left[u\left(i+1,j\right)+u\left(i-1,j\right)+u\left(i,j+1\right)+u\left(i,j-1\right)-4u\left(i,j\right)\right], \tag{B.12}$$

$$\frac{\partial u}{\partial x} = \frac{1}{2\Delta x}\left[u\left(i+1,j\right)-u\left(i-1,j\right)\right], \tag{B.13}$$

$$\frac{\partial u}{\partial y} = \frac{1}{2\Delta x}\left[u\left(i,j+1\right)-u\left(i,j-1\right)\right]. \tag{B.14}$$

At the lower $(j=0)$ and upper $(j=N_y)$ boundaries the stencils for the derivatives in the vertical direction must be modified. The velocity is specified at the boundaries so, for $j=0$, the value $u\left(i,j-\frac{1}{2}\right)$ is known and

$$\nabla^2 u = \frac{1}{\Delta x^2}\left[u\left(i+1,j\right)+u\left(i-1,j\right)+\frac{4}{3}u\left(i,j+1\right)+\frac{8}{3}u\left(i,j-\frac{1}{2}\right)-6u\left(i,j\right)\right], \tag{B.15}$$

$$\frac{\partial u}{\partial y} = \frac{1}{\Delta x}\left[\frac{1}{3}u\left(i,j+1\right)+u\left(i,j\right)-\frac{4}{3}u\left(i,j-\frac{1}{2}\right)\right]. \tag{B.16}$$

Similarly, for $j=N_y$ the value $u\left(i,j+\frac{1}{2}\right)$ is known and

$$\nabla^2 u = \frac{1}{\Delta x^2}\left[u\left(i+1,j\right)+u\left(i-1,j\right)+\frac{8}{3}u\left(i,j+\frac{1}{2}\right)+\frac{4}{3}u\left(i,j-1\right)-6u\left(i,j\right)\right], \tag{B.17}$$

$$\frac{\partial u}{\partial y} = \frac{1}{\Delta x}\left[\frac{4}{3}u\left(i,j+\frac{1}{2}\right)-u\left(i,j\right)-\frac{1}{3}u\left(i,j-1\right)\right]. \tag{B.18}$$

B.3 Pressure

The computational stencil for the pressure is shown in Fig. B.5. The relative indices for the u and v velocities are shown in Fig. B.6. From the continuous equations

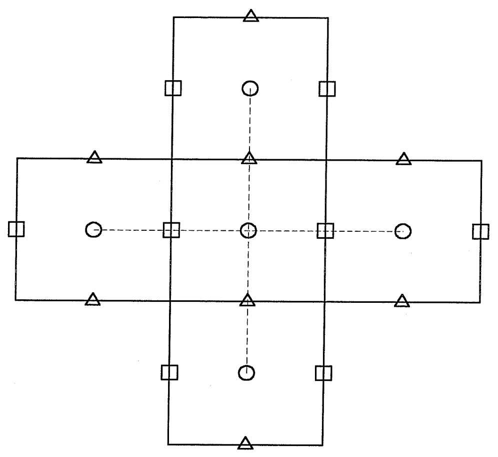

Figure B.5: Centered stencil for the pressure nodes

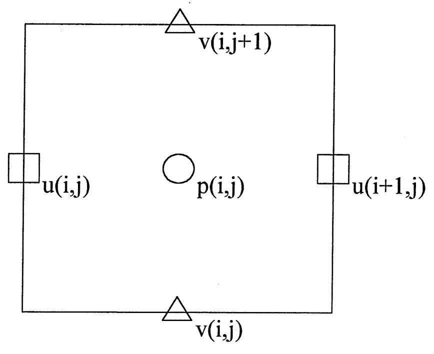

Figure B.6: Pressure cell with relative indices for all variables

it is possible to derive a Poisson equation governing the pressure (see [19])

$$\nabla^2 p = B_x \frac{\partial \theta}{\partial x} + B_y \frac{\partial \theta}{\partial y} - \left[\left(\frac{\partial u}{\partial x} \right)^2 + 2 \left(\frac{\partial u}{\partial y} \right) \left(\frac{\partial v}{\partial x} \right) + \left(\frac{\partial v}{\partial y} \right)^2 \right]. \tag{B.19}$$

While the right-hand side of the above is never directly discretized, the functional form of the forcing is required to accurately calculate the jumps across the interface.

The discrete equation governing the pressure is derived directly from the discrete continuity equation

$$\frac{u\,(i+1,j) - u\,(i,j)}{\Delta x} + \frac{v\,(i,j+1) - v\,(i,j)}{\Delta x} = 0, \tag{B.20}$$

and the expressions relating the physical and intermediate velocities

$$u\,(i,j) = \hat{u}\,(i,j) - \left(\frac{\Delta t}{\Delta x} \right) [p\,(i,j) - p\,(i-1,j)], \tag{B.21}$$

$$v\,(i,j) = \hat{v}\,(i,j) - \left(\frac{\Delta t}{\Delta x} \right) [p\,(i,j) - p\,(i,j-1)]. \tag{B.22}$$

Since our domain is periodic in the horizontal direction, the only special cases that occur are near the upper ($j = N_y$) and lower ($j = 0$) boundaries. For the interior pressure nodes ($0 < j < N_y$) substituting the above expressions for the physical velocity into the continuity equation and rearranging yields

$$\nabla^2 p = -\frac{1}{\Delta t} \left[\frac{\hat{u}\,(i+1,j) - \hat{u}\,(i,j)}{\Delta x} + \frac{\hat{v}\,(i,j+1) - \hat{v}\,(i,j)}{\Delta x} \right], \tag{B.23}$$

where the standard discretization for the Laplacian is obtained

$$\nabla^2 p = \frac{1}{\Delta x^2} [p\,(i+1,j) + p\,(i-1,j) + p\,(i,j+1) + p\,(i,j-1) - 4p\,(i,j)]. \tag{B.24}$$

At the lower ($j = 0$) boundary, the vertical velocity, $v\,(i,j)$, is specified so it is not necessary to substitute in an expression from this quantity. This modifies the equation governing the pressure which becomes ($j = 0$)

$$\nabla^2 p = -\frac{1}{\Delta t}\left[\frac{\hat{u}(i+1,j) - \hat{u}(i,j)}{\Delta x} + \frac{\hat{v}(i,j+1) - v(i,j)}{\Delta x}\right], \tag{B.25}$$

where a modified discretization for the Laplacian is obtained due to the presence of the lower boundary

$$\nabla^2 p = \frac{1}{\Delta x^2}\left[p(i+1,j) + p(i-1,j) + p(i,j+1) - 3p(i,j)\right]. \tag{B.26}$$

Similarly, at the upper boundary ($j = N_y$) the vertical velocity, $v(i,j+1)$, is also specified and the equation governing the pressure is ($j = N_y$)

$$\nabla^2 p = -\frac{1}{\Delta t}\left[\frac{\hat{u}(i+1,j) - \hat{u}(i,j)}{\Delta x} + \frac{v(i,j+1) - \hat{v}(i,j)}{\Delta x}\right], \tag{B.27}$$

where a modified discretization for the Laplacian is again obtained due to the presence of the upper boundary

$$\nabla^2 p = \frac{1}{\Delta x^2}\left[p(i+1,j) + p(i-1,j) + p(i,j-1) - 3p(i,j)\right]. \tag{B.28}$$

It is important to note that there is no need to supply boundary conditions on the pressure.

