Phase-Field Models with Multiple Resolution Grids: Stability and Convergence

Matthew Gallagher

The thesis is submitted to University College Dublin
in part fulfilment of the requirements for the degree of
BSc Applied and Computational Mathematics

School of Mathematics and Statistics
University College Dublin

Supervisor: Dr Chris Howland

April 13, 2025

Abstract

This report investigates 'phase-field models' used to simulate solidification or melting processes, where the sharp boundary between liquid and solid is approximated by a diffuse interface. In particular, we focused on model stability, convergence and computational efficiency, as these factors are crucial for accurately and efficiently simulating phase changes. Initially, we applied the phase-field method to a simple one-dimensional Stefan problem. The phase-field equations lack an analytical solution when applied to this problem, thereby necessitating the use numerical methods in order to solve them. We first implemented an explicit Euler scheme for solving the equations but subsequently adopted a Crank-Nicolson scheme for stability reasons. In order to improve the model's convergence, we proceeded to perform asymptotic analysis of the phase-field equations. We then found that employing a multiple resolution method on the phase-field equations managed to optimise the model's computational efficiency while preserving accuracy. Specifically, by solving for the phase variable on a fine grid and the temperature field on a coarser grid, we were able to reduce computational costs without significantly compromising convergence. This technique, although used in other areas of computational fluid dynamics, appears to be a novel application within the context of phase-field modelling. Finally, to validate the model's robustness against curvature effects, we extended the phase-field method to a two-dimensional problem with curvature.

Acknowledgements

First and foremost, I would like to thank my project supervisor, Dr. Chris Howland. Chris consistently went above and beyond what was expected of a supervisor—always making time to discuss any challenges I faced and responding to my queries with exceptional speed and clarity. His expertise, along with his ability to explain complex concepts in a clear and accessible way, greatly enhanced my understanding of the project. His guidance and feedback were instrumental in shaping both the direction and quality of this thesis. I would also like to thank the module coordinator, Dr. James Herterich, for providing the structure and support that enabled this project to take place.

Contents

1 Introduction **1**

1.1 Background . . . 1

1.2 Test Problem . . . 3

1.3 Phase-Field Method . . . 4

2 Numerical Methods **7**

2.1 Explicit Euler . . . 7

2.2 Crank-Nicolson . . . 9

2.3 Spatial Convergence . . . 11

3 Asymptotic Analysis **14**

3.1 Theory . . . 14

3.2 Spatial Convergence With Asymptotics . . . 18

3.3 Varying Interface Size . . . 19

4 Multiple Resolution **22**

4.1 Theory . . . 22

4.2 Implementation . . . 22

4.3 Interpolation . . . 23

4.4 Results . . . 24

5 2-D Problem **29**

5.1 Problem . . . 29

5.2 Results . . . 30

6 Conclusion **33**

7 Appendix **35**

7.1 Initial Conditions . . . 35

7.2 Time-Stepping Functions . . . 36

List of Figures

1.1 Diffuse Interface vs True Interface 5

2.1 Initial Conditions for ϕ and T 8
2.2 Explicit Euler Compared to Analytical Solution for Temperature 8
2.3 Explicit Euler Compared to Analytical Solution for Interface 9
2.4 Stability of Crank-Nicolson vs Explicit Euler when Varying Δx 11
2.5 Spatial Convergence of the Model 12
2.6 Error of the Model Over Time For Different Resolutions 12
2.7 Zoomed-in View of the Temperature Profile near the Interface 13

3.1 Effect of Asymptotic Expansion on Initial Condition 18
3.2 Spatial Convergence of Asymptotic Model 18
3.3 Asymptotic Expansion Smoothing Initial Error 19
3.4 Stability Condition when Varying ϵ^* 20
3.5 Convergence when Varying ϵ^* 20
3.6 Convergence when Varying ϵ^* with fixed ratio $\epsilon^* = 2\Delta x$ 21

4.1 Comparison of Cubic and Linear Interpolation Methods 24
4.2 Zoomed-in View of the Temperature Profile near the Interface Comparing Linear and Cubic Interpolation Methods 24
4.3 Spatial Convergence: Cubic vs Linear Interpolation 25
4.4 Spatial Convergence of Multiple Resolution Method Varying T Grid 27
4.5 Spatial Convergence when Varying ϵ^* with fixed ratio $\epsilon^* = 2\Delta x_\phi$ 28

5.1 Evolution of the Interface Radius Over Time for Different A^* Values 32
5.2 Radial Temperature Profiles in the Domain at Different Times 32

List of Tables

4.1 L2 Norm of the Error: Single Grid vs Multiple Resolution (Cubic Linear) 26
4.2 Run Time Comparison between Single and Multiple Resolution Methods . 26

Chapter 1

Introduction

1.1 Background

Many scientific and industrial questions involve fluid flows coupled with phase changes. These problems are complex and require unique mathematical tools. Moving boundary problems are the traditional method to model phase change phenomena. As these problems involve a moving interface, the boundary conditions form a vital (often nonlinear) part of the solution. Moving boundaries present many challenges, complicating numerical algorithms [1] and mathematical proofs [2].

In this report, we will look at an alternative method for phase-change simulation using phase-field models. Moving boundary problems model the interface, where phase change occurs as a sharp, well-defined boundary. In reality, phase change does not happen exactly at a single point; rather it takes place over a thin but finite transition region surrounding the interface. Phase-field models attempt to attack the phase change problem with a physically motivated approach by reinterpreting discontinuous boundaries as smooth transitions. In this way, phase-field models were derived from a free energy model near liquid-solid phase boundary by Langer (1986) [3]. The existence and regularity of phase-field models solutions was then shown by Caginalp (1986) [4]. Further work by Penrose and Fife (1990) [5] and Wang et al (1993) [6] formulated phase-field models based on an entropy functional thus making them thermodynamically consistent.

However, interpreting the 'physical' interface as smooth can be somewhat misleading. The actual region where phase change occurs is so small that it is beyond the resolution of practical simulations. On larger scales, the interface appears sharp and it is this sharp limit that phase-field models aim to recover as the interface thickness $\epsilon \to 0$. The extent to which these 'smooth' phase-field models can describe these real 'sharp' interface problems has been explored by Karma and Rappel (1998) [7] and is a central focus of this report.

Phase-field models represent distinct phases using a single smoothed phase-field ϕ. The evolution of these phases is then determined by a set of equations. This leads us to the chief benefit of phase-field models in that they are simple to simulate as they avoid explicit tracking of the interface. This contrasts with moving boundary problems which require specialised algorithms designed to track and apply boundary conditions at the interface.

Alternative methods to model phase-change simulation include:

- **Front Tracking:** This is a form of 'moving boundary' problem in which the moving interface is explicitly tracked [8].
- **Enthalpy Methods:** These methods entail reformulating the phase change problem by embedding the latent heat within the enthalpy term. This approach avoids explicitly tracking the interface [9].
- **Level Set Methods:** These methods represent the interface implicitly via a level set function, which helps in handling complex geometries [10].

The main drawback of these methods, in comparison to phase-field methods, is their complexity; therefore, we will not explore them further in this report.

Phase-field models remain a key area of research in modern fluid dynamics. Recent work on phase-field methods includes the modelling of solid-liquid interactions such as simulating melting and freezing of ice [11, 12, 13] which plays a crucial role in climate change research. There has also been recent research on applying phase-field modelling to liquid-liquid interactions [14, 15, 16].

In the simulation of natural systems, it is common that different variables are subject to different resolution requirements. For example, salt diffuses far more slowly than heat, leading to sharp gradients in salt concentration that require higher spatial resolution to accurately simulate than gradients in temperature [17].

Multiple-resolution methods are those in which different variables are solved on different computational grids. These methods have previously been applied to simulate turbulent convection by Ostilla-Monico et al (2015) [18]. In this paper, the authors used a grid with higher spatial resolution for the temperature gradient than the one used for momentum. They found substantial computational time and memory savings by applying a multiple-resolution method to the problem. Multiple resolution methods have also been used by Gotoh et al (2012) [19] to develop an accurate and efficient numerical method for the simulation of passive scalar dispersion in turbulent flow.

In phase-field models, the diffusive interface should be smaller than any physical gradient in the system. This imposes stricter resolution requirements on the phase field variable

ϕ, compared to other variables. Taking inspiration from the previous papers on multiple-resolution, we hypothesised that the small-scale gradients in ϕ would benefit from higher resolution. By implementing a multi-resolution approach, where ϕ is solved on a finer grid while other variables remain on a coarser, less computationally expensive domain, we aimed to achieve a more efficient numerical solution.

1.2 Test Problem

In order to validate the phase-field model we looked at applying it to a Stefan problem. In particular, we focused on the one-dimensional melting from a hot boundary problem, in which we start with an entirely solid substance that is heated from one side causing it to melt. This problem is useful as it has an analytical solution which can be used to verify the accuracy of our numerical methods later.

The equations below model melting due to a heated boundary. As heat diffuses into the material from the hot boundary, the interface $h(t)$ moves outward as more of the solid turns into liquid. The Stefan condition describes how the interface moves over time as a result of the latent heat absorbed during the phase change, balancing the difference in heat flux between the solid and liquid phases. To simplify the analysis, we assume that the entire solid is maintained at the melting temperature $T_m = 0$. Additionally, we neglect fluid motion and the presence of any solute and consider only the one-dimensional problem.

One-Dimensional Stefan Problem:

Heat equation (in liquid region $0 < x < h(t)$):

$$\frac{\partial T}{\partial t} = \frac{\partial^2 T}{\partial x^2}.$$

Stefan condition (at moving boundary $x = h(t)$):

$$\frac{dh}{dt} = \left[\frac{\partial T^+}{\partial x}\right]_{x=h(t)} - \left[\frac{\partial T^-}{\partial x}\right]_{x=h(t)}.$$

Here, T^- represents the temperature on the liquid side, while T^+ represents the temperature on the solid side. Since the solid phase is kept constant at the melting temperature $T_m = 0$, the Stefan condition simplifies accordingly:

$$\frac{dh}{dt} = -\left[\frac{\partial T^-}{\partial x}\right]_{x=h(t)}.$$

Initial and Boundary Conditions:

$$T(t=0) = 0 : x > 0,$$

$$T(x=0) = 1 : t \geq 0,$$

$$T(x=h(t)) = 0 : t > 0,$$

$$h(t=0) = 0.$$

Solved analytically using similarity variable $\eta = \frac{x}{2\sqrt{t}}$:

$$T(x,t) = 1 - \frac{\operatorname{erf}(\eta)}{\operatorname{erf}(\Lambda)} : 0 < x < h(t),$$

$$h = 2\Lambda\sqrt{t}.$$

From the above interface equation, we derive an expression for Λ:

$$\sqrt{\pi}\Lambda e^{\Lambda^2}\operatorname{erf}(\Lambda) = 1.$$

We used a root finder to find the value for Λ from the above equation: $\Lambda \approx 0.62$.

1.3 Phase-Field Method

We next tackled the above problem using a phase-field method. Here, the sharp boundary between liquid and solid is approximated by a diffuse interface. As stated above, instead of explicitly tracking the interface location (h), phase-field models represent it implicitly using a continuous variable ϕ. (Note: the position of $h(t)$ is where $\phi \approx 0.5$). As the variable ϕ is continuous this allows us to couple the equation for ϕ with the heat equation.

