Dendritic microstructure during rapid solidification

Thesis submitted to

Indian Institute of Science, Bangalore

in partial fulfilment for the award of the degree of

Master of Science(Research)

in

Materials Science

by

Bikramjit Karmakar

(11-01-04-20-92-19-1-17675)

Under the supervision of

Dr. Abhik Choudhury

UG Programme

Indian Institute of Science, Bangalore

Academic Year, 2019-20

June, 2020

DECLARATION

I certify that

(a) The work contained in this report has been done by me under the guidance of my supervisor.

(b) The work has not been submitted to any other Institute for any degree or diploma.

(c) I have conformed to the norms and guidelines given in the Ethical Code of Conduct of the Institute.

(d) Whenever I have used materials (data, theoretical analysis, figures, and text) from other sources, I have given due credit to them by citing them in the text of the thesis and giving their details in the references.

Date: June 25, 2020 (Bikramjit Karmakar)

Place: Bangalore (11-01-04-20-92-19-1-17675)

UG PROGRAMME

INDIAN INSTITUTE OF SCIENCE, BANGALORE

BANGALORE - 560012, INDIA

CERTIFICATE

This is to certify that the project report entitled "**Dendritic microstructure during rapid solidification**" submitted by **Bikramjit Karmakar** (Sr No. 11-01-04-20-92-19-1-17675) to Indian Institute of Science, Bangalore towards partial fulfilment of requirements for the award of degree of Master of Science(Research) in Materials Science is a record of bona fide work carried out by him under my supervision and guidance during Academic Year, 2019-20.

Date: June 25, 2020
Place: Bangalore

Dr. Abhik Choudhury
Materials Engineering
Indian Institute of Science, Bangalore
Bangalore - 560012, India

Abstract

Rapid Solidification refers to phase transformation occurring at conditions far from equilibrium. This happens at high cooling rate or at large undercoolings for which the solidification front advances at high rate, usually > 1 cm/s. Rapid solidification is a important processing technique thereby studying and modelling it bears significant technological importance.

Under the conditions of Rapid Solidification, the approximation low solutal peclet numbers no longer hold. Moreover, there is a transition from diffusion controlled growth. In this study we first determine how that transition happens. Further on, we determine the best phase field operating conditions and the maximum operable length scale. The length scale still being small, requiring the need of developing more efficient solver.

Further, we developed a Adaptive Mesh Refinement based Phase Field Solver, which significantly reduced the computational cost of the solutions. We benchmarked the solutions with a previously developed static mesh phase field solver. In the end, we make modifications to the classical mullins-sekerka instability analysis to incorporate large solutal peclet numbers using Aziz [1] and Galenko [2] form of partition coefficient, and compare the two.

Acknowledgements

Working on my Masters' Thesis during a pandemic was a unique experience. It was undoubtedly a difficult situation, and I couldn't have done it alone. I am indebted to every single essential worker who kept the world running. This time was also a process of becoming comfortable and open to acknowledging, critiquing, and accepting my own privilege.

Foremost, I would like to acknowledge my indebtedness to my supervisor - Dr. Abhik Choudhury, who made this work possible. His friendly guidance and expert advice has been invaluable throughout all stages of the work.

I also express my gratitude to Prof. Abinandanan, my faculty advisor who guided me throughout my college studies.

I also thank the UG Programme and the Dept. of Materials Engineering for the facilities provided. I also thank SERC for the computational facilities.

The members of our lab group have contributed immensely to my time during the project. The group has been a source of friendships as well as good advice and discussions. Thanks to Sumit, Fiyanshu, Bunty, Swapnil, Wadood and extended labmate - Gargeya and Jitin.

I have been fortunate to come across many good friends, without whom life would be bleak, and Janhavi deserves a special mention here. Thanks to Sap, Naskar, Angana and Aritra Da.

I feel a deep sense of gratitude for my parents and sibling without whose unconditional love, emotional and psychological support, this would not have been possible. Maa, Baba, Didi - Thanks.