B.4 Temperature

The temperature and pressure nodes lie at exactly the same spatial locations and share a common indexing scheme; see Fig. B.6. The computational stencils associated with the temperature, however, differ somewhat from those used by the pressure (see Fig. B.7). In the liquid, the temperature satisfies

$$\frac{\partial \theta}{\partial t} + u\frac{\partial \theta}{\partial x} + v\frac{\partial \theta}{\partial y} = H\nabla^2\theta, \tag{B.29}$$

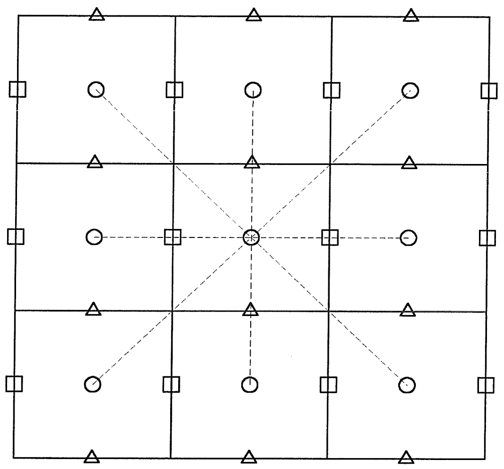

Figure B.7: Centered stencil for the temperature nodes

while in the solid, the temperature obeys

$$\frac{\partial \theta}{\partial t} = H\nabla^2\theta. \tag{B.30}$$

For locations, $0 < j < N_y$, the above quantities are calculated at the (i,j) temperature node using

$$u = \frac{1}{2}\left[u\left(i+1,j\right) + u\left(i,j\right)\right], \tag{B.31}$$

$$v = \frac{1}{2}\left[v\left(i,j+1\right) + v\left(i,j\right)\right], \tag{B.32}$$

$$\frac{\partial \theta}{\partial x} = \frac{1}{2\Delta x}\left[\theta\left(i+1,j\right) - \theta\left(i-1,j\right)\right], \tag{B.33}$$

$$\frac{\partial \theta}{\partial y} = \frac{1}{2\Delta x}\left[\theta\left(i,j+1\right) - \theta\left(i,j-1\right)\right], \tag{B.34}$$

$$\begin{aligned} \nabla^2\theta \quad = \quad & \frac{1}{6\Delta x^2}\left\{4\left[\theta\left(i+1,j\right) + \theta\left(i-1,j\right) + \theta\left(i,j+1\right) + \theta\left(i,j-1\right)\right] + \right. \\ & \theta\left(i+1,j+1\right) + \theta\left(i+1,j-1\right) + \theta\left(i-1,j+1\right) + \theta\left(i-1,j-1\right) \\ & \left. -20\theta\left(i,j\right)\right\} \end{aligned} \tag{B.35}$$

. Near the upper and lower vertical boundaries, the derivatives in the y-direction must be modified. For $j = 0$, the value of $\theta\left(i, j - \frac{1}{2}\right)$ is specified and

$$\nabla^2\theta = \frac{1}{\Delta x^2}\left[\theta\left(i+1,j\right) + \theta\left(i-1,j\right) + \frac{4}{3}\theta\left(i,j+1\right) + \frac{8}{3}\theta\left(i,j-\frac{1}{2}\right) - 6\theta\left(i,j\right)\right], \tag{B.36}$$

$$\frac{\partial\theta}{\partial y} = \frac{1}{\Delta x}\left[\frac{1}{3}\theta\left(i,j+1\right) + \theta\left(i,j\right) - \frac{4}{3}\theta\left(i,j-\frac{1}{2}\right)\right]. \tag{B.37}$$

Similarly, for $j = N_y$, the value of $\theta\left(i, j + \frac{1}{2}\right)$ is specified and

$$\nabla^2\theta = \frac{1}{\Delta x^2}\left[\theta\left(i+1,j\right) + \theta\left(i-1,j\right) + \frac{8}{3}\theta\left(i,j+\frac{1}{2}\right) + \frac{4}{3}\theta\left(i,j-1\right) - 6\theta\left(i,j\right)\right], \tag{B.38}$$

$$\frac{\partial\theta}{\partial y} = \frac{1}{\Delta x}\left[\frac{4}{3}\theta\left(i,j+\frac{1}{2}\right) - \theta\left(i,j\right) - \frac{1}{3}\theta\left(i,j-1\right)\right]. \tag{B.39}$$

Appendix C The jump conditions for the Navier-Stokes equations

The determination of the jump conditions for the fluid velocities and pressure is complicated by the lack of a separate evolution equation for the pressure. The jump conditions for Stokes flow were derived by LeVeque and Li in [39]. The calculation of the jump conditions for the full Navier-Stokes equations we present here is a generalization of that work.

For this derivation, it is convenient to use indicial notation and work with the Navier-Stokes equations in vector form. The scaled incompressible Navier-Stokes equations in vector form are

$$\frac{\partial u_i}{\partial x_i} = 0, \tag{C.1}$$

$$\frac{\partial u_i}{\partial t} + u_j \frac{\partial u_i}{\partial x_j} = -\frac{\partial p}{\partial x_i} + D \frac{\partial^2 u_i}{\partial x_j \partial x_j} + b_i + F_i, \tag{C.2}$$

where b_i is a body force due to buoyancy and F_i is a singular forcing term that exerts the same force on the fluid that a no-slip surface, Γ, would. In particular, if the interface is given by $X_i(q,t)$, where $q \in [0,1]$ is the parametrization variable, the interface forcing can be expressed as

$$F_i = \int_0^1 f_i(q,t)\, \delta\left(x_i - X_i(q,t)\right) dq. \tag{C.3}$$

The above forcing is singular since the integrand contains a two-dimensional delta function. It is probably worthwhile to note that although, in this work, the singular forcing on the interface models the effects of a solid surface embedded in the domain, we will treat the problem as if the entire domain is filled with a fluid. This is entirely consistent with our goal of determining the flow in the liquid region of the domain. Even though the computed flow will associate a fluid pressure and velocity with every point in the domain, liquid or solid, only the flow in the liquid region is utilized or even physically relevant.