The size of our diffusive interface is $O(\epsilon)$. It is approximated by a tanh profile, explicitly:

$$\phi = \frac{1}{2}\left[1 + \tanh\left(\frac{x - h_0}{2\epsilon}\right)\right].$$

This approximation of the diffuse interface is valid as we will recover the ‘true’ sharp interface problem in the limit $\epsilon \to 0$.

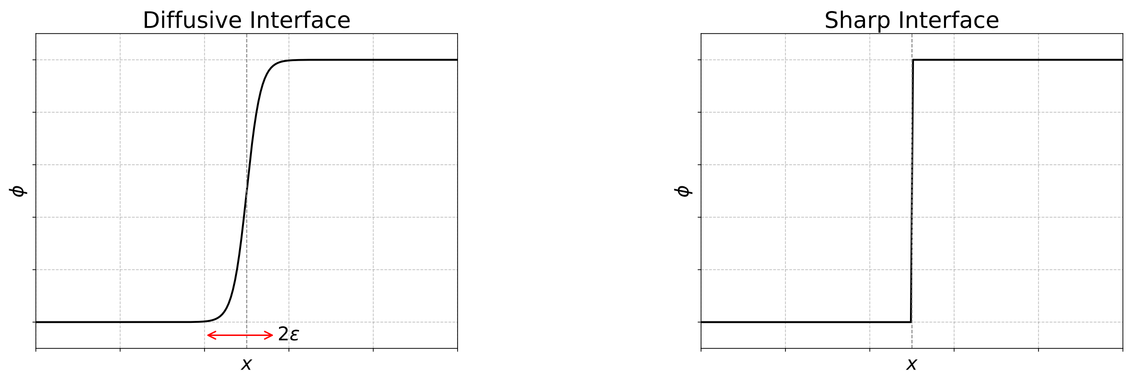

Figure 1.1: Comparison of diffuse interface (phase-field) to ‘true’ interface.

To attack the above problem, we consider the phase-field model of Hester et al. (2020) [20] for the evolution of a solid-liquid interface due to thermal conduction.

$$\partial_t T = \kappa \partial_{xx} T + \frac{L}{c_p} \partial_t T,$$

$$\epsilon \frac{5}{6} \frac{L}{\kappa} \partial_t \phi = \gamma \partial_{xx} \phi - \frac{1}{\epsilon^2} \phi(1-\phi)[\gamma(1-2\phi) + \epsilon(T - T_m)].$$

We non-dimensionalise the problem using a length scale H and a diffusive time scale:

$$\epsilon^* = \frac{\epsilon}{H}, \qquad t^* = \left(\frac{\kappa}{H^2}\right) t, \qquad x^* = \frac{x}{H}, \qquad T^* = \frac{T}{\Delta T}.$$

We arrive at the dimensionless equations:

$$\partial_{t^*} T^* = \partial_{x^* x^*} T^* + \mathcal{S} \partial_{t^*} T^*,$$

$$\partial_{t^*} \phi = D \left[\partial_{x^* x^*} \phi - \frac{1}{\epsilon^{*2}} \phi(1-\phi)(1 - 2\phi + AT^*) \right].$$

The model therefore depends on the four dimensionless control parameters:

$$\mathcal{S} = \frac{L}{c_p \Delta T}, \qquad \epsilon^* = \frac{\epsilon}{H}, \qquad D = \frac{6\gamma H}{5L\epsilon}, \qquad A = \frac{\epsilon \Delta T}{\gamma}.$$

The Stefan number is a physical parameter relating the latent heat to the sensible heat, but the other three are only model parameters. However, the four control parameters are not independent, rather D can be expressed in terms of the other three control parameters:

$$D = \frac{6\gamma H}{5L\epsilon} = \frac{6}{5}\frac{c_p\Delta T}{L}\frac{\gamma}{\epsilon\Delta T} = \frac{6}{5}\frac{1}{\mathcal{S}}\frac{1}{A}.$$

For ease of calculation, we set $\mathcal{S} = 1$. Also for clarity, we remove the asterisks from the above terms. Our equations become:

$$\partial_t T = \partial_{xx} T + \partial_t T,$$

$$\partial_t \phi = \frac{6}{5A}\left[\nabla^2\phi - \frac{1}{\epsilon^2}\phi(1-\phi)(1-2\phi+AT)\right].$$

These equations do not have an analytical solution and required the use of numerical methods in order to solve them. However, it is worth noting that for $T = 0$, the tanh profile is actually a steady solution for these equations.

Chapter 2

Numerical Methods

2.1 Explicit Euler

The first numerical method we tried in order to solve the phase-field equations coupled to temperature was explicit Euler. As a time stepper, explicit Euler updates the solution at each discrete time step by using the value at the current time step to approximate the next value. At each step, we solve the equation for ϕ first as it appears in our temperature equation. In the following calculations, i denotes the spatial position in the grid, while j represents the time step.

We use central finite difference to find a a 2nd order accurate approximation of the diffusive ($\nabla^2\phi$) term:

$$D_{\phi_j} = \frac{1}{(\Delta x)^2}(\phi_{i-1,j} - 2\phi_{i,j} + \phi_{i+1,j}).$$

We then update ϕ in time using forward difference to find a 1st order accurate approximation:

$$\phi_{i,j+1} = \phi_{i,j} + \Delta t\left(\frac{6}{5A}\right)\left[D_{\phi_j} + \frac{1}{\epsilon^2}\phi_{i,j}(1-\phi_{i,j})(1-2\phi_{i,j}+AT_{i,j})\right].$$

After solving for ϕ, we move onto solving for T. We again use central difference to find a 2nd order accurate approximation of the diffusive ($\nabla^2 T$) term:

$$D_{T_j} = \frac{T_{i-1,j} - 2T_{i,j} + T_{i+1,j}}{(\Delta x)^2}.$$

We then update T in time using a 1st order forward difference approximation for $\partial_t T$ and $\partial_t\phi$:

$$T_{i,j+1} = T_{i,j} + \phi_{i,j+1} - \phi_{i,j} + \Delta t(D_{T_j}).$$

However, we cannot directly start our simulation with the initial interface position $h_0 = 0$ and initial time $t_0 = 0$ for two main reasons. The first is that the temperature field is not smooth at $t = 0$ as our initial conditions impose that $T = 1$ at $x = 0$ and $T = 0$ for $x > 0$. The second is that, at $h_0 = 0$, we do not capture the full tanh profile of ϕ. We therefore slightly perturb the position of the interface starting at $h_0 = 0.1$.

If we move the interface away from 0, we also cannot start our simulation at $t_0 = 0$. In order to calculate t_0, we use the analytical solution $t_0 = (\frac{h_0}{2\Lambda})^2$. We also need to use the analytical solution for temperature to determine the temperature profile within the grid for $x < h_0$. In this way, the initial conditions for ϕ and T provided to the time stepper are based on the analytical solution.

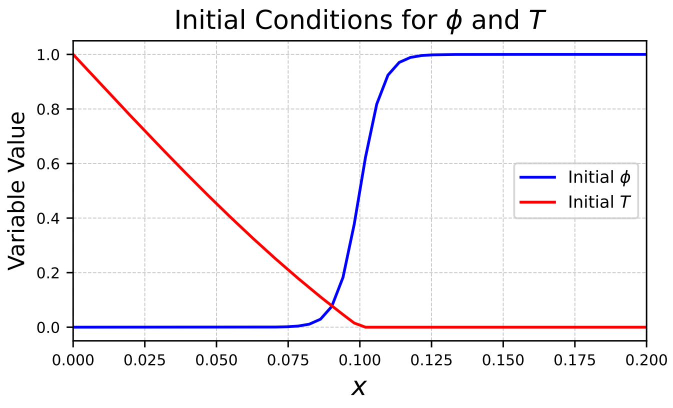

Figure 2.1: Initial conditions for ϕ and T inserted into the time stepper.

We set $A = 1$ in our initial simulations for numerical stability purposes (We will have a further analysis of this A term later in the report). As Figures 2.2 and 2.3 show, the numerical solution produced by explicit Euler match the analytical solution relatively well.

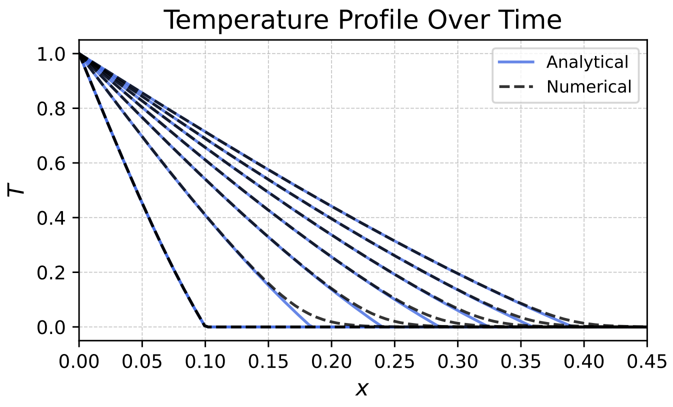

Figure 2.2: Temperature profile over time. The black dashed line represent the numerical solution obtained using the explicit Euler method, the blue line shows the corresponding analytical solution.

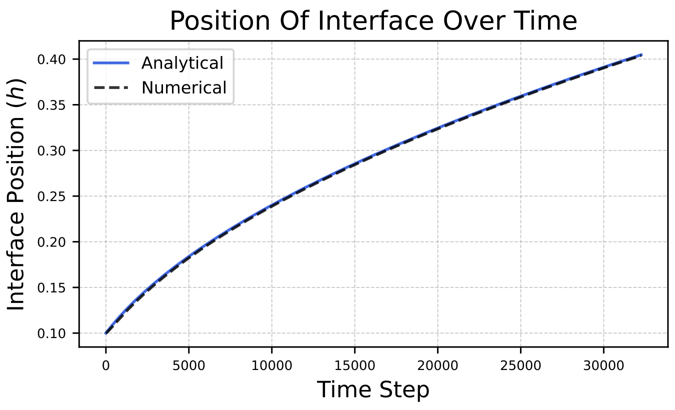

Figure 2.3: Position of the interface over time. The black dashed line represent the numerical solution obtained using the explicit Euler method, the blue line shows the corresponding analytical solution.

However, upon further analysis, this method faced a number of issues. The most obvious being the model's stability. As we are implementing a finite difference scheme on both the diffusive and nonlinear terms in our equations, both terms will have strict stability requirements. In other words, the stability depends both on Δx and on ϵ. This causes significant issues when we want to study the convergence of different resolutions, as the finer we make our grid, the smaller time steps we are required to take. For this reason, we decided to implement a Crank-Nicolson scheme (which is unconditionally stable) on the diffusive terms in the coupled equations.

Another issue that cropped up is that the phase-field model only produces the analytical solution in the limit $\epsilon \to 0$. However, even as we decrease ϵ, we must still simulate for finite values of ϵ. This introduces an error that is inherent to the model rather than error with our numerical method. This 'model' error can be seen in the 'jump' in difference between the analytical and numerical profile for temperature at the position of the interface. This 'jump' in error is due to the model 'smoothing' the interface with the continuous ϕ term. Implementing the Crank-Nicolson scheme will improve the model convergence, however, it will not fix this inherent model error.