Contents

List of Figures

List of Tables

Symbols

ϕ	phase field variable
c	overall composition
c^S	solid composition
c^L	liquid composition
D^S	solid diffusivity
D^L	liquid diffusivity
μ	chemical potential
M_ϕ	phase field mobility
δ	interface width
k_E	equilibrium partition coefficient
$k(V)$	non-equilibrium velocity dependent partition coefficient
σ	surface energy
V	velocity of the solidification front
V_{DI}	characteristic velocity for non-equilibrium
V_D	characteristic velocity for complete diffusionless transformation
m_L	Liquidus slope
c_∞	farfield composition
R	gas constant
T	Operating temperature of directional solidification
T_m	melting temperature of the solid
v_m	molar volume
h_α	interpolation function

Chapter 1

Introduction

Computational Models can help to comprehend a complex world that is beyond immediate understanding. It helps to gain new scientific insights and seek technological innovations. The primary purpose of computational modeling is to gain a deeper understanding of experimental systems and use this understanding to improve technology. A secondary goal of computer modeling is to generate a new insight by predicting complex experimental observations accurately. The specific aim of this work is to study dendritic solidification during rapid solidification i.e. a form of non-equilibrium phase transformation. Dendritic solidification at low velocity has been well studied both analytically and computationally[3, 4, 5]. However only a handful of studies have been done to study dendritic solidification during rapid solidification [6, 7]. Rapid solidification processing being an increasing popular method is being put to commercial practice by many industries of materials processing and manufacturing engineering [8]. This makes modelling rapid solidification microstructures of scientific and technologial importance, thereby forming the motivation of this study.

1.1 Rapid Solidification Processing

Rapid solidification processing (RSP) has become an important topic in solidification [8, 9] research and has shown itself to be useful in many potentially interesting applications. One generally understands the term, RSP, to mean the use of high cooling rates or large undercoolings to produce high rates of advancement of the solidification front (typically $V > 1$cm/s). Infact rapid solidification(RS) is a very loosely defined term in scientific literature. Under RS conditions, the low Péclet number approximations which have been developed in the literature for studying the low velocity growth microtructures no longer hold. The assumtion that the diffusion distance is larger than the scale of the microstructure are no longer valid and more general solutions are required. Such solutions are discussed and modified in the later chapter of the thesis.

Rapid growth can occur for one of two reasons: 1. High undercooling of the melt, which can be achieved by slow cooling in the absence of efficient heterogeneous nucleants (bulk undercooling), or by rapid quenching (powder fabrication). 2. Rapidly moving temperature fields, as observed during surface treatment or welding by means of high power density sources such as lasers or electron beams.

Another important type of process for rapid solidification is melt spinning or rotating chill block casting which, depending upon the nucleation behaviour of the melt, can belong to either of the above cases.

1.2 The phase field method

The Phase-field method has become a crucial and an extremely versatile technique for simulating microstructure evolution at the mesoscale since it is the natural way to use the thermodynamic data to study the kinetics of the microstructural evolution

[10, 11]. Because of the diffuse-interface approach, it allows us to check the evolution of arbitrary complex grain morphologies with no presumption on their shape or mutual distribution. It is also straightforward to account for various thermodynamic driving forces for microstructure evolution, like bulk and interfacial energy, elastic potential energy and electric or magnetic energy, and also the effect of various transport processes, like mass diffusion, heat conduction and convection. [12]. It is essentially based on the idea that microstructural interfaces are diffuse at nanoscale and can be represented by smoothly varying order parameter. Phase-field implicitly incorporates curvature-driven physics and handles interface creation which is hard to capture with a sharp interface model. Also, the phase-field equations naturally capture behavior occurring in the bulk away from interfaces making it ideally suited for modeling the complex morphologies that arise in the microstructure study.

Evolution equations for the order parameters are found by applying a variational approach to the free energy functional. When the time derivative is set equal to the divergence of a flux, evolution is governed by the Cahn-Hilliard equation, which applies to conserved order parameters such as composition as in Eq. (1.1)

$$\frac{\partial c}{\partial t} = \nabla \cdot \left(M(c) \nabla \frac{\delta F}{\delta c} \right) \tag{1.1}$$

Setting the time derivative equal proportional to the variational derivative yields the Allen-Cahn equation. This equation does not conserve ϕ and can be used to model phase order parameters:

$$\frac{\partial \phi}{\partial t} = -M(\phi) \frac{\delta F}{\delta \phi} \tag{1.2}$$

1.3 OpenFOAM as a modelling tool in materials science

OpenFOAM is a C++ based toolbox used to develop numerical solvers, typically used for CFD [13]. OpenFOAM implements the components of mesh handling, linear system and solver support, discretisation operators and physical models in library form, where they are re-used over a number of top-level solvers. Implementation of complex physical models follows the idea of mimicking the form of partial differential equations in software. Auxiliary tools, from pre-processing, mesh manipulation, data acquisition, dynamic mesh handling etc. are built into the system. The analogous nature of equations in Phase-Field and CFD, and the ability of OpenFOAM libraries to 'modify' fields opens the possibility of developing highly efficient solvers using the extensive libraries of OpenFOAM. Using such libraries, an efficient parallelized phase field solver involving Adaptive Mesh refinement has been developed as a part of this project.