Our goal is to determine the jumps in u_i and p and their derivatives across Γ due to application of the singular forcing on that surface. To this end we replace the continuity equation with the pressure Poisson equation. The pressure Poisson equation is obtained by taking the gradient of the momentum equation, (C.2), and using the continuity equation, (C.1), which, after some manipulation, yields

$$\frac{\partial^2 p}{\partial x_i \partial x_i} = \frac{\partial F_i}{\partial x_i} + \frac{\partial b_i}{\partial x_i} - \left(\frac{\partial u_j}{\partial x_i}\right)\left(\frac{\partial u_i}{\partial x_j}\right). \tag{C.4}$$

Now we need to introduce some notation. The surface along which the singular forcing acts, Γ, is a simple smooth curve that is periodic in the x_0 (horizontal) direction. The vectors n_i and τ_i are the normal and tangent to Γ, respectively. We will be considering a domain Ω_ε which contains a single period of Γ. The domain is bounded by $\Gamma_{+\varepsilon}$ and $\Gamma_{-\varepsilon}$ which are x_0 periodic simple smooth curves that do not touch Γ and have no point farther than $+\varepsilon$ or $-\varepsilon$ away, respectively, as measured along the normal to Γ (see Fig. C.1).

First we examine the continuity of the velocity field. Multiplying the continuity equation by w, an arbitrary x_0-periodic and twice continuously differentiable function, and integrating over Ω_ε yields, after some manipulation,

$$\int\limits_{\Omega_\varepsilon} \frac{\partial}{\partial x_i}\left(w u_i\right) dA - \int\limits_{\Omega_\varepsilon} u_i \frac{\partial w}{\partial x_i} dA = 0. \tag{C.5}$$

The velocity field is bounded so the second term will be $O\left(\varepsilon\right)$. Using the divergence theorem on the first term, we find

$$\int\limits_{\partial\Omega_\varepsilon} w u_i n_i ds = \int\limits_{\Gamma_{+\varepsilon}} w u_i n_i ds - \int\limits_{\Gamma_{-\varepsilon}} w u_i n_i ds = O\left(\varepsilon\right). \tag{C.6}$$

Taking the limit as $\varepsilon \to 0$ and using the fact that w is an arbitrary function, we

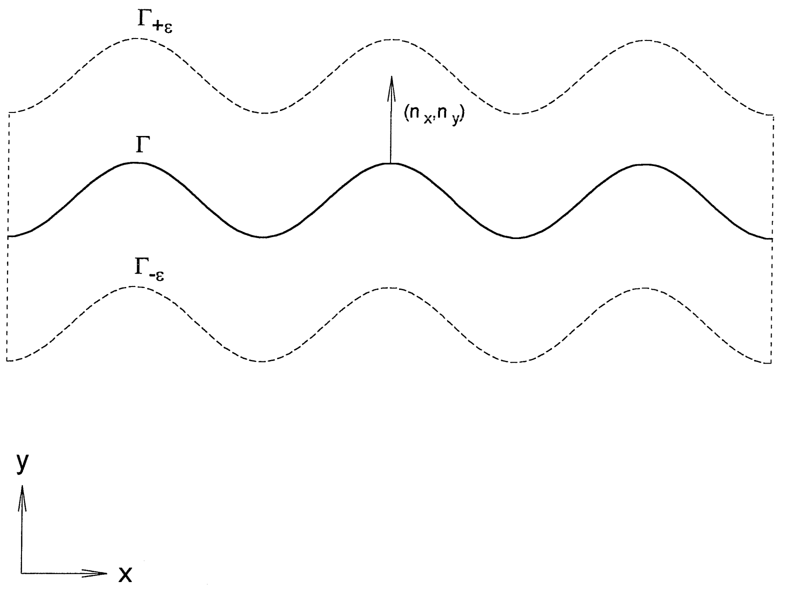

Figure C.1: Integration domain surrounding the interface

conclude

$$[u_i n_i] = 0, \tag{C.7}$$

where $[\cdot]$ denotes the jump in a quantity across the interface, X_i,

$$[a] = \lim_{\varepsilon \to 0^+} a\left(X_i + \varepsilon n_i\right) - \lim_{\varepsilon \to 0^+} a\left(X_i - \varepsilon n_i\right). \tag{C.8}$$

Assuming that the pressure is bounded at the interface, a quick examination of C.2 and C.3 reveals that the forcing is not singular enough to support jumps in the tangential velocity, thus

$$[u_i \tau_i] = 0. \tag{C.9}$$

Combining these two results, we conclude that the velocity field must be continuous on Γ

$$[u_i] = 0. \tag{C.10}$$

While it was necessary to assume that the pressure is bounded everywhere to derive the above, we will find that this assumption is consistent with the restrictions on the pressure we determine now.

The pressure Poisson equation contains terms that are quite singular. Not every term in (C.4), however, is singular enough to make a contribution to the pressure jump relations. Multiplying (C.4) by w and integrating over Ω_ε yields

$$\int_{\Omega_\varepsilon} w\frac{\partial^2 p}{\partial x_i \partial x_i} dA = \int_{\Omega_\varepsilon} w\frac{\partial F_i}{\partial x_i} dA + \int_{\Omega_\varepsilon} w\left\{\frac{\partial b_i}{\partial x_i} - \left(\frac{\partial u_j}{\partial x_i}\right)\left(\frac{\partial u_i}{\partial x_j}\right)\right\} dA. \tag{C.11}$$

The buoyancy term, which is proportional to temperature, is assumed to be continuous across the interface and we know from (C.10) that the velocity is as well. Therefore, the last integral above will vanish as $\varepsilon \to 0$ so we need only consider the first two terms:

$$\int_{\Omega_\varepsilon} w\frac{\partial^2 p}{\partial x_i \partial x_i} dA = \int_{\Omega_\varepsilon} w\frac{\partial F_i}{\partial x_i} dA + O(\varepsilon). \tag{C.12}$$

Using the identity

$$w\frac{\partial^2 p}{\partial x_i \partial x_i} = \frac{\partial}{\partial x_i}\left(w\frac{\partial p}{\partial x_i}\right) - \frac{\partial}{\partial x_i}\left(p\frac{\partial w}{\partial x_i}\right) + p\frac{\partial^2 w}{\partial x_i \partial x_i}, \tag{C.13}$$

and the divergence theorem, we find

$$\int_{\Omega_\varepsilon} w\frac{\partial^2 p}{\partial x_i \partial x_i} dA = \int_{\partial\Omega_\varepsilon} w\frac{\partial p}{\partial n} ds - \int_{\partial\Omega_\varepsilon} p\frac{\partial w}{\partial n} ds + \int_{\Omega_\varepsilon} p\frac{\partial^2 w}{\partial x_i \partial x_i} dA. \tag{C.14}$$