2.2 Crank-Nicolson

As stated above, we modified our initial numerical method by implementing a Crank-Nicolson scheme on the diffusive terms in our coupled equations. A Crank-Nicolson scheme averages the explicit and implicit Euler schemes by evaluating at both the current time step j and the next time step j+1.

In our case, we apply this scheme to the diffusive term. Using the same notation for D_{ϕ_j} as above, our equation for updating ϕ becomes:

$$\phi_{i,j+1} = \phi_{i,j} + \Delta t \left(\frac{6}{5A}\right)\left[\frac{1}{2}(D_{\phi_{j+1}} + D_{\phi_j}) + \frac{1}{\epsilon^2}\phi_{i,j}(1-\phi_{i,j})(1-2\phi_{i,j}+AT_{i,j})\right].$$

Re-arranging, we obtain:

$$\underbrace{\left[1 - \frac{3\Delta t}{5A}D_\phi\right]}_{M}\phi_{j+1} = \phi_{i,j} + \Delta t \left(\frac{6}{5A}\right)\left[\frac{1}{2}D_{\phi_j} + \frac{1}{\epsilon^2}\phi_{i,j}(1-\phi_{i,j})(1-2\phi_{i,j}+AT_{i,j})\right].$$

We call the right side of the equation b and calculate:

$$\phi_{j+1} = M^{-1}b.$$

After solving for ϕ, we move onto solving for T; our equation for updating T becomes:

$$T_{i,j+1} = T_{i,j} + \phi_{i,j+1} - \phi_{i,j} + \frac{\Delta t}{2}(D_{T_{j+1}} + D_{T_j}).$$

Re-arranging, we obtain:

$$\underbrace{\left[1 - \frac{\Delta t}{2}D_T\right]}_{C}T_{j+1} = T_{i,j} + \phi_{i,j+1} - \phi_{i,j} + \frac{\Delta t}{2}(D_{T_j}).$$

We call the right side of the equation d and calculate:

$$T_{j+1} = C^{-1}d.$$

The main improvement that this scheme provides over explicit Euler is its stability. As shown in Figure 2.4, the stability of the Crank-Nicolson scheme does not depend on the value of Δx, meaning we can make our grid as fine as we want without having to decrease our time step. This contrasts with explicit Euler in which we must ensure $\Delta t \propto (\Delta x)^2$ for stability. This change makes analysis of convergence of different grid spacing for a fixed ϵ much easier.

The Crank-Nicolson scheme is, however, not unconditionally stable. This is due to the fact the nonlinear terms are still being treated explicitly in our coupled equations. We will discuss the stability constraint on ϵ later in the report; however, it is important to note that the stability of the system still depends on ϵ. We also note the additional constraint $\epsilon \geq \Delta x$, which comes from the fact our solution will become unstable if the size of our interface is smaller than the spatial resolution [21]. Therefore, if we want to make ϵ smaller (to make the interface sharper and reduce model error), then we also need to reduce Δx. This is necessary to ensure that the smooth tanh profile is resolved by the grid. Consequently, this reduction demands a finer grid with smaller time steps, further increasing computational costs.

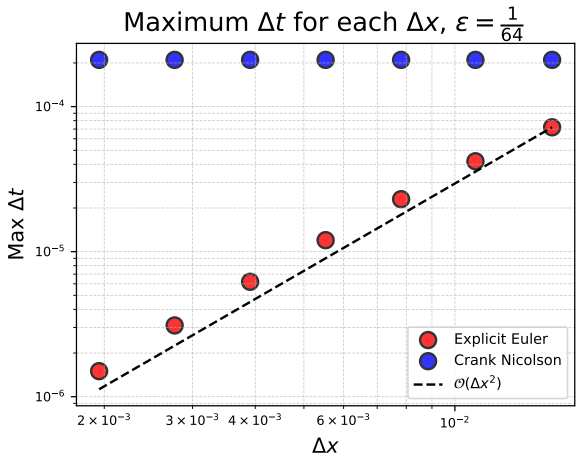

Figure 2.4: Stability comparison between the Crank–Nicolson and explicit Euler methods as Δx is varied, with $\epsilon = \frac{1}{64}$. Red dots indicate the maximum stable time step Δt for each Δx using the explicit Euler method, while blue dots show the corresponding values for the Crank–Nicolson scheme.

2.3 Spatial Convergence

We next wanted to analyse the convergence of our numerical solver to the analytical solution. If there was no model error, we would expect $\mathcal{O}(\Delta x^2)$ convergence of the model to the analytical solution as we decrease Δx. To quantify our model's convergence, we evaluated the position of the interface $h(t)$ produced by our model at 100 equally spaced time intervals and compared it to the analytical solution of $h(t)$. This comparison was made by calculating the L2 norm of the difference in $h(t)$ across all time steps, with a lower value indicating greater accuracy.

In Figure 2.5, we quantify the convergence of the model as we increase spatial resolution. A series of simulations were run keeping ϵ constant and varying Δx and Δt so that $\frac{\Delta t}{(\Delta x)^2}$ is kept constant. The model initially converges at a rate $\mathcal{O}(\Delta x^2)$, but this convergence eventually plateaus and any resolution higher than a certain point shows little to no improvement in convergence to the analytical solution. Therefore, our numerical solution does not show spatial convergence to the analytical solution.

Before investigating what was causing this lack of convergence, we first verified that we had implemented the numerical method correctly. We did this by first running a simulation with very high resolution. We then compared this high resolution case to the lower resolution simulations which we had already ran. As Figure 2.5 shows, the lower resolution cases do converge to the high resolution case, as we would expect at a rate

$\mathcal{O}(\Delta x^2)$.

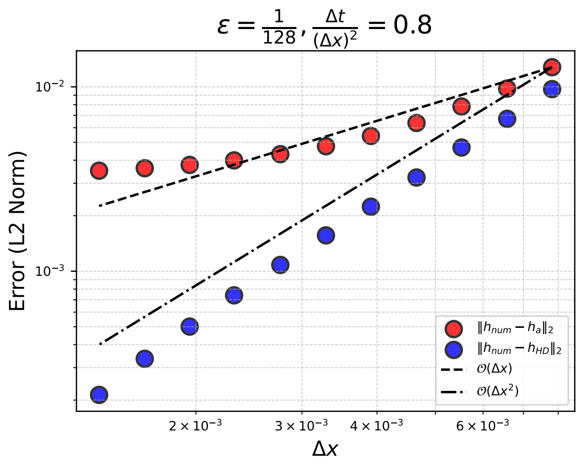

Figure 2.5: Spatial convergence of the model with fixed $\epsilon = \frac{1}{128}$ and $\frac{\Delta t}{(\Delta x)^2} = 0.8$, while varying the spatial resolution Δx. Red dots represent the error in interface position relative to the analytical solution. Blue dots show the error relative to a high-resolution numerical reference solution.

Upon further analysis, the majority of the error is caused by an initial jump in error that occurs in the first time step. In Figure 2.6, we plot the error at each time step for different resolutions. The graph shows that the initial time step introduces an error that persists throughout the simulation. The graph also shows a decrease in this initial error jump the more fine we make the resolution. While this improvement is substantial initially, it begins to plateau as we make our resolution increasingly fine.

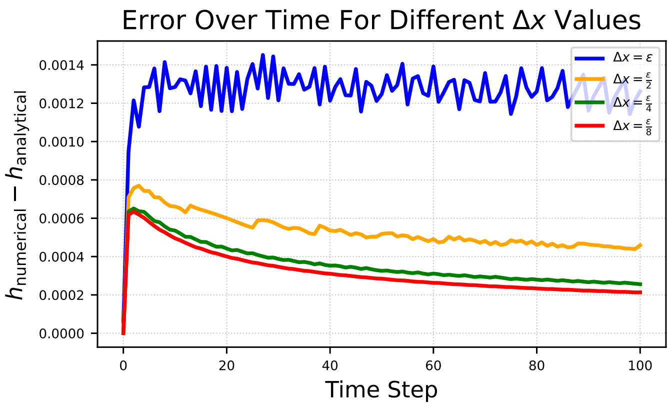

Figure 2.6: Error in interface position over time for different spatial resolutions Δx, comparing the model to the analytical solution.

Taking a closer look at the temperature profile, we can see where this initial error jump comes from. Figure 2.7 shows a magnified picture of the temperature profile at the interface at the first time step. Since we used the sharp (analytical) solution as our initial condition, there is no error initially. However, error is introduced at the first time step as the sharp boundary at the interface is smoothed into a continuous boundary within the phase-field equations. To try and correct this initial error jump, we introduced some asymptotic analysis.

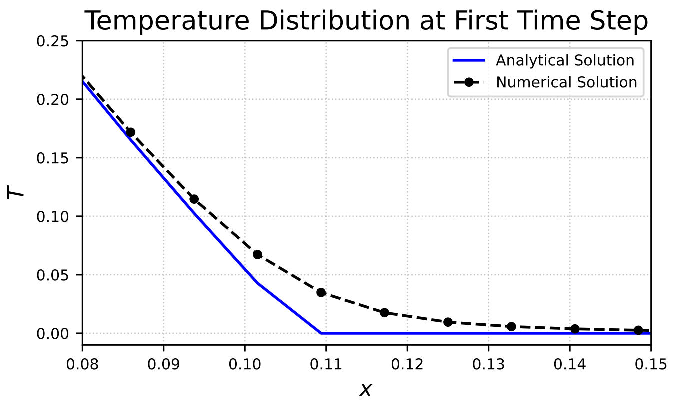

Figure 2.7: Zoomed-in view of the temperature profile near the interface at the first time step. The black dashed line represents the numerical solution from the model, the blue line shows the corresponding analytical solution.

Chapter 3

Asymptotic Analysis

3.1 Theory

To address the above issue, we introduced asymptotic analysis to better understand and potentially correct the initial error jump. Specifically, we performed asymptotic expansions of our variables ϕ and T in our phase-field equations. We consider separate expansions in the outer regions away from the interface and in an inner region near the interface. It is important to note that approach specifically focuses on the initial condition to better capture the temperature profile from the outset. We will use ϕ in our example below but the same expansion applies for T:

Outer region to the right of the interface:

$$\phi^+(x,t) = \sum_{k=0}^{N} \epsilon^k \phi_k^+(x,t) = \phi_0^+(x,t) + \epsilon\phi_1^+(x,t) + \epsilon^2\phi_2^+(x,t)\dots$$

Outer region to the left of the interface:

$$\phi^-(x,t) = \sum_{k=0}^{N} \epsilon^k \phi_k^-(x,t) = \phi_0^-(x,t) + \epsilon\phi_1^-(x,t) + \epsilon^2\phi_2^-(x,t)\dots$$

Inner region near the interface:

$$\phi(\xi,\tau) = \sum_{k=0}^{N} \epsilon^k \phi_k(\xi,\tau) = \phi_0(\xi,\tau) + \epsilon\phi_1(\xi,\tau) + \epsilon^2\phi_2(\xi,\tau)\dots$$

Here, the inner variables are defined as $\xi = \frac{x-h}{\epsilon}$ and $\tau = t + vx$, where $v = \frac{dh}{dt}$ is the velocity of the interface. The leading order solutions match our original phase-field equations except the $\partial_t\phi$ term in the temperature equation which is imposed at a higher order:

Leading order for T:

$$\partial_t T_0^- = \partial_{xx} T_0^-, \qquad T_0^+ = 0, \qquad T_0(\xi) = T_0^\pm\big|_{x=0}.$$