1.4 Literature Review

Dendrites are the basic microstructural form for most crystalline materials. They express the underlying crystalline symmetry, as well as the growth conditions which existed when the dendrite was formed. They arise due to an instability of the solidification front to random, infinitesimal perturbations. Starting with the morphological stability theory of Mullins and Sekerka [14], the dynamics of pattern selection is now reasonably well understood. The review of the theory is well presented by Langer [15] and Kessler et al.[16]. There is consensus that "microscopic solvability" theory [17, 18] agrees very well with numerical calculations [19, 20] . The comparisons were performed in two dimensions 2D, where accurate time-dependent simulations of dendrite growth are tractable using phase-field methods [21]. All of this was done

in the the low interface velocity regime where small Peclet number approximations were made throughout.

There are two microstructural length scales which are of interest with respect to dendritic growth, the first is the R_{tip} which refers to the radius of the dendrite tip and the second is the spacing between the dendrite primary arms observed during the growth of array of dendrites. R_{tip} is the microstructural scale that is uniquely selected as a function of growth conditions like the undercooling during isothermal growth or the imposed thermal gradient during directional solidification, the anisotropy strength etc. The primary dendrite spacing also depends on processing parameters, however it is not unique, and is known to vary over a range that depends typically on the thermal and compositional history [22]. The work presented here deals with finding the unique R_{tip} during rapid solidification.

1.4.1 Overview of analytical theories for dendritic growth

The theoretical understanding behind the phenomenon of dendrite growth starts with the determination of the diffusion fields in the liquid ahead of a single growing dendrite arm (known as "free dendrite growth"), which is first presented by Ivantsov [23] The solutions by Ivanstsov assume the shape of the dendrite to be given by a parabola in 2D and by a paraboloid of revolution in 3D, and determine the Peclet number $\left(\frac{VR_{tip}}{2D_{th}}\right)$ as a function of the undercooling. Thus, according to above there is exists no unique selection of either the tip velocity (V) or the tip radius (Rtip), but only of the product VR_{tip}, which is in contrast to the experimental observations of Huang and Glicksman [24] and Glicksman [25]. One of the conjectures for the non-uniqueness in the predictions of V and R_{tip} by the solutions in [23, 26] is attributed to the lack of incorporation of capillarity in those theories.

The first calculation of a dynamical evolution of a dendrite was done by Oldfield [27]. Beginning with an Ivantsov shape for the parabola, the temperature field was

computed at each iteration of evolution using the Gibbs-Thomson condition at the interface in combination with the Ivantsov solution far away from the tip of the dendrite. The interface was moved with the speed determined out of the Stefan boundary condition. Oldfield revealed that the interface with surface vitality, was unsteady and had the option to determine just "split tips". Hence, he figured a steadiness rule, coming about because of a harmony between the surface vitality and the annoyance in the warmth condition. This is on indistinguishable lines from calculation of the basic frequencies in the Mullins-Sekerka investigation [14].

Langer and Muller-Krumbhaar [28] formalized the observation by Oldfield, by performing a stability analysis and proving that the needle crystal is unstable for all finite wavelengths of isotropic surface energy. This analysis goes by the name of marginal stability. Essentially, it is statement motivated from the critical wavelength arising in Mullins-Sekerka analysis of a planar front.

Post this, Kessler and Levine [16] put forth a numerical stability analysis of the dendritic interface. The general solutions of this stability analysis give rise to shapes which are cusps, (non-zero slopes at the tip). Only one distinct solution with a smooth dendrite tip exists, which is derived by the microscopic solvability condition. The microscopic solvability criterion [29] also yields a condition of the form $R_{tip}^2 v^* = C(\epsilon)$, where the constant is now a function of ϵ, the strength of anisotropy.

Extensions of the marginal stability criterion have also been proposed for free dendrite growth in alloys. One of the first studies in that regard is by Langer [30] which is continued and elaborated in [31], where the effects of the second component are restricted to dilute concentrations only. A parallel theoretical development by Lipton et al. [32, 33] is found to be in excellent agreement with the experiments in [24, 25] and the theory in [31]. This is extended to include a velocity dependent partition coefficient and a temperature dependent diffusivity for rapid directional solidification of binary alloy dendrites by Kurz et al. [34]. The extension of this theory to directionally solidified dendrites in ternary alloys has been presented by Bobadilla et al.