Noting that the last term of the above is $O(\varepsilon)$ because of the smoothness assumptions on w and taking the limit as $\varepsilon \to 0$ yields

$$\lim_{\varepsilon \to 0} \int_{\Omega_\varepsilon} w\frac{\partial^2 p}{\partial x_i \partial x_i} dA = \int_{\Gamma} w\left[\frac{\partial p}{\partial n}\right] ds - \int_{\Gamma} \frac{\partial w}{\partial n}[p]\, ds. \tag{C.15}$$

Now consider the integral on the right-hand side of (C.12). Using the identity

$$w\frac{\partial F_i}{\partial x_i} = \frac{\partial}{\partial x_i}(wF_i) - \frac{\partial w}{\partial x_i}F_i, \tag{C.16}$$

and the divergence theorem, we find

$$\int_{\Omega_\varepsilon} w\frac{\partial F_i}{\partial x_i} dA = \int_{\partial\Omega_\varepsilon} wn_iF_i ds - \int_{\Omega_\varepsilon} \frac{\partial w}{\partial x_i} F_i dA. \tag{C.17}$$

From the definition (C.3) we note that the above is equivalent to

$$\int_{\Omega_\varepsilon} w\frac{\partial F_i}{\partial x_i} dA = -\int_{\Gamma} \frac{\partial w}{\partial x_i} f_i ds, \tag{C.18}$$

where it is important to note that F_i, the integral of the interface forcing times a delta function, has been replaced by f_i, the magnitude of the interface forcing, in the integrand. Now combining (C.12), (C.15) and (C.18) and taking the limit as $\varepsilon \to 0$, we find

$$\int_{\Gamma} w\left[\frac{\partial p}{\partial n}\right] ds - \int_{\Gamma} \frac{\partial w}{\partial n}[p]\, ds = -\int_{\Gamma} \frac{\partial w}{\partial x_i} f_i ds, \tag{C.19}$$

which is not immediately useful because of the form the w dependence takes. It is possible to rescue the situation, however, by rewriting

$$\frac{\partial w}{\partial x_i} = \frac{\partial w}{\partial n} n_i + \frac{\partial w}{\partial s} \tau_i, \tag{C.20}$$

where s is arc length, which hold along Γ and implies

$$\int_{\Gamma} \frac{\partial w}{\partial x_i} f_i ds = \int_{\Gamma} \left(\frac{\partial w}{\partial n} n_i + \frac{\partial w}{\partial s}\tau_i\right) f_i ds. \tag{C.21}$$

Focusing our attention on the tangential derivative, we note that the identity

$$\frac{\partial}{\partial s}\left(w\tau_i f_i\right) = \frac{\partial w}{\partial s}\tau_i f_i + w\frac{\partial\left(\tau_i f_i\right)}{\partial s} \tag{C.22}$$

allows us to conclude

$$\int_{\Gamma} \left(\frac{\partial w}{\partial s}\tau_i f_i\right) ds = \int_{\Gamma} \frac{\partial}{\partial s}\left(w\tau_i f_i\right) ds - \int_{\Gamma} w\frac{\partial\left(\tau_i f_i\right)}{\partial s} ds. \tag{C.23}$$

The first integral on the right-hand side of the above vanishes because the integrand consists of x_0-periodic functions integrated over a period of an x_0-periodic curve. Using this fact and substituting (C.23) into (C.21), we find

$$\int_{\Gamma} \frac{\partial w}{\partial x_i} f_i ds = \int_{\Gamma} \left(\frac{\partial w}{\partial n} n_i f_i - w \frac{\partial (\tau_i f_i)}{\partial s} \right) ds. \tag{C.24}$$

Using this identity in (C.19) and collecting like terms yields

$$\int_{\Gamma} w \left\{ \left[\frac{\partial p}{\partial n} \right] - \frac{\partial (\tau_i f_i)}{\partial s} \right\} ds - \int_{\Gamma} \frac{\partial w}{\partial n} \left\{ [p] - n_i f_i \right\} ds = 0. \tag{C.25}$$

Because w is almost completely arbitrary, this allows us to conclude that

$$[p] = n_i f_i \tag{C.26}$$

and

$$\left[\frac{\partial p}{\partial n} \right] = \frac{\partial (\tau_i f_i)}{\partial s}. \tag{C.27}$$

Now that the jump conditions for the pressure have been established, the jump conditions for the velocity can be computed. Integrating (C.2) multiplied by w over Ω_ε and using the continuity of the velocity field and the buoyancy forcing across the interface, we derive

$$\int_{\Omega_\varepsilon} w \left(D \frac{\partial^2 u_i}{\partial x_j \partial x_j} \right) dA = \int_{\Omega_\varepsilon} w \frac{\partial p}{\partial x_i} dA - \int_{\Omega_\varepsilon} w F_i dA + O(\varepsilon). \tag{C.28}$$

Using the definition of F_i and arguments similar to those used for the pressure, we find

$$\int_{\partial\Omega_\varepsilon} w \left(D \frac{\partial u_i}{\partial n} \right) ds = \int_{\partial\Omega_\varepsilon} w (n_i p) ds - \int_{\partial\Omega_\varepsilon} w f_i ds + O(\varepsilon), \tag{C.29}$$

and taking the limit as ε goes to zero, we get

$$D \left[\frac{\partial u_i}{\partial n} \right] = [p] n_i - f_i. \tag{C.30}$$

Finally, using the fact that $[p] = f_j n_j$ and

$$n_k \left(f_j n_j n_k - f_k\right) = 0, \tag{C.31}$$

$$\tau_k \left(f_j n_j n_k - f_k\right) = -\tau_k f_k, \tag{C.32}$$

we see that the jump relation for the velocity field is

$$D \left[\frac{\partial u_i}{\partial n}\right] = -\left(\tau_k f_k\right) \tau_i. \tag{C.33}$$

While the above results are most easily derived in vector form, it is useful to summarize the results in terms of scalar variables as well. Thus, the Navier-Stokes equations,

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0, \tag{C.34}$$

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} = -\frac{\partial p}{\partial x} + D\nabla^2 u + B_x\theta, \tag{C.35}$$

$$\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} = -\frac{\partial p}{\partial y} + D\nabla^2 v + B_y\theta, \tag{C.36}$$

satisfy the following jump conditions for the horizontal velocity

$$[u] = 0, \tag{C.37}$$

$$\left[\frac{\partial u}{\partial n}\right] = -\frac{f_\tau \tau_x}{D}, \tag{C.38}$$

the vertical velocity

$$[v] = 0, \tag{C.39}$$

$$\left[\frac{\partial v}{\partial n}\right] = -\frac{f_\tau \tau_y}{D}, \tag{C.40}$$

and the pressure

$$[p] = f_n, \tag{C.41}$$

$$\left[\frac{\partial p}{\partial n}\right] = \frac{\partial f_\tau}{\partial s}. \tag{C.42}$$

With the above jump conditions known, we can compute the jumps in all the derivatives of the flow variables.