Leading order for ϕ:

$$\phi_0^- = 0, \qquad \phi_0^+ = 1, \qquad \phi_0(\xi) = \frac{1}{2}\left[1 + \tanh\left(\frac{\xi}{2}\right)\right].$$

As shown by Hester et al. (2020) [20], first-order corrections are only present in the interface region. We turn our focus to the temperature equation:

$$\partial_t T = \partial_{xx} T + \partial_t \phi.$$

In the inner region, we make the change of variables:

$$\partial_x \to \frac{1}{\epsilon}\partial_\xi, \qquad \partial_t \to \partial_\tau - \frac{v}{\epsilon}\partial_\xi.$$

The expanded temperature equation becomes:

$$(\partial_\tau - \frac{v}{\epsilon}\partial_\xi)(T_0 + \epsilon T_1 + \dots) = \frac{1}{\epsilon^2}\partial_{\xi\xi}(T_0 + \epsilon T_1 + \dots) + (\partial_\tau - \frac{v}{\epsilon}\partial_\xi)(\phi_0 + \epsilon\phi_1 + \dots).$$

At Leading Order $\mathcal{O}\left(\frac{1}{\epsilon^2}\right)$:

$$0 = \frac{1}{\epsilon^2}\partial_{\xi\xi} T_0.$$

If we expand the outer solutions around x = h, we obtain:

$$T_0^\pm(\epsilon\xi) \sim T_0^\pm(h) + (\epsilon\xi)\partial_x T_0^\pm(h) + (\epsilon\xi)^2 \partial_{xx} T_0^\pm(h) + \cdots$$

Using the Van Dyke matching rule, the inner expansion $T(\frac{x}{\epsilon})$ should match the expression above in the limit $\epsilon \to 0$. This gives $T_0 \equiv 0$ at leading order.

Therefore, we focus on the equation at first order $\mathcal{O}\left(\frac{1}{\epsilon}\right)$:

$$0 = \frac{1}{\epsilon}\partial_{\xi\xi} T_1 - \frac{v_0}{\epsilon}\partial_\xi \phi_0.$$

This gives us the rescaled inner equation for temperature

$$\partial_{\xi\xi} T_1 = v_0 \partial_\xi \phi_0.$$

Integrating once gives:

$$\partial_\xi T_1 = v_0 \phi_0 + c_1.$$

Applying the matching rule again implies that at first order:

$$\lim_{\xi \to -\infty} \partial_\xi T_1 = \partial_x T_0^-(h), \qquad \lim_{\xi \to \infty} \partial_\xi T_1 = \partial_x T_0^+(h).$$

Combining this with the integrated first-order equation, we can then derive v_0. As $\xi \to -\infty$, $\phi_0 \to 0$:

$$\partial_x T_0^-(h) = c_1.$$

As $\xi \to \infty$, $\phi_0 \to 1$:

$$\partial_x T_0^+(h) = v_0(1) + \partial_x T_0^-(h),$$

$$v_0 = [\partial_x T_0^+ - \partial_x T_0^-]_{x=h}.$$

This shows that the velocity of the moving boundary is proportional to the temperature gradients on either side of the boundary. Physically, this represents the fact that the movement of the interface is driven by the difference in heat flux entering and leaving the boundary. Hence, our solution for v_0 matches the sharp interface condition, as desired.

Now we have determined c_1 and v_0, we integrate again:

$$T_1(\xi) = v_o \int_0^\xi \phi_0(\eta)\partial\eta + \xi \partial_x T_0^-(h) + c_0,$$

$$T_1(\xi) = v_o \int_0^\xi \frac{1}{2}\left[1 + \tanh\left(\frac{\eta}{2}\right)\right] \partial\eta + \xi \partial_x T_0^-(h) + c_0,$$

$$T_1(\xi) = v_o \left[\frac{\xi}{2} + \log\left(\cosh\left(\frac{\xi}{2}\right)\right)\right] + \xi \partial_x T_0^-(h) + c_0.$$

Note that the limiting behaviour of this expression as $\xi \to \pm\infty$ is:

$$T_1(\xi) \sim c_0 - v_0 \log 2 + \xi \partial_x T_0^\pm(h).$$

Hester et al. [20] demonstrate that the first-order correction term satisfies $T_1^\pm = 0$. Therefore, matching the inner and outer solutions at the interface $x = h$ requires that $c_0 = v_0 \log 2$.

A composite solution can be constructed by adding the adding the outer and inner expansions together and subtracting the solution in an overlap region. The key point here is the overlap region is different for $\xi > 0$ and $\xi < 0$ and its gradient is discontinuous at $x = h$. By subtracting the appropriate overlap solution in the appropriate regions, we obtain a composite solution whose gradient is continuous at $x = h$:

$$T_{\text{comp}} = \begin{cases} T_0^-(x) + \epsilon T_1(\xi) - (x+h)\frac{dT_0^-}{dx}(h) & \text{if } x < h, \\ \underbrace{T_0^+(x)}_{\text{outer}} + \underbrace{\epsilon T_1(\xi)}_{\text{inner}} - \underbrace{(x+h)\frac{dT_0^+}{dx}(h)}_{\text{overlap}} & \text{if } x > h. \end{cases}$$

<u>For $x < h$:</u>

$$T_{\text{new}} = T_0^-(x) + \epsilon v_o \left[\frac{\xi}{2} + \log\left(\cosh\left(\frac{\xi}{2}\right)\right) + \log 2\right] + \epsilon\xi\partial_x T_0^-(h) - (x-h)\partial_x T_0^-(h),$$

$$T_{\text{new}} = T_0^-(x) + \epsilon v_o \left[\frac{\xi}{2} + \log\left(2\cosh\left(\frac{\xi}{2}\right)\right)\right].$$

<u>For $x > h$:</u>

$$T_{\text{new}} = \epsilon v_o \left[\frac{\xi}{2} + \log\left(\cosh\left(\frac{\xi}{2}\right)\right) + \log 2\right] + \epsilon\xi\partial_x T_0^-(h) - (x-h)\partial_x T_0^+(h).$$

Using the fact $\partial_x T_0^-(h) = [\partial_x T_0^-(h) - v_0]$, we get:

$$T_{\text{new}} = \epsilon v_o \left[\frac{\xi}{2} + \log\left(2\cosh\left(\frac{\xi}{2}\right)\right)\right] + (x-h)\left[\partial_x T_0^-(h) - v_0\right] - (x-h)\partial_x T_0^+(h),$$

$$T_{\text{new}} = \epsilon v_o \left[-\frac{\xi}{2} + \log\left(2\cosh\left(\frac{\xi}{2}\right)\right)\right].$$

Combining, we arrive at:

$$T_{\text{new}} = \begin{cases} T_0(x) + \epsilon v_o \left[\log\left(2\cosh\left(\frac{\xi}{2}\right)\right) + \frac{\xi}{2}\right] & \text{if } x < h, \\ \epsilon v_o \left[\log\left(2\cosh\left(\frac{\xi}{2}\right)\right) - \frac{\xi}{2}\right] & \text{if } x > h. \end{cases}$$

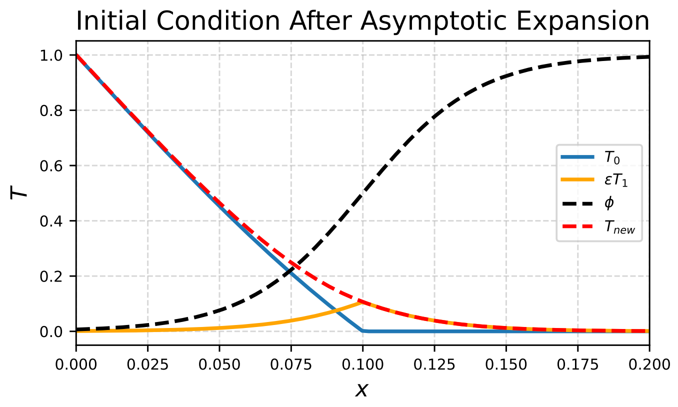

Figure 3.1: Illustration of the effect of asymptotic expansion on the initial temperature condition. The blue line shows the leading-order temperature profile T_0, while the orange line represents the first-order correction ϵT_1. The red dashed line combines these to form the corrected initial condition T_{new} after applying the asymptotic expansion.

3.2 Spatial Convergence With Asymptotics

We now revisit the convergence of our model to see if this asymptotic expansion has had an effect. In Figure 3.2, we compared our new initial condition using asymptotic expansions to our previous initial condition to see if there was an improvement in spatial convergence. As the graph shows, the asymptotic expansions have provided a substantial improvement in spatial convergence in comparison to our original initial condition. When using the asymptotic initial condition, the model follows a $\mathcal{O}(\Delta x^2)$ convergence for a longer period. The model still runs into the issue of the convergence plateauing as we make our resolution increasingly fine. This, however, only occurs at very fine resolutions. Further improvements to convergence could possibly be made by going into 2nd order of the asymptotic analysis.

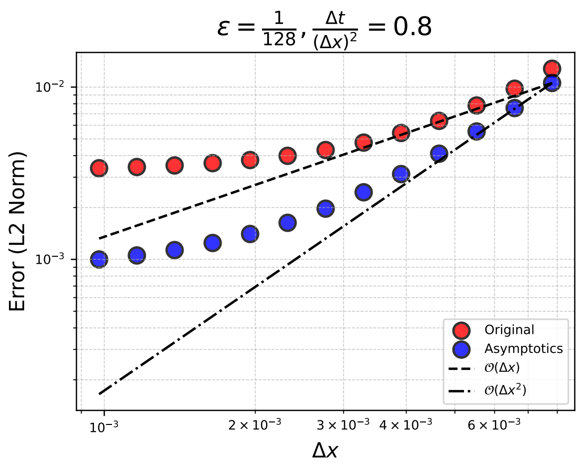

Figure 3.2: Spatial convergence of the model with fixed $\epsilon = \frac{1}{128}$ and $\frac{\Delta t}{(\Delta x)^2} = 0.8$, while varying the spatial resolution Δx. Blue dots represent the error in interface position, relative to the analytical solution, when using asymptotic initial conditions. Red dots show the corresponding error when using the original initial conditions.

From our earlier analysis, we recall the majority of the error in our model is caused by the 'jump' in error that occurs in the first time step. This was our motivation for introducing the asymptotic expansion. As Figure 3.3 shows, the expansion is doing exactly what we wish in this regard, 'smoothing' this initial jump in error.

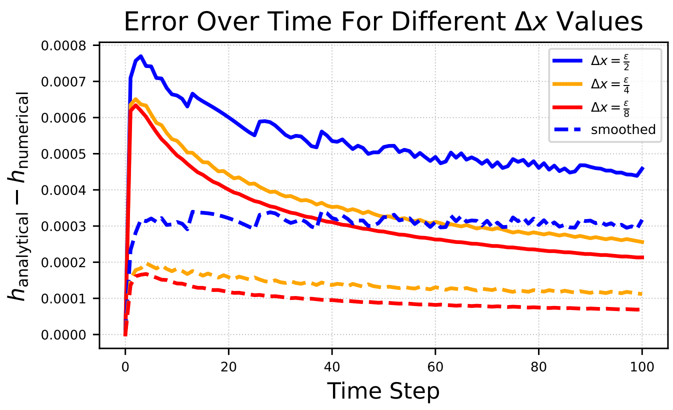

Figure 3.3: Error in interface position over time for different spatial resolutions Δx, comparing the model to the analytical solution. Solid lines represent simulations using the original initial conditions, while dashed lines show results using asymptotic initial conditions for the same resolutions.