[35] and the corresponding modification to describe rapid solidification is presented by Rappaz et al. [36].

One of the simplified theory is the LGK theory. Its also the most widely used approximate theories of dendritic growth. The tip is assumed to be a parabola in 2D or paraboloid in 3D, and the contributions, namely, the effect of the Gibbs-Thomson condition, the change in melting point due to the variation of the composition and the imposed thermal field are taken into account. The operating state of the tip of the parabola is assumed to be derived from the marginal stability criterion, wherein the radius of the tip is equated to the minimum wavelength of perturbation that is critical in the Mullin-Sekerka theory of planar front growth. The solutions obtained from the intersection of the Ivantsov solution and the stability criterion derives, the radius and velocity of the tip of the parabola, which are found to be in good agreement with the experimental observations.

1.4.2 History of Dendrite Simulations in Phase-Field

Dendritic simulations date back to the advent of phase field simulations. Beginning with the first large scale simulation of thermal snow flake dendrites by Kobayashi [37], parallel attempts at quantitative comparisons were made by Wheeler [5], where the morphology of dendrite tip is compared to needle crystal solutions proposed by Ivantsov for pure Ni dendrites. Along with this, investigation of the operating state of the dendrite tip and matching with the marginal stability criterion and the micro-solvabilty theories was also carried out. The first simulations of solutal dendrites in a Ni–Cu system are found in the studies of Warren and Boettinger [38] who employ a thermodynamic consistent model for the investigation The phase-field simulations agree well with analytical models and experiments. The thin interface limit for crystallization of pure materials [19, 39] earlier proposed by Karma for pure metals and later extended along with Losert and co-workers [40] for a binary alloy. These works illustrate that one could choose an interface width in the mesoscale range and

still perform phase-field simulations independent of interface kinetics. Thus, quantitative simulations became theoretically possible in the range of low undercooling. However, Dendritic solidification in the whole range of undercoolings still remained a challenge. It has been established that if the thin-interface correction is properly incorporated most phase-field models irrespective of their thermodynamic consistency, converge identically with similar computational costs for a given range of undercoolings. This range is identified as region where the interface peclet number (ratio of the interface width and the diffusion length) is small [41]. With diffusion length during rapid solidification being extremely small the thin interface limit cannot be used. Since, at a given undercooling and strength of anisotropy, the velocity is fixed, the only degree of freedom that remains for adjusting the interface peclet number is the reduction of the interface width. This proves to be computationally expensive for large undercoolings. To combat this effective computational algorithms are needed. Adaptive mesh refinement is one of them, and is used in this study.

1.5 Organisation of the thesis

In Chapter 2, we study transition from diffusion controlled (low velocity growth) in the grand potential based phase field for a range of undercoolings and for different interface widths. This is allows us to determine the extent of deviation from diffusion controlled growth to study the extent of non equilibrium in a during temporal growth. The dependency on interface widths allows us the identify the highest maximum interface width we can work with. At higher the interface width, we can run the simulations at larger length scale i.e. with fewer points. This makes the simulations faster.

In Chapter 3, we develop the parallel load balanced adaptive mesh refinement based phaseField Solver in OpenFOAM. Here we show the unmatched computational gains

from such a solver. We perform low undercooling dendritic simulations using this to demonstrate the efficacy of this solver

In chapter 4, we extended the instability analysis for higher velocity for larger peclet numbers (higher velocity) incroporating various approaches of rapid solidification which also includes a velocity dependent partition coefficient.

Chapter 5 comprises of the conclusions, unfinished planned work, problems faced and future scope of work.

Chapter 2

Non-Equilibrium in Grand Potential model

The Grand-Potential model [42] is a multi-phase field model where the driving is calculated as the difference between the grand potentials of the two phases. The grand-potential formulation has been used in variety of applications in phase field. For rapid solidification, steady state calculations were made in [43]. However for studying dynamic evolution in microstructure, understanding the transition to non equilibrium regime becomes important. This is done by studying the transition from diffusion controlled growth at different driving forces, eventually leading upto rapid solidification.