The pressure satisfies the Poisson equation given by

$$\nabla^2 p = F = B_x \frac{\partial \theta}{\partial x} + B_y \frac{\partial \theta}{\partial y} - \Lambda, \tag{C.43}$$

where

$$\Lambda = \left(\frac{\partial u}{\partial x}\right)^2 + 2\left(\frac{\partial u}{\partial y}\right)\left(\frac{\partial v}{\partial x}\right) + \left(\frac{\partial v}{\partial y}\right)^2. \tag{C.44}$$

Using the results derived for Poisson's equation in Chapter 3 we can easily find the jumps in the first derivatives

$$\begin{bmatrix} [p_x] \\ [p_y] \end{bmatrix} = \begin{bmatrix} \tau_x & n_x \\ \tau_y & n_y \end{bmatrix} \begin{bmatrix} \frac{\partial f_n}{\partial s} \\ \frac{\partial f_\tau}{\partial s} \end{bmatrix}. \tag{C.45}$$

The jumps in the second derivatives are given by

$$\begin{bmatrix} [p_{xx}] \\ [p_{xy}] \\ [p_{yy}] \end{bmatrix} = \begin{bmatrix} \tau_x & -\tau_y & \tau_y^2 \\ \tau_y & \tau_x & -\tau_x\tau_y \\ -\tau_x & \tau_y & \tau_x^2 \end{bmatrix} \begin{bmatrix} \frac{\partial [p_x]}{\partial s} \\ \frac{\partial [p_y]}{\partial s} \\ [F] \end{bmatrix}, \tag{C.46}$$

which requires that the jump in the forcing, $[F]$, be known. This jump is given by

$$[F] = B_x [\theta_x] + B_y [\theta_y] - [\Lambda], \tag{C.47}$$

where

$$[\Lambda] = \left[\left(u_x^+\right)^2 - \left(u_x^-\right)^2\right] + 2\left[\left(u_y^+ v_x^+\right) - \left(u_y^- v_x^-\right)\right] + \left[\left(v_y^+\right)^2 - \left(v_y^-\right)^2\right], \tag{C.48}$$

and

$$u_x^+ = \lim_{\varepsilon \to 0^+} u_x \left(X + \varepsilon n_x, Y + \varepsilon n_y\right), \tag{C.49}$$

$$u_x^- = \lim_{\varepsilon \to 0^+} u_x \left(X - \varepsilon n_x, Y - \varepsilon n_y \right), \tag{C.50}$$

etc. While the jumps in the temperature, $[\theta_x]$ and $[\theta_y]$, are known, it is not clear that $[\Lambda]$ can be calculated solely from our knowledge of the jumps in the velocity. One possible way to compute $[\Lambda]$ would be to interpolate the first derivatives of the velocity field off the computational grid points. This is somewhat cumbersome, however, and, for our problem at least, unnecessary.

The easiest way to compute $[\Lambda]$ would be to find some way of calculating the values of all the first derivatives of velocity. Following this approach, we would need to compute the values of eight quantities: u_x^+, u_x^-, u_y^+, u_y^-, v_x^+, v_x^-, v_y^+ and v_y^-. This is a lot of unknowns but, fortunately, there is a substantial amount of information available. For our problem, the velocity at the interface is specified

$$u\left(X, Y, t\right) = U\left(s, t\right), \tag{C.51}$$

$$v\left(X, Y, t\right) = V\left(s, t\right), \tag{C.52}$$

and, as we found above, continuous. Thus, differentiating the above and evaluating it on each side of the interface yields four equations:

$$\tau_x u_x^+ + \tau_y u_y^+ = \frac{\partial U}{\partial s}, \tag{C.53}$$

$$\tau_x u_x^- + \tau_y u_y^- = \frac{\partial U}{\partial s}, \tag{C.54}$$

$$\tau_x v_x^+ + \tau_y v_y^+ = \frac{\partial V}{\partial s}, \tag{C.55}$$

and

$$\tau_x v_x^- + \tau_y v_y^- = \frac{\partial V}{\partial s}. \tag{C.56}$$

We also know that the flow must satisfy the continuity equation, so evaluating that on both sides of the interface supplies two further equations

$$u_x^+ + v_y^+ = 0, \tag{C.57}$$

$$u_x^- + v_y^- = 0, \tag{C.58}$$

bringing our total to six. It is tempting, at this point, to use the jumps in the normal derivatives across the interface

$$n_x \left(u_x^+ - u_x^-\right) + n_y \left(u_y^+ - u_y^-\right) = [u_n], \tag{C.59}$$

$$n_x \left(v_x^+ - v_x^-\right) + n_y \left(v_y^+ - v_y^-\right) = [v_n], \tag{C.60}$$

to close the system. Unfortunately, only one of these is linearly independent of the equations already present. Thus there is only enough information for one equation, which, in its most convenient form, comes from a linear combination of the above

$$\tau_x n_x \left(u_x^+ - u_x^-\right) + \tau_x n_y \left(u_y^+ - u_y^-\right) + \tau_y n_x \left(v_x^+ - v_x^-\right) + \tau_y n_y \left(v_y^+ - v_y^-\right) = -\frac{f_\tau}{D}. \tag{C.61}$$