3.3 Varying Interface Size

Suppose we now want to vary ϵ in our model to see how it affects convergence. To do this, we must first revisit our phase-field equations. Our parameter $A = \frac{\epsilon^* \Delta t}{H\gamma}$ still explicitly depends on the interface thickness. Therefore, we want to extract this dependence as $A = \epsilon^* A^*$. Our governing equation for ϕ then reads:

$$\partial_t \phi = \frac{6}{5A^*\epsilon^*}\left[\underbrace{\nabla^2\phi - \frac{1}{(\epsilon^*)^2}\phi(1-\phi)(1-2\phi}_{\sim\frac{1}{A^*(\epsilon^*)^3}} + \underbrace{A^*\epsilon^* T)}_{\sim\frac{1}{(\epsilon^*)^2}}\right].$$

As indicated above, we must carefully choose the size of A^* relative to ϵ^*. There is a limit to how large we can make A^* without compromising the solution.

$$A^* = \mathcal{O}(1) \implies \partial_{xx}\phi - \phi(1-\phi)(1-2\phi).$$

This equation's solution is a tanh profile for ϕ which is what we want. In fact, we will recover the tanh profile for ϕ as long as A^* remains up to $\mathcal{O}(\frac{1}{\epsilon^*})$. However, if A^* exceeds this limit, the ϕ profile begins to lose shape and no longer accurately model the sharp interface problem. The interface becomes distorted and loses its well-defined structure.

The convergence of ϵ^* can now be analysed taking A^* to be fixed. For convenience, we let $A^* = 128$. Before analysing convergence, it is necessary to check the stability condition on ϵ^*.

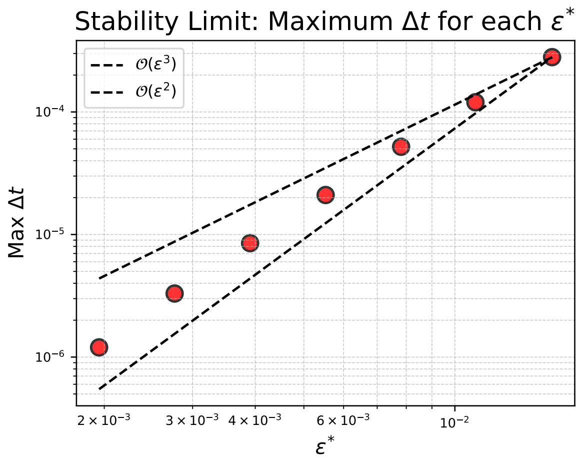

Figure 3.4: Stability condition of the model as ϵ^* is varied. Red dots indicate the maximum stable time step Δt for each ϵ^*.

As shown in Figure 3.4, the stability condition when varying ϵ^* is strict, lying between $\mathcal{O}(\Delta\epsilon^3)$ and $\mathcal{O}(\Delta\epsilon^2)$. This strictness arises because the non-linear term in the equation for ϕ scales as $\mathcal{O}(\frac{1}{(\epsilon^*)^3})$, making it the 'stiff term' in the equation. Consequently, reducing ϵ^* drastically increases the computational cost of running a simulation. This makes analysis of convergence of ϵ^* challenging. Next, we investigate if varying ϵ^* has an effect on convergence.

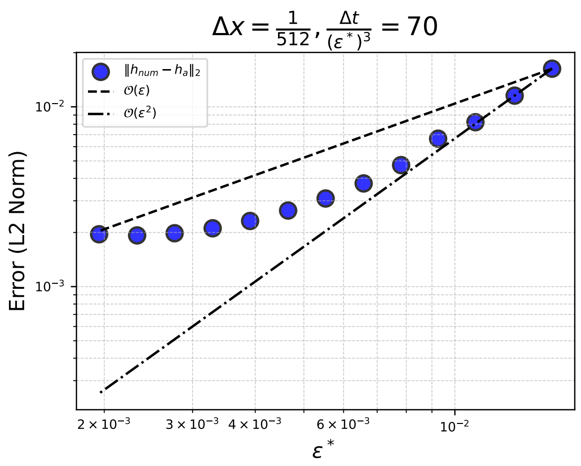

Figure 3.5: Convergence of the model with fixed spatial resolution $\Delta x = \frac{1}{512}$ and time step ratio $\frac{\Delta t}{(\epsilon^*)^3} = 70$, while varying ϵ^*. Blue dots represent the error in interface position relative to the analytical solution for each value of ϵ^*.

In Figure 3.5, we keep a constant resolution and vary the size of ϵ^* with the lowest value of $\epsilon^* = \Delta x$. As the graph shows, the size of ϵ^* does have an effect on the convergence of the model. However, as we make ϵ^* increasingly small, the convergence rate eventually plateaus. We get approximately the same convergence for $\epsilon^* = \Delta x$ and $\epsilon^* = 2\Delta x$.

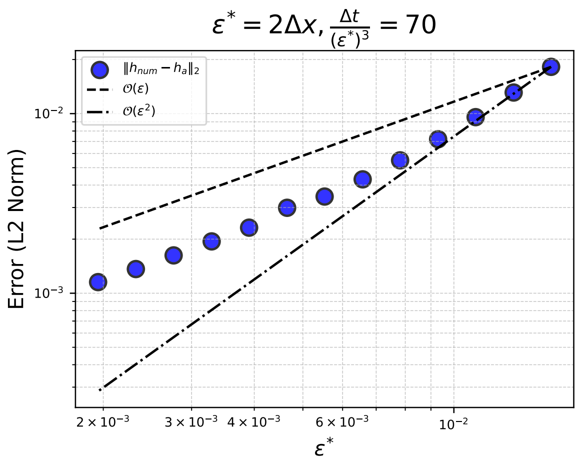

Figure 3.6: Convergence of the model with fixed ratio $\epsilon^* = 2\Delta x$ and $\frac{\Delta t}{(\epsilon^*)^3} = 70$, while varying ϵ^*. Blue dots represent the error in interface position relative to the analytical solution for each value of ϵ^*.

In Figure 3.6, we explore the effect of varying both ϵ^* and Δx while maintaining the ratio $\epsilon^* = 2\Delta x$. This approach yields the best convergence observed thus far. Although the convergence of the model does eventually slow down, the effect is less pronounced compared to our previous analyses. Ideally, when running simulations, we would minimise ϵ^* while maintaining the ratio $\epsilon^* = 2\Delta x$ in order to achieve the highest accuracy. However, as mentioned earlier, reducing ϵ^* significantly increases computational cost. In the following section, we will discuss multiple-resolution methods designed to optimise simulation efficiency and reduce overall runtime.

Chapter 4

Multiple Resolution

4.1 Theory

Multiple resolution methods are those in which different variables are solved on different computational grids. This allows a more efficient numerical solution as only one variable has to be solved on the largest, most expensive computational domain.

In our case, we wished to understand the relationship between grid size and our variables ϕ and T on the accuracy of our model. We wanted to find out which (if either) variable required a more fine grid and which variable would have the same accuracy on a coarse grid. If only one of the variables was needed to be solved for on a more fine grid it could substantially reduce the simulation time of our model at the same accuracy.

4.2 Implementation

We implemented our multiple resolution scheme on the Crank-Nicolson method described above and we also used an asymptotic expansion on the initial condition. The main difference with the multiple resolution scheme is that we set two separate grids, one with N_ϕ grid points (where ϕ is solved for) and one with N_T grid points (where T is solved for).

In our Crank-Nicolson scheme, at each time step ϕ is solved for before T. However, as now ϕ and T are stored on grids with different numbers of grid points, this causes issues. We therefore need to find the values of T at the grid points on the ϕ grid. We do this by interpolating our T grid onto our ϕ grid at the beginning of each time step. Once ϕ is solved for, we interpolate our ϕ grid onto our T grid and solve for T.

4.3 Interpolation

Before delving deeper into convergence analysis, we first wanted to make sure we were implementing the most accurate interpolation method. To do this, we compared two interpolation techniques: linear interpolation and cubic spline interpolation. For linear interpolation, we used the scipy.interpolate function interp1d, while for cubic interpolation, we utilised the scipy.interpolate function CubicHermiteSpline.

Figure 4.1 illustrates what the interpolation process is 'actually doing'. For these plots, we sampled arbitrary T and $\partial_t\phi$ profiles and applied our interpolation methods to them. In this case, we used a coarse grid for our T variable and a fine grid for our ϕ variable. The first variable to be interpolated is T, which is mapped onto the ϕ grid. This direction of interpolation is more challenging as it involves mapping data from a coarser grid onto a finer one, meaning there are fewer original data points to sample from. From the plot, we can observe that linear and cubic interpolation deal with the original temperature profile differently. Although the difference is small, it is visually noticeable; the linear interpolation produces a more jagged profile, while the cubic interpolation results in a much smoother curve.

The second plot in Figure 4.1 shows how interpolation affects $\partial_t\phi$, the other variable which we need to interpolate for. In this case, the difference between linear and cubic interpolation is negligible. This is expected, as here we are interpolating from the finer ϕ grid onto the coarser T grid, making interpolation simpler as we are estimating values between points that are closely spaced. That said, both interpolation methods still struggle to accurately capture the peak values of $\partial_t\phi$. This is to be expected as sharp features such as peaks are difficult to resolve when interpolating from a fine grid onto a coarser one.

Figure 4.2 compares linear and cubic interpolation for the temperature variable T after the model has evolved to a simulation time of $t = 0.01$. As before, linear interpolation produces a noticeably more jagged profile, while cubic interpolation yields a smoother curve. Interestingly, a closer visual inspection suggests that the linearly interpolated values may align more closely with the analytical solution than the cubically interpolated ones. This could be due to slight 'oversmoothing' introduced by the cubic interpolation. However, the observed difference is subtle and difficult to quantify from visual inspection alone. To determine which interpolation method is more accurate, we proceed with a convergence analysis in the next section.

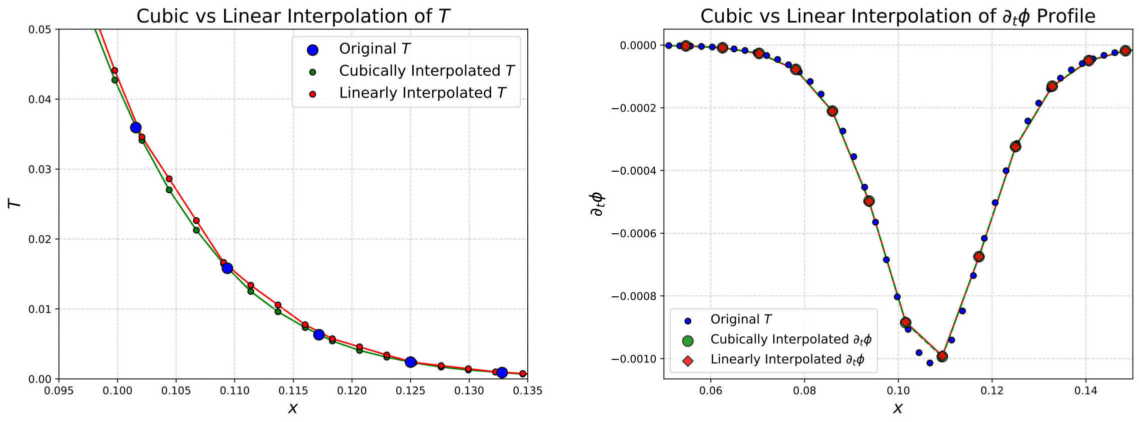

Figure 4.1: Comparison of cubic and linear interpolation methods. The left plot shows how cubic (green dots) and linear (red dots) interpolation approximate a temperature profile T, relative to the original data (blue dots). The right plot compares the same interpolation methods applied to a $\partial_t \phi$ profile.