2.1 The Grand Potential model

The GP model is based on the grand potential functional given by [42]

$$\Omega(T, \boldsymbol{\mu}, \boldsymbol{\phi}) = \int_{\Omega} \left[\Psi(T, \boldsymbol{\mu}, \boldsymbol{\phi}) + \left(\epsilon a(\boldsymbol{\phi}, \nabla \boldsymbol{\phi}) + \frac{1}{\epsilon} w(\boldsymbol{\phi}) \right) \right] d\Omega \tag{2.1}$$

where Ω is the volume of the system and the ϵ the gradient energy coefficient is assumed to be a constant. The phase-field order parameter ϕ is 1 in solid and 0 in liquid. The grand potential can be interpolated by

$$\Psi(T, \boldsymbol{\mu}, \boldsymbol{\phi}) = \sum_{\alpha=1}^{N} \Psi_\alpha(T, \boldsymbol{\mu}) h_\alpha(\boldsymbol{\phi}) \tag{2.2}$$

where

$$\Psi_\alpha(T, \boldsymbol{\mu}) = f_\alpha\left(\boldsymbol{c}^\alpha(\boldsymbol{\mu}, T), T\right) - \sum_{i=1}^{K-1} \mu_i c_i^\alpha(\boldsymbol{\mu}, T) \tag{2.3}$$

The phase evolution equation equation reads as

$$\omega\epsilon\frac{\partial\phi_\alpha}{\partial t} = \epsilon\left(\nabla\cdot\frac{\partial a(\phi,\nabla\phi)}{\partial\nabla\phi_\alpha} - \frac{\partial a(\phi,\nabla\phi)}{\partial\phi_\alpha}\right) - \frac{1}{\epsilon}\frac{\partial w(\phi)}{\partial\phi_\alpha} - \frac{\partial\Psi(T,\mu,\phi)}{\partial\phi_\alpha} \tag{2.4}$$

The double-well potential $w(\phi)$ is

$$w(\phi) = 9\gamma\phi^2(1-\phi)^2 \tag{2.5}$$

We have only two phase i.e; $(\phi_\alpha + \phi_\beta = 1)$.$a(\phi, \nabla\phi)$ is the gradient energy density. In our case a is only function of $(\nabla\phi)$ and has the form

$$a(\nabla\phi) = \gamma\left[a_c(q)\right]^2|q|^2 \tag{2.6}$$

where $q = |\nabla\phi|$ is the normal vector to the interface. γ controls the interfacial energy of the system and is known as surface energy density.a $_c(q)$ describes the anisotropy of evolving phase boundary. In case of 4 fold symmetry $a_c(\nabla\phi) = 1 + \delta\cos(4\theta)$ where θ is the orientation of the interface and is computed in 2D as [44]

$$\tan\theta = \left(\frac{\partial\phi}{\partial y}\right) / \left(\frac{\partial\phi}{\partial x}\right) \tag{2.7}$$

where x and y define the coordinate system. Using above expression for θ, anisotropy can be represented as

$$a_c(\nabla\phi) = 1 - \delta\left[3 - \frac{4\left((\nabla_x\phi)^4 + (\nabla_y\phi)^4 + (\nabla_z\phi)^4\right)}{|\nabla\phi|^4}\right] \tag{2.8}$$

where δ is the strength of aniostropy. $\nabla_x\phi$ is the x -component of $\nabla\phi$ and so on. Since a is only function of $\nabla\phi$, thus from the evolution equation of phase field $\left(\frac{\partial a(\phi,\nabla\phi)}{\partial\phi_\alpha}\right) = 0$. The first term on the right hand side of evolution equation is then written as

$$\frac{\partial a}{\partial\nabla\phi} = 2\gamma a_c(\nabla\phi)\frac{\partial a_c}{\partial\nabla\phi}|\nabla\phi|^2 + 2\gamma\nabla\phi\left[a_c(\nabla\phi)\right]^2 \tag{2.9}$$

The driving force ΔF_α is then written as

$$\Delta F^\alpha = \left[\Psi_\alpha(T,\mu) - \Psi_\beta(T,\mu)\right]\frac{\partial h_\alpha(\phi)}{\partial\phi_\alpha} \tag{2.10}$$

where $h_\alpha(\phi)$ is an interpolation function given as

$$h(\phi) = \phi^2\left(6\phi^3 - 15\phi + 10\right) \tag{2.11}$$

Its important to note that the linearization of driving force is don't here as in [42], because such a approximation is only valid for smaller driving forces.