Note that the right-hand side of the above follows from the definitions of $[u_n]$ and $[v_n]$. This leaves us with seven linearly independent equations in eight unknowns. The last equation is motivated only by our desire to close the system,

$$\tau_x n_x u_x^+ + \tau_x n_y u_y^+ + \tau_y n_x v_x^- + \tau_y n_y v_y^- = W, \tag{C.62}$$

where the value of W is unknown. The complete set of equations is

$$\begin{bmatrix} \tau_x & \tau_y & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \tau_x & \tau_y & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \tau_x & \tau_y & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \tau_x & \tau_y \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ \tau_x n_x & \tau_x n_y & -\tau_x n_x & -\tau_x n_y & \tau_y n_x & \tau_y n_y & -\tau_y n_x & -\tau_y n_y \\ \tau_x n_x & \tau_x n_y & 0 & 0 & 0 & 0 & \tau_y n_x & \tau_y n_y \end{bmatrix} \begin{bmatrix} u_x^+ \\ u_y^+ \\ u_x^- \\ u_y^- \\ v_x^+ \\ v_y^+ \\ v_x^- \\ v_y^- \end{bmatrix} = \begin{bmatrix} \frac{\partial U}{\partial s} \\ \frac{\partial U}{\partial s} \\ \frac{\partial V}{\partial s} \\ \frac{\partial V}{\partial s} \\ 0 \\ 0 \\ -\frac{f_\tau}{D} \\ W \end{bmatrix}, \tag{C.63}$$

which has a determinant of $\left(\tau_x^2 + \tau_y^2\right)^4 = 1$. Solving for the first derivatives yields a series of lengthy expressions that are dependent on all the terms found on the right-hand side, including W. Fortunately, we do not need to know the values of the first derivatives individually. Substituting the values obtained above into the definition of $[\Lambda]$ yields

$$[\Lambda] = -\frac{2f_\tau}{D}\left(n_x\frac{\partial U}{\partial s} + n_y\frac{\partial V}{\partial s}\right) \tag{C.64}$$

which is independent of the unknown quantity W.

The value of the $[F]$ can now be calculated without resorting to interpolation

$$[F] = B_x\left[\theta_x\right] + B_y\left[\theta_y\right] + \frac{2f_\tau}{D}\left(n_x\frac{\partial U}{\partial s} + n_y\frac{\partial V}{\partial s}\right). \tag{C.65}$$

In contrast to the pressure, the jumps in the derivatives of the velocities are fairly easy to compute. Formally at least, we can view the momentum equations as Poisson's equation which allows us to use the results derived in Chapter 3. Thus, rewriting the momentum equation in the horizontal direction as

$$\nabla^2 u = M = \frac{1}{D}\left[\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} + \frac{\partial p}{\partial x} - B_x\theta\right], \tag{C.66}$$

the jumps in the first derivatives of the horizontal velocity are

$$\left[u_x\right] = n_x\left[u_n\right], \tag{C.67}$$

$$\left[u_y\right] = n_y\left[u_n\right]. \tag{C.68}$$

The jumps in the second derivatives are given by

$$\begin{bmatrix} [u_{xx}] \\ [u_{xy}] \\ [u_{yy}] \end{bmatrix} = \begin{bmatrix} \tau_x & -\tau_y & \tau_y^2 \\ \tau_y & \tau_x & -\tau_x\tau_y \\ -\tau_x & \tau_y & \tau_x^2 \end{bmatrix} \begin{bmatrix} \frac{\partial[u_x]}{\partial s} \\ \frac{\partial[u_y]}{\partial s} \\ [M] \end{bmatrix}. \tag{C.69}$$

These can be calculated once the jump in the forcing,

$$[M] = \frac{1}{D}\left([u_t] + U\,[u_x] + V\,[u_y] + [p_x]\right), \tag{C.70}$$

the interface velocities $U\,(s,t)$ and $V\,(s,t)$, the jump in the time derivative

$$[u_t] = -\left(\frac{\partial X}{\partial t}\,[u_x] + \frac{\partial Y}{\partial t}\,[u_y]\right) \tag{C.71}$$

and the jump in the x-direction pressure derivative, $[p_x]$, are known.

Similarly, rewriting the momentum equation in the vertical direction as

$$\nabla^2 v = N = \frac{1}{D}\left[\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} + \frac{\partial p}{\partial y} - B_y\theta\right], \tag{C.72}$$

the jumps in the first derivatives of the vertical velocity are

$$[v_x] = n_x\,[v_n]\,, \tag{C.73}$$

$$[v_y] = n_y\,[v_n]\,, \tag{C.74}$$

while the jumps in the second derivatives are given by

$$\begin{bmatrix} [v_{xx}] \\ [v_{xy}] \\ [v_{yy}] \end{bmatrix} = \begin{bmatrix} \tau_x & -\tau_y & \tau_y^2 \\ \tau_y & \tau_x & -\tau_x\tau_y \\ -\tau_x & \tau_y & \tau_x^2 \end{bmatrix} \begin{bmatrix} \frac{\partial [v_x]}{\partial s} \\ \frac{\partial [v_y]}{\partial s} \\ [N] \end{bmatrix}. \tag{C.75}$$

These can be calculated once the jump in the forcing,

$$[N] = \frac{1}{D}\left([v_t] + U\,[v_x] + V\,[v_y] + [p_y]\right), \tag{C.76}$$

the interface velocities $U\,(s,t)$ and $V\,(s,t)$, the jump in the time derivative

$$[v_t] = -\left(\frac{\partial X}{\partial t}\,[v_x] + \frac{\partial Y}{\partial t}\,[v_y]\right) \tag{C.77}$$

and the jump in the y-direction pressure derivative, $[p_y]$, are known.

Bibliography

[1] M. Abramowitz and I. A. Stegun, *Handbook of Mathematical Functions*, Dover, New York, 1965.

[2] L. M. Adams, *A multigrid algorithm for immersed interface problems*, in Proceedings, Copper Mountain Multigrid Conference, 1995.

[3] R. Almgren, *Variational algorithms and pattern formation in dendritic solidification*, J. Comput. Phys., **106**, pp. 337-354 (1993).

[4] E. Bänsch and A. Schmidt, *A finite element method for dendritic growth*, in Computational Crystal Growers Workshop, J. E. Taylor ed., AMS Selected Lectures in Mathematics (Amer. Math. Soc., Providence, 1992).

[5] R. Barrett, M. Berry, T. F. Chan, J. Demmel, J. M. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine and H. Van der Vorst, *Templates for the solution of linear systems: building blocks for iterative methods*, SIAM, 1994.

[6] C. Benard, D. Gobin and A. Zanoli, *Moving boundary problem: heat conduction in the solid phase of a phase-change material during melting driven by natural convection in the liquid*, Int. J. Heat Mass Trans., **29**(11), pp. 1669-1681 (1986).

[7] K. Brattkus and D. I. Meiron, *Numerical simulations of unsteady crystal growth*, SIAM J. Appl Math, **52**(5), pp. 1303-1320 (1992).

[8] R. J. Braun, B. T. Murray and J. Soto Jr., *Adaptive finite-difference computations of dendritic growth using a phase-field model*, Modeling Simul. Mater. Sci. Eng., **5**, pp. 365-380 (1997).

[9] C. G. Broyden, *A Class of Methods for Solving Nonlinear Simultaneous Equations*, Math. Comp., **19**, pp. 577-593 (1965).

[10] A. Buchholz and S. Engler, *The influence of forced convection on solidification interfaces*, Comp. Mat. Sci., **7**, pp. 221-227 (1996).