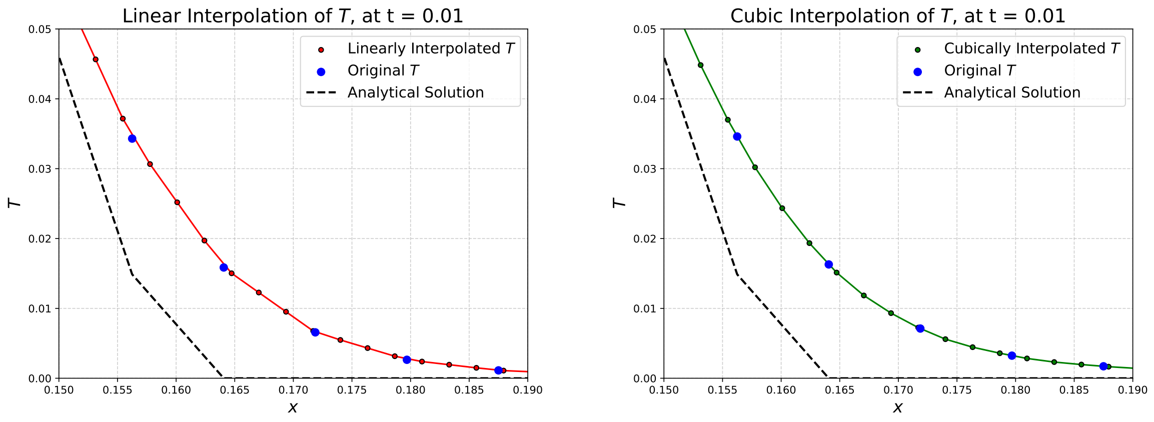

Figure 4.2: Zoomed-in view of the temperature profile near the interface at simulation time $t = 0.01$. The left plot compares the analytical solution to the result obtained using linear interpolation, while the right plot shows the comparison using cubic interpolation.

4.4 Results

We next aimed to determine which interpolation method performs better and how the multiple resolution method compares to the single grid method overall. We did this by examining the spatial convergence of the models by keeping ϵ constant while varying our grid resolutions. In Figure 4.3, we used a coarse grid for T with $N_T = 128$ across all of our simulations and varied the size of N_ϕ. The plot clearly shows that linear interpolation exhibits better spatial convergence to the analytical solution compared to cubic interpolation.

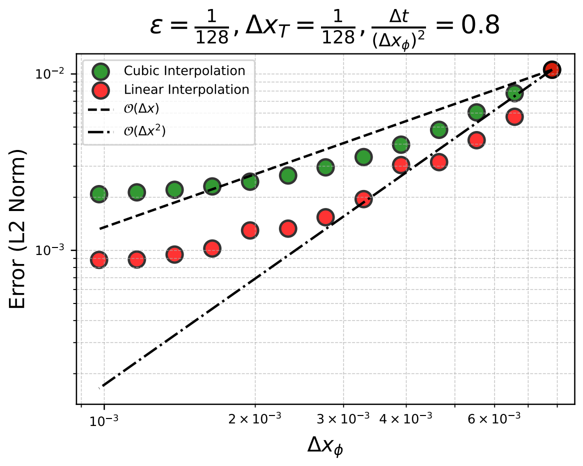

Figure 4.3: Spatial convergence of the model using a multiple resolution method, with $\epsilon = \frac{1}{128}$, $\Delta x_T = \frac{1}{128}$ and $\frac{\Delta t}{(\Delta x_\phi)^2} = 0.8$, while varying the resolution of the ϕ grid Δx_ϕ. Red dots represent the error in interface position relative to the analytical solution when using linear interpolation. Green dots show the corresponding error when using cubic interpolation.

However, one noticeable issue with linear interpolation is the lack of smoothness in convergence. Upon closer inspection, the model has unexpected spikes in error when the number of points in the ϕ grid is a multiple of the number of points in the T grid. This occurs because, when the points in the two grids are multiples of one another, all the points in the T grid align exactly with points in the ϕ grid. In this situation, the interpolation essentially becomes redundant, as the interpolated value will just match the existing point value rather than providing a more nuanced estimate. On the other hand, when the points are slightly offset (i.e., not aligned), linear interpolation works as intended yielding more accurate results. In contrast, cubic interpolation avoids this issue by using multiple neighboring points to estimate the interpolated value. Thus, cubic interpolation produces more consistent and smooth convergence.

Table 4.1 shows the L2 norm of the interface position error using our multiple resolution method (both linear and cubic interpolation), keeping ϵ and N_T constant while increasing N_ϕ as in the plot above. It also shows the L2 error from our previous single grid method, keeping ϵ constant and increasing N (Here, N represents both N_ϕ and N_T). The table verifies our observation that linear interpolation shows better convergence than cubic. Therefore, it makes more sense to continue with linear interpolation even with lack of smoothness in convergence.

N_ϕ	**Linear Interpolation**	**Cubic Interpolation**	**Single Grid**
181	4.212×10^{-3}	6.058×10^{-3}	5.527×10^{-3}
256	3.045×10^{-3}	3.974×10^{-3}	3.126×10^{-3}
362	1.54×10^{-3}	2.956×10^{-3}	1.969×10^{-3}
512	1.297×10^{-3}	2.451×10^{-3}	1.404×10^{-3}
724	9.465×10^{-4}	2.205×10^{-3}	1.113×10^{-3}
1024	8.829×10^{-4}	2.082×10^{-3}	9.987×10^{-4}

Table 4.1: L2 norm of the interface position error for varying ϕ grid resolutions N_ϕ, with the T grid resolution fixed at $N_T = \frac{1}{128}$, in the multiple resolution cases. The plot compares the single grid method with multiple resolution approaches using both linear and cubic interpolation.

The table also shows also highlights the strong correlation between increasing the resolution of the ϕ grid on the convergence of the model. In fact, our multiple resolution method with linear interpolation shows slightly better convergence to the analytical solution compared to the single grid method.

This is a powerful result as the single grid method is considerably more computationally expensive at higher resolutions, due to the need for both the ϕ and T grids to be fine. In contrast, the multiple resolution method, which uses a coarse grid for T and a fine grid for ϕ achieves comparable or even better accuracy while reducing computational costs. As shown in Table 4.2, the multiple resolution method requires approximately half the runtime of the single grid approach in high-resolution cases. While the single grid method may be slightly more efficient at lower resolutions (due to the overhead introduced by interpolation), this cost becomes negligible at higher resolutions. Therefore, for high-resolution convergence studies, the multiple resolution method proves to be the more efficient and practical choice.

N_ϕ	**Multiple Resolution**	**Single Grid**
1024	160.04	298.54
762	42	77.78
512	13.57	18.18
362	3.75	3.76
256	1.52	0.78
181	0.42	0.09

Table 4.2: Run time comparison (in seconds) between single grid and multiple resolution methods for different ϕ grid resolutions N_ϕ, with the T grid resolution N_T fixed at $N_T = \frac{1}{128}$, in the multiple resolution case.

We now conduct the same convergence analysis using a coarse grid for ϕ and a fine grid for T. In Figure 4.4, we kept ϵ constant and used a coarse grid for ϕ with $N_\phi = 128$ through all of our simulations and varied the size of N_T. We also used linear interpolation to interpolate between the T and ϕ grids. In comparison to Figure 4.3, this graph shows little to no convergence. These two results verify that the majority of the spatial convergence comes for a more refined ϕ grid. Therefore, the multiple resolution method has practical use when using a fine ϕ and a coarse T grid.

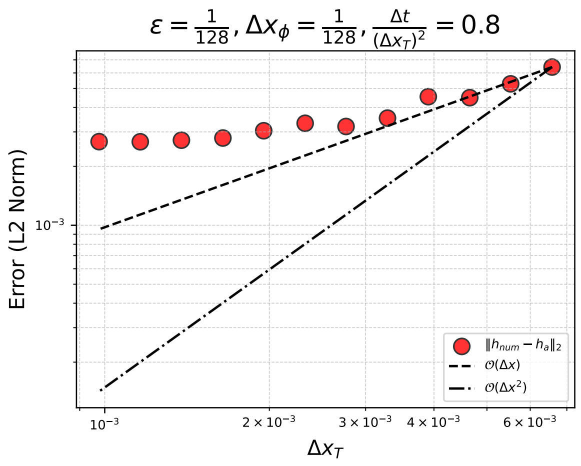

Figure 4.4: Spatial convergence of the model using a multiple resolution method, with $\epsilon = \frac{1}{128}$, $\Delta x_\phi = \frac{1}{128}$ and $\frac{\Delta t}{(\Delta x_T)^2} = 0.8$, while varying the resolution of the T grid Δx_T. Red dots represent the error in interface position relative to the analytical solution.

We previously found that the best convergence occurred when the ratio $\epsilon^* = 2\Delta x$ was maintained. Next, we aimed to explore whether the multiple resolution method could be applied in this context. However, since we have the constraint that $\epsilon > \Delta x$, we needed to approach this carefully.

In Figure 4.5, we kept N_T constant while varying ϵ^*, ensuring that the ratio $\epsilon^* = 2\Delta x_\phi$ was maintained. This approach satisfies the constraint $\epsilon > \Delta x_\phi$, which is equivalent to our previous constraint $\epsilon > \Delta x$ as the ϕ variable holds the information about the interface.

The graph initially shows convergence comparable with the single grid case. However, when the size of the interface (ϵ^*) becomes sufficiently small, the model stops converging to the analytical solution and the model error actually increases. This occurs because the disparity between the interface size and the spacing of the T grid becomes too large. Therefore, the multiple resolution method is not entirely effective when decreasing the ratio $\epsilon^* = 2\Delta x_\phi$ as we might have wished. Nonetheless, it remains valuable for improving computational efficiency, as long as ϵ^* is not significantly smaller than Δx_T.

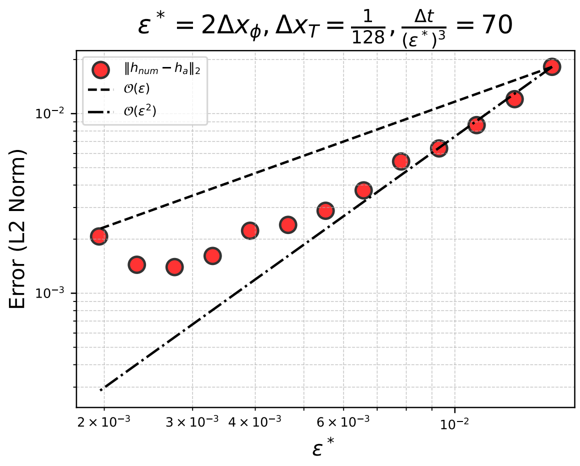

Figure 4.5: Spatial convergence of the model with $\epsilon^* = 2\Delta x_\phi$, $\Delta x_T = \frac{1}{128}$ and $\frac{\Delta t}{(\epsilon^*)^3} = 70$, while varying ϵ^*. Red dots represent the error in interface position relative to the analytical solution.