Calculations for growth into an undercooled melt starting from a small solid ($\phi = 1$) seed with appropriate levels of anisotropy typically exhibit needlelike growth with parabolic growth forms in the directions with the smallest values of σ. Calculations performed using a coarse mesh will usually exhibit side branching typical of real dendrites because discretization errors introduce noise into the calculation. (A coarse square mesh can also induce a synthetic four- fold symmetry.) As the computational mesh is refined, however, side branches necessarily disappear. Therefore, noise has

been introduced at controlled levels to induce side branching in most simulations. This is typically implemented using random fluctuations of a source term added to the phase-field equation that is localized to regions where ϕ is between 0 and 1; i.e., in the interfacial region.This permits the inclusion of nucleation processes into simulations using the phase-field method. Thus, phase field equation is modified by adding a noise term which is given as

$$\frac{\partial \phi}{\partial t} \rightarrow \frac{\partial \phi}{\partial t} - 6rM\phi^2(1-\phi)^2 \tag{2.12}$$

where r is a random number distributed uniformly between 0 and 1, and a new number is generated for every point of the grid, at each time-step. M is an amplitude of fluctuations which in our case is 0.003

The final evolution equation reads as

$$\begin{aligned} \omega\epsilon\frac{\partial \phi_\alpha}{\partial t} = \epsilon\nabla\cdot\frac{\partial a(\nabla\phi)}{\partial \nabla\phi_\alpha} - 18\frac{\gamma_{\alpha\beta}}{\epsilon}\phi(1-\phi)\left(1-2\phi_\alpha\right) \\ \left[\Psi_\alpha(T,\mu) - \Psi_\beta(T,\mu)\right]\frac{\partial h_\alpha(\phi)}{\partial \phi_\alpha} + 6rM\phi^2(1-\phi)^2 \end{aligned} \tag{2.13}$$

This model instead of solving evolution equation for concentration fields, solves directly for the thermodynamic variable μ, which relate the phase concentrations c_i^α instead of solving for phase concentrations themselves. This is possible because the concentrations $c_i^\alpha(\mu, T)$ are written as explicit functions of the thermodynamic variable μ. The evolution equation for chemical potential can be written as:

$$\begin{aligned} &\left(\tfrac{\partial c^\alpha(\mu,T)}{\partial \mu}h_\alpha(\phi) + \tfrac{\partial c^\beta(\mu,T)}{\partial \mu}\left[1-h_\alpha(\phi)\right]\right)\tfrac{\partial \mu}{\partial t} \\ &= \nabla\cdot\Big[\Big(D^\alpha g_\alpha(\phi)\tfrac{\partial c^\alpha(\mu,T)}{\partial \mu}\Big) \\ &\quad + D^\beta\left[1-g_\alpha(\phi)\right]\tfrac{\partial c^\beta(\mu,T)}{\partial \mu}\Big)\nabla\mu\Big] \\ &- \left[c^\alpha(\mu,T) - c^\beta(\mu,T)\right]\tfrac{\partial h_\alpha(\phi)}{\partial t} \end{aligned} \tag{2.14}$$

The unknown terms from the above equations can be determined from the free-energy function The free energy functions we use are

$$f_L = \frac{RT}{V_m}\left\{c_L \ln\left(c_L\right) + \left(1 - c_L\right)\ln\left(1 - c_L\right)\right\} \tag{2.15}$$

$$f_S = \frac{RT}{V_m}\left\{c_S \ln\left(c_S\right) + \left(1 - c_S\right)\ln\left(1 - c_S\right) - c_S \ln\left(k_e\right) + \left(1 - c_S\right)\ln\left[\frac{1 + \frac{(T_m - T)}{m_e}}{1 + \frac{k_c(T_m - T)}{m_c}}\right]\right\} \tag{2.16}$$

The composition c_S and c_L can be written as explicit functions of μ where $\mu = \frac{df}{dc}$ Thus

$$c_S(\mu) = \frac{1}{\frac{1}{k_e Z}\exp\left(-\frac{\mu}{A}\right) + 1}, \quad c_L(\mu) = \frac{1}{\exp\left(-\frac{\mu}{A}\right) + 1}; \tag{2.17}$$

where

$$A = \frac{RT}{v_m}, \quad Z = \left[\frac{1 + \frac{(T_m - T)}{m_e}}{1 + \frac{k_c(T_m - T)}{m_c}}\right]; \tag{2.18}$$

2.2 Transition from diffusion Controlled growth

The growth kinetics of a interface which involves long range solute diffusion can be modeled by solving a diffusion problem. Such a growth is called diffusion controlled growth. Such a growth is parabolic in time, i.e. position of the interface is parabolic in time $x \propto \sqrt{t}$ or $v \propto \frac{1}{\sqrt{t}}$. However, such a relationship doesn't hold at rapid solidification. This has been observed by Wei and Herlach in their [45], showing a transition happens at rapid solidification. It has also been shown by Bhadeshia [46] that local equilibrium, which does not occur during rapid solidification is needed for diffusion controlled growth. Thus this acts as an indicator of the correctness of our phase field model length scale.