[11] G. Caginalp and E. Socolovsky, *Phase field computations of single-needle crystals, crystal growth, and motion by mean curvature*, SIAM J. Sci. Comput., **15**(1), pp. 106-126 (1994).

[12] Y. Cao and A. Faghri, *A numerical analysis of phase-change problems including natural convection*, J. Heat Trans.-T. ASME, **112**(3), pp. 812-816 (1990).

[13] S. Chen, B. Merriman, S. Osher, and P. Smereka, *A simple level set method for solving Stefan problems*, J. Comput. Phys., **135**, pp. 8-29 (1997).

[14] A. J. Chorin, *Numerical Solution of the Navier-Stokes Equations*, Math. Comp., **22**, pp. 745-762 (1968).

[15] A. J. Chorin, *Curvature and Solidification*, J. Comput. Phys., **57**, 472-490 (1985).

[16] M. S. Engelman and M. A. Jamnia, *Transient flow past a circular cylinder - a benchmark solution*, Int. J. Num. Meth. Fluids, **11**(7), pp. 985-1000 (1990).

[17] M. Fabbri and V. R. Voller, *The phase-field method in the sharp-interface limit: a comparison between model potentials*, J. Comput. Phys., **130**, pp. 256-265 (1997).

[18] G. J. Fix and J. T. Lin, *Numerical Simulations of nonlinear phase transitions-I. the isotropic case*, Nonlinear Anal. Theory, Methods Appl., **12**(8), pp. 811-823 (1988).

[19] C. A. J. Fletcher, *Computational techniques for fluid dynamics 2: Specific techniques for different flow categories*, Springer-Verlag, 1991.

[20] M. E. Glicksman, S. R. Coriell and G. B. McFadden, *Interaction of flows with the crystal-melt interface*, Ann. Rev. Fluid Mech., **18**, pp. 307-335 (1986).

[21] M. E. Glicksman and S. P. Marsh, *The Dendrite*, Handbook of Crystal Growth, D. T. J. Hurle, ed., Vol. 1, Elsevier Science Publishers, 1993, pp. 1077-1108.

[22] D. Goldstein, R. Handler and L. Sirovich, *Modeling a no-slip flow boundary with an external force field*, J. Comput. Phys., **105**, pp. 354-366 (1993).

[23] P. M. Gresho, S. T. Chan, R. L. Lee and C. D. Upson, *A modified finite element method for solving the time-dependent incompressible Navier-Stokes equations*, Int. J. Num. Meth. Fluids, **4**(7), pp. 619-640 (1984).

[24] F. H. Harlow and J. E. Welch, *Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface*, Phys. Fluids, **8**(12), pp. 2182-2189 (1965).

[25] J. C. Heinrich, D. R. Poirier and D. F. Nagelhout, *Mesh generation and flow calculations in highly contorted geometries*, Comput. Meth. Appl. Mech. Engrg., **133**, pp. 79-92 (1996).

[26] S. E. Hibbert, N. C. Markatos and V. R. Voller, *Computer simulation of moving-interface convection, phase-change processes*, Int. J. Heat Mass. Trans, **31**(9), pp. 1785-1795 (1988).

[27] T. Y. Hou, Z. Li, S. Osher and H. Zhao, *A hybrid method for moving interface problems with application to the Hele-Shaw flow*, J. Comput. Phys., **134**, pp. 236-252 (1997).

[28] A. Iserles, *A first course in the numerical analysis of differential equations,* Cambridge University Press, 1996.

[29] D. Juric and G. Tryggvason, *A front tracking method for dendritic solidification*, J. Comput. Phys., **123**, pp. 127-148 (1996).

[30] Alain Karma and Wouter-Jan Rappel, *Quantitative phase-field modeling of dendritic growth in two and three dimensions*, Phys. Rev. E, **57**(4), pp. 4323-4349 (1998).

[31] H. B. Keller, *Lectures on Numerical Methods in Bifurcations Problems*, Springer Verlag, 1987.

[32] J. Kim and P. Moin, *Application of a fractional-step method to incompressible Navier-Stokes equations*, J. Comput. Phys., **59**, pp. 308-323 (1985).

[33] R. Kobayashi, *Modeling and numerical simulations of dendritic crystal growth*, Physica D, **63**(3), pp. 410-423 (1993).

[34] W. Kurz and D. J. Fisher, *Fundamentals of Solidification*, Trans Tech Publications, Switzerland, 1986.

[35] J. S. Langer, *Instability and pattern formation in crystal growth*, Rev. Mod. Phys., **52**, pp. 1-28.

[36] L. G. Leal, *Laminar flow and convective transport processes,* Butterworth-Heinemann, 1992.

[37] Y. Lee, R. N. Smith, M. E. Glicksman and M. B. Koss, *Effects of bouyancy on the growth of dendritic crystals*, in Annual Review of Heat Transfer, vol. 7, C. Tien ed., Begell House, Inc., 1996.

[38] R. J. LeVeque and Z. Li, *The immersed interface method for elliptic equations with discontinuous coefficients and singular sources*, SIAM J. Numer. Anal., **31**, pp. 1019-1041 (1994).

[39] R. J. LeVeque and Z. Li, *Immersed interface methods for Stokes flow with elastic boundaries or surface tension*, University of Washington Applied Mathematics Technical Report 95-01, 1995.

[40] Z. Li, *A note on immersed interface method for three-dimensional elliptic equations*, Comput. Math. Appl., **31**(3), pp. 9-17 (1996).

[41] Z. Li, *Immersed interface methods for moving interface problems*, Num. Alg., **14**(4), pp. 269-293 (1997).

[42] Z. Li, D. F. McTigue, and J. T. Heine, *A numerical method for diffusive transport with moving boundaries and discontinuous material properties*, Int. J. Numer. Anal. Methods Geomechanics, **21**, pp. 653-662 (1997).

[43] Z. Li, *A fast iterative algorithm for elliptic interface problems*, SIAM J. Num. Anal., **35**(1), pp. 230-254 (1998).

[44] D. R. Lynch and Kevin O'Neill, *Continuously Deforming Finite Elements for the Solution of Parabolic Problems, with and without Phase Change*, Int. J. Numer. Methods Eng., **17**, pp. 81-96 (1981).

[45] D. R. Lynch, *Unified approach to simulation on deforming elements with application to phase change problems*, Comput. Phys., **47**, pp. 387-411 (1982).

[46] D. R. Lynch and J. M. Sullivan Jr., *Heat conservation in deforming element phase change simulation*, Comput. Phys., **57**, pp. 303-317 (1985).