Chapter 5

2-D Problem

5.1 Problem

Returning to our phase-field equations, we take a deeper look at the A^* parameter, which we previously kept constant. This parameter is defined as $A^* = \frac{\Delta T}{H\gamma}$, where γ denotes the surface energy at the interface. In our earlier one-dimensional analysis, this parameter did not have much of an effect. However, when considering problems involving curvature, A^* becomes more important.

Building on the work of Favier et al. (2019) [21], we extend our analysis to a two-dimensional axisymmetric case. This additional validation is important, as the previous one-dimensional study did not account for curvature effects. To evaluate the convergence of the phase-field model under conditions where curvature plays a significant role, we examine a simplified axisymmetric Stefan problem.

A solid disk of initial radius r_i is immersed inside an infinite fluid domain. The initial temperature distribution is given by $T(r) = \frac{1}{2}\left[1 + \tanh(\beta(r - r_i))\right]$ where r = 0 is the centre of the disk and we let $\beta = 100$. We choose a disk with initial radius $r_i = 0.1$. The solid is mostly at $T = 0$ while the liquid is mostly at $T = 1$. The melting temperature is also fixed at $T_m = \frac{1}{2}$. However, it is important to note that we cannot simulate on an infinite domain, so we restrict the domain to a box of finite size. The boundary conditions for this box are $T = 1$ at all four boundaries.

We now apply the phase-field method to this problem. As this problem is in two dimensions, it adds some complexity to our analysis. We slightly adapt our initial condition for ϕ:

$$\phi = \frac{1}{2}\left[1 + \tanh\left(\frac{r - r_i}{2\epsilon}\right)\right].$$

Here, ϕ now expressed as a function of radius r rather than spatial coordinate x. $\phi = 0.5$

is still the position of the interface. Our grid is now two-dimensional, incorporating both x and y coordinates, rather than being limited to a single spatial dimension (x). The radial coordinate is computed as $r = \sqrt{x^2 + y^2}$.

For this problem, we use the same phase-field equations coupled with temperature. However, we must slightly modify the diffusive terms. Previously, in one dimension, the Laplacian was given by $\nabla^2\phi = \partial_{xx}\phi$. Now in two spatial dimensions, it extends to $\nabla^2\phi = \partial_{xx}\phi + \partial_{yy}\phi$, with the same adjustment applying to T.

This modification effects how we implement our numerical methods. We move back to explicit Euler as it is simpler to implement on this more complex problem. We previously used central finite difference on this diffusive term (alteration to notation, n = time step, i = x-direction, j = y-direction):

$$D_{\phi^n} = \frac{1}{(\Delta x)^2}(\phi^n_{i-1} - 2\phi^n_i + \phi^n_{i+1}).$$

For the two-dimensional problem, we use a five-point stencil scheme on this diffusive term. We also note that, as the problem is axisymmetric, this implies that $N_x = N_y$ and $\Delta x = \Delta y$. For the diffusive ϕ term, the five-point stencil scheme is:

$$D_{\phi^n} = \frac{1}{(\Delta x)^2}(\phi^n_{i-1,j} + \phi^n_{i,j-1} - 4\phi^n_{i,j} + \phi^n_{i,j+1} + \phi^n_{i+1,j}).$$

We implement the remainder of the explicit Euler method in the same manner as previously described. Next, we proceed with the analysis of this two-dimensional problem.

5.2 Results

As stated above, we wished to investigate the effect that curvature has on the model. The Gibbs-Thomson effect describes how the melting temperature of a substance is influenced by its curvature. In the context of phase changes, it essentially states that the melting temperature decreases as the curvature of the interface increases.

With the Gibbs-Thompson effect in mind, we formulate our new condition for temperature at the interface ($r(t) = h$):

$$T = T_m - \gamma\kappa,$$

where κ is curvature and γ is the surface energy. As we are working on a problem with a circle, curvature is inversely proportional to the circle's radius:

$$T - T_m = -\frac{\gamma}{r}.$$

We non-dimensionalise as follows $T^* = \frac{T-T_m}{\Delta T}$ and $r^* = \frac{r}{H}$ to arrive at:

$$T^* = -\frac{1}{r^*}\left(\frac{\gamma}{H\Delta T}\right) = -\frac{1}{A^* r^*}.$$

Here, T^* quantifies how much the interface temperature deviates from the melting temperature. This contrasts the one-dimensional case in which the melting temperature was equal to the interface temperature. Additionally, in the two-dimensional case, the non-dimensional parameter A^* is inversely proportional to melting temperature, which makes it significantly more influential in this context compared to the one-dimensional case. We now wish to investigate how the value of A^* effects the model.

It is important to note that, in most physical systems, the surface energy γ is very small. This implies that a large value of A^* is desirable to accurately capture the underlying physics. We recall the previously discussed constraint that A^* must be at most $\mathcal{O}(\frac{1}{\epsilon^*})$. This poses some challenges when simulating our model. If the value of A^* is too small, the curvature effect will become exaggerated, resulting in an unrealistic solution. Therefore, we aim to maximise A^* while staying within this constraint. However, increasing A^* requires a corresponding decrease in ϵ^* which, due to stability constraints, increases the simulation runtime. Additionally, since we are using the explicit Euler method, our simulation has stricter stability requirements compared to the one-dimensional case where we employed the Crank-Nicolson scheme.

In Figure 5.1, we plot the radius of the solid disc over time. The 'benchmark' solution, taken from Favier et al. (2019) [21], represents the semi-analytical solution associated with no Gibbs-Thompson effect. We compared this benchmark solution with the phase-field model for various values of A^*. We also ensured the constraint $\mathcal{O}\left(\frac{1}{\epsilon^*}\right)$ satisfied by proportionally decreasing ϵ^* when A^* is increased.

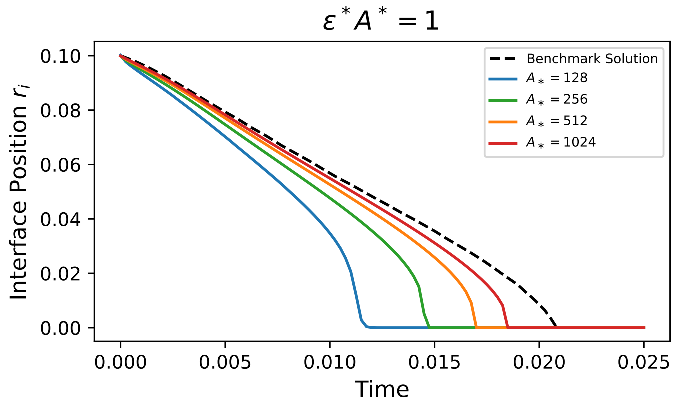

Figure 5.1: Evolution of the interface radius over time for different A^* values, with $\epsilon^* A^* = 1$ held constant.

The initial development is quite similar in all cases, since the interface curvature is relatively low. As the simulation progresses and the disc shrinks (radius decreases), the interface curvature increases, leading to faster melting in cases where A^* is smaller. This behaviour aligns with our theoretical expectations, as smaller A^* values reduce the effective melting temperature, thereby increasing the melting rate.

The graph also shows, that as A^* increases, the model converges towards the benchmark solution. This suggests that in the limit as $\epsilon^* \to 0$ and $A^* \to \infty$, we should recover the condition at the interface without the Gibbs Thompson effect and thus we would obtain the semi-analytical solution. However, as previously discussed, decreasing ϵ^* imposes stricter stability constraints, significantly increasing the model's run time. Future work could focus on implementing a Crank-Nicolson scheme for the two-dimensional case to help address these stability issues.

In Figure 5.2, we plot the radial temperature profile in the domain at different times for various values of A^*. The trend discussed earlier becomes even more apparent here; for cases with smaller A^*, we see the interface temperature drop significantly as the disc melts. Reassuringly, despite such variation in the solid and phase boundary, the temperature profile in the liquid phase remains very similar between all cases.

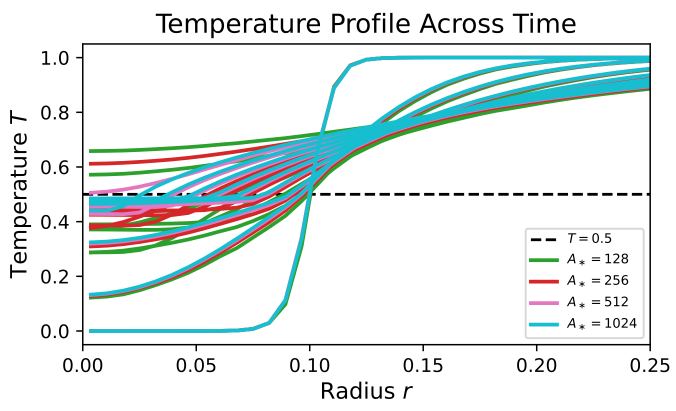

Figure 5.2: Radial profile of temperature in the domain at different times.

Chapter 6

Conclusion

Throughout this report, we have explored various aspects of phase-field models focusing on their stability, convergence, and computational efficiency. Initially, we explored the application of phase-field models to a simple one-dimensional Stefan problem, inspired by the work of Hester et al. (2020) [20]. Due to the lack of an analytical solution, we employed numerical techniques, initially using explicit Euler and subsequently implementing a Crank-Nicolson scheme for improved stability. Additionally, motivated by the work of Favier et al. (2019) [21], we extended the model to a two-dimensional problem to assess the model's accuracy accounting for curvature effects.

A key outcome of this report was the improvement in computational efficiency of the standard phase-field model through the implementation of a multiple resolution method, as opposed to the conventional single grid approach. While multiple resolution methods have been successfully applied to other problems [18], their application with regards to phase-field models appears to be relatively novel within the literature.

By solving for the ϕ variable on a fine grid and in turn solving for T on a coarse grid, we significantly reduced computational costs while maintaining comparable accuracy to the single grid method. However, decreasing ϵ^*, while maintaining the ratio $\epsilon^* = 2\Delta x_\phi$ (which proved most effective for improving convergence in the single grid case), did not show the same improvement in accuracy when using the multiple resolution method. As the difference between the interface size and the spacing of the T grid increased, the accuracy of the model decreased, showing a limitation of this approach.

Throughout this report, the stability of our model parameters has been strongly constrained by the presence of a stiff term in the governing equations. One possible approach to address this would be to implement a fully implicit scheme. While this may be feasible for a simple one-dimensional Stefan problem, it becomes impractical for more realistic, higher-dimensional problems due to the need to solve a large, computationally expensive implicit system.

Another challenge encountered was the inability to achieve full convergence to the analytical solution. This issue is due to error inherent to the model itself rather than any numerical methods we implemented. Future work could involve conducting higher-order asymptotic analysis to determine whether the model can achieve full convergence. Additionally, exploring alternative numerical techniques, such as spectral methods, could provide valuable insights into how the phase-field model compares in terms of stability and convergence. Further research could also include extending the application of the phase-field model to more complex, higher-dimensional, or challenging problems.