2.3 Results: Determining the maximum possible length-scale

A higher interface width allows us to work at higher length scale and thereby less computational nodes. This allows the simulation to be faster. However, one cannot have indefinitely increase the interface width. Increasing the interface width leads to unstable interface causing phenomenon like interface expansion. The maximum permissible interface is thus determined by working at different interface widths, making sure the interface is stable at that width. We also ensure that parabolic growth as in diffusion controlled growth is consistent at lower driving forces. For this we plot the n in $v \propto t^n$ at different interface widths for a Cu-0.07Ni alloy. We simulate planar interface growths at different interface widths for the alloy, and find the exponent n for each case, which should be -0.5 for diffusion controlled growth regime. The interface is non-dimensionalized with the interface width at maximum interface width for which $n = -0.5$. For all the simulations, the number of computational nodes inside the interface remains same, and the physical length is changed. The data in Table 2.1 is used for simulations from [47]. We develop a parallelized OpenFOAM solver for the same. To ensure that there is no interface kinetics at small driving forces we appropriately determine the ω using

$$\omega = \epsilon \frac{\left[c^{\beta}\left(\mu_{eq}, T\right) - c^{\alpha}\left(\mu_{eq}, T\right)\right]^2}{\left(D^{\beta}\right) \frac{\partial c^F(\mu^0, T)}{\partial \mu}} (M + F) \tag{2.19}$$

We also plot the transition from diffusion-controlled regime at higher driving forces. For this we plot the velocity exponent for different supersaturation in figure 2.2 The interface width determined from above is used here.

TABLE 2.1: Thermophysical and Simulation data for Cu-0.07Ni

T_A	1728 K
T_B	1358 K
σ	2.8×10^{-5} J cm^{-2}
D_L	10^{-5} cm^2 s^{-1}
D_S	10^{-10} cm^2 s^{-1}
k_E	0.7965
c_∞	0.0717441
m_L	-310.9 K

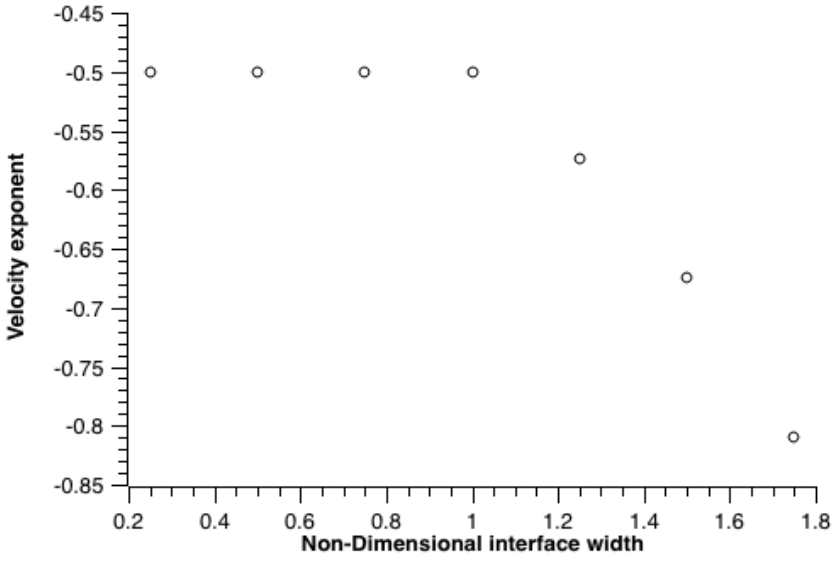

FIGURE 2.1: Variation in velocity exponent for different interface widths

2.4 Discussion

2.4.1 Maximum interface width and transition

The dependence of diffusion controlled growth coefficient with interface width is plotted as in Figure 2.1. The maximum interface width value at which the coefficient