[47] D. J. McDaniel and N. Zabaras, *A least-squares front-tracking finite element method analysis of phase change with natural convection*, Int. J. Num. Meth. Eng., **37**, pp. 2755-2777 (1994).

[48] D. I. Meiron, *Boundary integral formulation of the two-dimensional symmetric model of dendritic growth*, Physica D, **23**, pp. 329-339.

[49] D. I. Meiron, *Selection of steady states in the two-dimensional symmetric model of dendritic growth*, Phys. Rev. A, **33**(4), pp. 2704-2715.

[50] C. Misbah, H. Müller-Krumbhaar and D. E. Temkin, *Interface structure at large supercooling*, J. Phys. I, **1**, pp. 585-601 (1991).

[51] W. W. Mullins and R. F. Sekerka, *Stability of a planar interface during solidification of a dilute binary alloy*, J. Appl. Phys., **34**, pp. 323-329 (1964).

[52] H. Onuma and Y. Miyata, *Numerical simulation for melt flow around dendrite tip*, J. Jap. Inst. Met., **56**(11), pp. 1329-1335 (1992).

[53] N. Provatas, N. Goldenfeld and J. Dantzig, *Efficient computation of dendritic microstructures using adaptive mesh refinement*, Phys. Rev. Lett., **80**(15), pp. 3308-3311 (1998).

[54] W. Y. Raw and S. L. Lee, *Application of weighting function scheme on convection-conduction phase change problems*, Int. J. Heat Mass Trans., **34**(6) pp. 1503-1513 (1991).

[55] A. R. Roosen and J. E. Taylor, *Modeling crystal growth in a diffusion field using fully faceted interfaces*, Comput. Phys., **114**, pp. 113-128 (1994).

[56] A. A. Samarskii, P. N. Vabishchevich, O. P. Iliev and A. G. Churbanov, *Numerical Simulation of convection/diffusion phase change problems-a review*, Int. J. Heat Mass Trans., **36**(17), pp. 4095-4106.

[57] A. Sarkar and V. M. K. Sastri, *Heat transfer during melting in rectangular enclosures-a finite element analysis*, Int. J. Num. Meth. Fluids, **14**, pp. 83-93 (1992).

[58] A. Schmidt, *Computation of three-dimensional dendrites with finite elements*, J. Comput. Phys., **125**, pp. 293-312 (1996).

[59] J. A. Sethian and J. Strain, *Crystal growth and dendritic solidification*, J. Comput. Phys., **98**, pp. 231-253 (1992).

[60] W. Shyy, M. M. Rao and H. S. Udaykumar, *Scaling procedure and finite volume computations of phase-change problems with convection*, Eng. Anal. Bound. Elem., **16**, 123-147 (1995).

[61] J. B. Smith, *Shape instabilities and pattern formation in solidification: A new method for numerical solution of the moving boundary problem*, J. Comput. Phys., **39**, pp. 112-127 (1981).

[62] F. Sotiropoulos and S. Abdallah, *The discrete continuity equation in primitive variables solutions of incompressible flow*, J. Comput. Phys., **95**, pp. 212-227 (1991).

[63] J. Strain, *A boundary integral approach to unstable soldification*, J. Comput. Phys., **85**, pp. 342-389 (1989).

[64] J. Strain, *Linear Stability of Planar Solidification Fronts*, Physica D, **30**, pp. 297-320.

[65] G. Strang, *Linear Algebra and its applications*, Harcourt Brace Jovanovich, 1988.

[66] J. M. Sullivan Jr., D. R. Lynch, and K. O'Neill, *Finite element simulation of planar instabilities during solidification of an undercooled melt*, J. Comput. Phys., **69**, pp. 81-111 (1987).

[67] J. M. Sullivan Jr. and D. R. Lynch, *Non-linear simulation of dendritic solidification of an undercooled melt*, Int. J. Numer. Methods Eng., **25**, pp. 415-444 (1988).

[68] P. N. Swarztrauber, *Fast poisson solver*, Studies in Numerical Analysis, G.H. Golub, ed., vol 24, The Mathematical Association of America, 1984, pp. 319-370.

[69] K. Tsiveriotis and R. A. Brown, *Boundary-Conforming Mapping Applied to Computations of Highly Deformed Solidification Interfaces*, Int. J. Numer. Methods Fluids, **14**, pp. 981-1003 (1992).

[70] H. S. Udaykumar and W. Shyy, *Simulation of interfacial instabilities during solidification-I. Conduction and capillarity effects*, Int. J. Heat Mass Transfer, **38**(11), pp. 2057-2073 (1995).

[71] H. S. Udaykumar, M. M. Rao and W. Shyy, *ELAFINT - A mixed Eulerian-Lagrangian method for fluid flows with complex and moving boundaries*, Int. J. Numer. Meth. Fluids, **22**(8), pp. 691-712 (1996).

[72] A. S. Usmani, R. W. Lewis and K. N. Seetharamu, *Finite element modeling of natural-convection-controlled change of phase*, Int. J. Numer. Meth. Fluids, **14**, pp. 1019-1036 (1992).

[73] J. A. Viecelli, *A method for including arbitrary external boundaries in the MAC incompressible fluid computing technique*, J. Comput. Phys., **4**, pp. 543-551 (1969).

[74] J. A. Viecelli, *A Computing method for incompressible flows bounded by moving walls*, J. Comput. Phys., **8**, pp. 119-143 (1971).

[75] V. R. Voller and C. R. Swaminathan, *Fixed Grid Techniques for Phase Change Problems: A Review*, Int. J. Numer. Methods Eng., **30**, pp. 875-898.

[76] S. Wang and R. F. Sekerka, *Algorithms for phase field computation of the dendritic operating state at large supercoolings*, J. Comput. Phys., **127**, pp. 110-117 (1996).

[77] F. M. White, *Viscous Fluid Flow*, McGraw Hill, 1991.

[78] A. Wiegmann and K. P. Bube, *The Explicit Jump Immersed Interface Method: Finite Difference Methods for PDE's with Piecewise Smooth Solutions*, SIAM J. Numer. Anal [to appear].

[79] J. Yoo and B. Rubinsky, *A finite element method for the study of solidification processes in the presence of natural convection*, Int. J. for Num. Meth. Eng., **23**, pp. 1785-1805 (1986).

[80] D. P. Young, R. G. Melvin, M. B. Bieterman, F. T. Johnson, S. S. Samant and J. E. Bussoletti, *A locally refined rectangular grid finite element method: Application to computational fluid dynamics and comuptational physics*, J. Comput. Phys., **92**, pp. 1-66 (1992).

[81] C. Zhang and R. J. LeVeque, *The immersed interface method for acoustic wave equations with discontinuous coefficients*, Wave Motion, **25**, pp. 237-263 (1997).