In summary, this report successfully demonstrated the potential of multiple resolution methods to improve the computational efficiency of phase-field models. While there are limitations, the approach is a promising direction for efficient simulation of phase change processes, warranting further investigation.

Chapter 7

Appendix

This appendix includes key blocks of code used to generate the data presented in this report. The complete source code and associated data files are included in the accompanying zip file submitted with the report. The variables used in the code blocks below are initialised elsewhere and can be found in the full source code provided.

7.1 Initial Conditions

```python
# Similarity solution parameter
def f_lam(x):
    return np.sqrt(np.pi)*x*np.exp(x**2)*erf(x) - 1.0
lam = root_scalar(f_lam, bracket=[0, 1]).root
# Initial interface position
h = 0.1
# Initial conditions for phi and T
phi = 0.5*(1 + np.tanh((x - h)/(2*eps)))
T = np.zeros(x.size)
T[x<h] = 1.0 - erf(lam*x[x<h]/h)/erf(lam)
```

Listing 7.1: Initial conditions for ϕ and T before asmytotic expansion.

```python
# Similarity solution parameter
def f_lam(x):
    return np.sqrt(np.pi)*x*np.exp(x**2)*erf(x) - 1.0
lam = root_scalar(f_lam, bracket=[0, 1]).root
# Initial interface position
h = 0.1
# Initial conditions for phi
phi = 0.5*(1 + np.tanh((x - h)/(2*eps)))

# Initial conditions for T
T = np.zeros(x.size)
T_1plus = np.log(2 * np.cosh((x[x>h] - h) / (2 * eps))) - (x[x>h] -
    h) / (2 * eps)
T_1minus = np.log(1+np.exp((x[x<h]-h)/eps))
Sv0 = (2*lam**2)/h
T[x<h] = 1.0 - erf(lam*x[x<h]/h)/erf(lam) + eps*Sv0*T_1minus # T0 +
    T1_minus
T[x>h] = eps*Sv0*T_1plus # T1_plus
dphi = np.zeros(x.size)
```

Listing 7.2: Initial conditions for ϕ and T before asmytotic expansion.

7.2 Time-Stepping Functions

```python
def step_time():
    global phi, T
    rhs[:] = dt*D2phi@phi + dt*(
        -De2*phi*(1 - phi)*(1 - 2*phi + A*T) # Compute b
    )
    rhs[0] = 0.0 # Enforce boundary condition
    dphi[:] = iLphi@rhs # M^{-1}b
    phi += dphi
    rhs[:] = T + 0.5*dt*D2T@T + dphi # Compute d
    rhs[0] = 1.0 # Enforce boundary condition
    T[:] = iLT@rhs # C^{-1}b
    return
```

Listing 7.3: Time step for Crank-Nicolson.

```python
def step_time():
    global phi, T
    # Interpolate T grid on to phi grid
    interp_func = interp1d(x_t, T, kind='linear')
    T_interp = interp_func(x_phi)
    rhs_phi[:] = dt*D2phi@phi + dt*(
        -De2*phi*(1 - phi)*(1 - 2*phi + A*T_interp) # Compute b
    )
    rhs_phi[0] = 0.0 # Enforce boundary condition
    dphi[:] = iLphi@rhs_phi # M^{-1}b
    phi += dphi
    # Interpolate phi grid on to T grid
    interp_func2 = interp1d(x_phi, dphi, kind='linear')
    dPhi_interp = interp_func2(x_t)
    rhs_T[:] = T + 0.5*dt*D2T@T + dPhi_interp # Compute d
    rhs_T[0] = 1.0 # Enforce boundary condition
    T[:] = iLT@rhs_T # C^{-1}d
    return
```

Listing 7.4: Time step for Crank-Nicolson using linear interpolation.

```python
def step_time():
    global phi, T
    dT_dx = np.gradient(T, x_t)  # Calculate gradient of T

    # Cubic Hermite Spline interpolation for T
    interp_func = CubicHermiteSpline(x_t, T, dT_dx)
    T_interp = interp_func(x_phi)

    rhs_phi[:] = dt*D2phi@phi + dt*(
        -De2*phi*(1 - phi)*(1 - 2*phi + A*T_interp)  # Compute b
    )
    rhs_phi[0] = 0.0  # Enforce boundary condition
    dphi[:] = iLphi@rhs_phi  # M^{-1}b
    phi += dphi

    dphi_dx = np.gradient(dphi, x_phi)  # Calculate gradient of dphi

    # Cubic Hermite Spline interpolation for dphi
    interp_func2 = CubicHermiteSpline(x_phi, dphi, dphi_dx)
    dphi_interp = interp_func2(x_t)

    rhs_T[:] = T + 0.5*dt*D2T@T + dphi_interp  # Compute d
    rhs_T[0] = 1.0  # Enforce boundary condition
    T[:] = iLT@rhs_T  # C^{-1}d
```

Listing 7.5: Time step for Crank-Nicolson using cubic interpolation.

```python
def phase_field(T, Phi, e, A, dx, dt, t, x):

    D2Phi = np.zeros([len(x), len(t)])
    D2T = np.zeros([len(x), len(t)])

    for j in range(len(t) - 1):
        for i in range(len(x) - 2):
            # Solve for Phi
            D2Phi[i+1, j+1] = (6/(5*A))*(((Phi[i, j]-2*Phi[i+1, j]+
                Phi[i+2, j])/dx**2) \
                             - (1/e**2)*(Phi[i+1,j])*(1-Phi[i+1,j])*(1
                                - 2 * Phi[i+1,j] + A * T[i+1, j]))
            Phi[i+1, j+1] = Phi[i+1, j] + dt * D2Phi[i+1, j+1]

            # Solve for T
            D2T[i+1, j+1] = (1 / dx**2) * (T[i, j] - 2 * T[i+1, j] +
                 T[i+2, j])
            T[i+1, j+1] = T[i+1, j] + dt * D2T[i+1, j+1] + Phi[i+1,
                j+1] - Phi[i+1, j]
    return T, Phi
```

Listing 7.6: Explicit Euler time step function.

```python
def step_time():
    global phi, T
    dphi = np.zeros((x.size, y.size))
    dphi[:,:] +=  dt*(-De2*phi*(1 - phi)*(1 - 2*phi + A*(T-Tm)))
    dphi[:,1:Ny-1] +=  D*dt*( (1/(dy**2))*np.diff(phi,2,axis=1) )
    dphi[1:Nx-1,:] +=  D*dt*( (1/(dx**2))*np.diff(phi,2,axis=0) )
    phi += dphi

    dT = np.zeros((x.size, y.size))
    dT[1:Nx-1,1:Ny-1] += dphi[1:Nx-1,1:Ny-1]
    dT[:,1:Nx-1] +=  dt*( (1/(dx**2))*np.diff(T,2,axis=1) )
    dT[1:Ny-1,:] +=  dt*( (1/(dy**2))*np.diff(T,2,axis=0) )
    T += dT
    return
```

Listing 7.7: Time step for 2-D Problem.

References

[1] J. Donea, A. Huerta, J.-Ph. Ponthot, and A. Rodríguez-Ferran. Arbitrary Lagrangian-Eulerian Methods. In E. Stein, R. de Borst, and T.J.R. Hughes, editors, *Encyclopedia of Computational Mechanics*, volume 1, chapter 14, pages 413–437. Wiley, 2004.

[2] M. Hadzic and S. Shkoller. Well-posedness for the classical Stefan problem and the zero surface tension limit. *Archive for Rational Mechanics and Analysis*, 223(1):213–264, 2017.

[3] J.S. Langer. Models of pattern formation in first-order phase transitions. In *Directions in Condensed Matter Physics: Memorial Volume in Honor of Shang-Keng Ma*, pages 165–186. World Scientific, 1986.

[4] G. Caginalp. An analysis of a phase field model of a free boundary. *Archive for Rational Mechanics and Analysis*, 92:205–245, 1986.

[5] O. Penrose and P.C. Fife. Thermodynamically consistent models of phase-field type for the kinetic of phase transitions. *Physica D: Nonlinear Phenomena*, 43(1):44–62, 1990.

[6] S-L. Wang, R.F. Sekerka, A.A. Wheeler, B.T. Murray, S.R. Coriell, R.J. Braun, and G.B. McFadden. Thermodynamically-consistent phase-field models for solidification. *Physica D: Nonlinear Phenomena*, 69(1-2):189–200, 1993.

[7] A. Karma and W-J. Rappel. Quantitative phase-field modeling of dendritic growth in two and three dimensions. *Physical Review E*, 57(4):4323, 1998.

[8] D. Juric and G. Tryggvason. A front-tracking method for dendritic solidification. *Journal of Computational Physics*, 123(1):127–148, 1996.

[9] V.R. Voller, C.R. Swaminathan, and B.G. Thomas. Fixed grid techniques for phase change problems: a review. *International Journal for Numerical Methods in Engineering*, 30(4):875–898, 1990.

[10] S. Chen, B. Merriman, S. Osher, and P. Smereka. A simple level set method for solving stefan problems. *Journal of Computational Physics*, 135(1):8–29, 1997.

[11] E.W. Hester, C.D. McConnochie, C. Cenedese, L-A. Couston, and G. Vasil. Aspect ratio affects iceberg melting. *Physical Review Fluids*, 6(2):023802, 2021.

[12] L-A. Couston, E.W. Hester, B. Favier, J.R. Taylor, P.R. Holland, and A. Jenkins. Topography generation by melting and freezing in a turbulent shear flow. *Journal of Fluid Mechanics*, 911:A44, 2021.

[13] D. Perissutti, C. Marchioli, and A. Soldati. Morphodynamics of melting ice over turbulent warm water streams. *International Journal of Multiphase Flow*, 181:105007, 2024.

[14] H-R. Liu, C.S. Ng, K.L. Chong, D. Lohse, and R. Verzicco. An efficient phase-field method for turbulent multiphase flows. *Journal of Computational Physics*, 446:110659, 2021.

[15] A. Roccon, F. Zonta, and A. Soldati. Phase-field modeling of complex interface dynamics in drop-laden turbulence. *Physical Review Fluids*, 8(9):090501, 2023.

[16] S.S. Jain. Accurate conservative phase-field method for simulation of two-phase flows. *Journal of Computational Physics*, 469:111529, 2022.

[17] T. Radko. *Double-Diffusive Convection.* Cambridge University Press, 2013.

[18] R. Ostilla-Mónico, Y. Yang, E.P. Van Der Poel, D. Lohse, and R. Verzicco. A multiple-resolution strategy for direct numerical simulation of scalar turbulence. *Journal of Computational Physics*, 301:308–321, 2015.

[19] T. Gotoh, S. Hatanaka, and H. Miura. Spectral compact difference hybrid computation of passive scalar in isotropic turbulence. *Journal of Computational Physics*, 231(21):7398–7414, 2012.

[20] E.W. Hester, L-A. Couston, B. Favier, K.J. Burns, and G.M. Vasil. Improved phase-field models of melting and dissolution in multi-component flows. *Proceedings of the Royal Society A*, 476(2242):20200508, 2020.

[21] B. Favier, J. Purseed, and L. Duchemin. Rayleigh–Bénard convection with a melting boundary. *Journal of Fluid Mechanics*, 858:437–473, 2019.