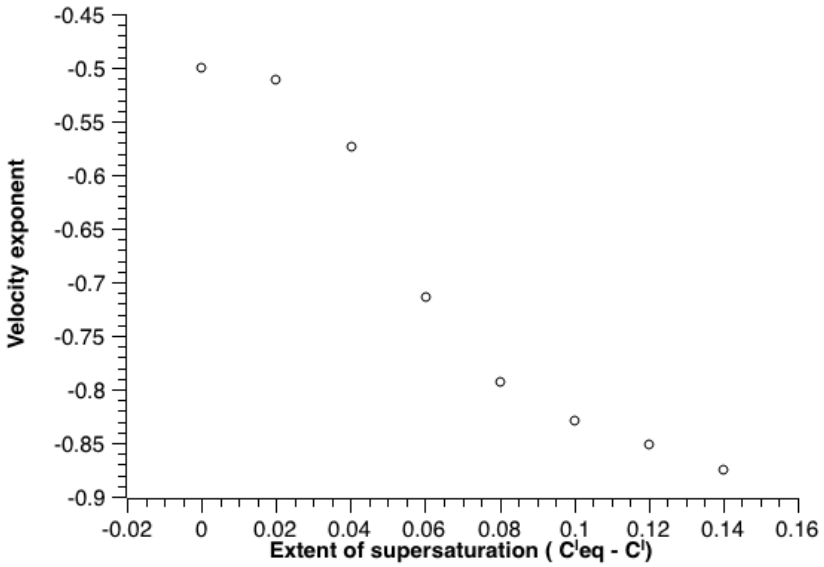

FIGURE 2.2: Variation in velocity exponent for different supersaturation

becomes -0.5 becomes the working interface width for simulation for this system. As we reduce the interface width further we see that this value is converged. Figure 2.2 shows the variation of the velocity exponent with increasing supersaturation at the maximum determined interface width. It is seen that the values initially slowly change followed by rapid change, and then again followed by slowing down of the change. This set of simulations also helps us to ensure that the computational interface is stable at large range of driving forces.

2.4.2 The thin-interface limit and the anti-trapping current

To work with higher interface thickness Karma and his group came up with the thin interface limit [48] and used the anti trapping current to eliminate the chemical potential jump at the interface. This allowed working a higher interface widths with quantitatively reliable phase-field simulation. However, such cannot be used during rapid solidification since trapping here is a naturally occurring phenomenon. Therefore we have to work at the diffuse interface limit, at lower interface widths, making the the simulations computationally expensive.

2.4.3 Computational Cost

The are two main reasons contributing to the high computational cost of simulations. The first being the high degree of non-linearity in the governing equations. Logarithmic free energies are used, which is otherwise assumed to be a parabola in the near equilibrium limit. This contributes to the stiffness of the equation. The next being the smaller length scale. Working in a smaller length scale requires us to use much more computational nodes. Our 1D simulation took about 10^6 points. This is extremely computationally expensive, and makes 2D simulations impractical. To overcome this we develop a phase field solver with adaptive mesh refinement(AMR). subfig

Chapter 3

Load balanced AMR phase field solver in OpenFOAM

3.1 Introduction to Adaptive Mesh Refinement

For numerical solution of partial differential equations (PDE) a discrete domain is chosen where the PDEs are solved. Usually, The spacing between these grid points are uniform. The spacing of the grid points determines the local error and hence the accuracy of the solution. The spacing also determines the number of calculations to be made to cover the domain of the problem and thus the cost of the computation. Finer grid spacing leads to lower error and higher accuracy, but significantly increases the computational cost.

The first way to improve this to start with a variable mesh, i.e. refined in the regions of high chemical potential gradient. However, with a moving interface, the chemical potential gradients at different regions change quickly. Thereby, this renders a static variable mesh approach of little to no use in terms of gain in computational cost.

In the adaptive mesh refinement(AMR) technique we start with a base coarse grid. As the solution proceeds we identify the regions requiring more resolution by some

parameter characterizing the solution. The finer grids are super imposed in this region, and finer and finer subgrids are added recursively until the desired level of refinement is reached. The values on the new grids are computed by interpolation of from the older points using appropriate interpolation schemes.

3.2 Adaptive Mesh Refinement in OpenFOAM

OpenFOAM consists of two librarires which handles madpative mesh refinement. They are called `dynamicFvMesh` [49] and `topoChangerFvMesh` [50]. `dynamicFvMesh` handles mesh motion without topological changes, and `topoChangerFvMesh` handles mesh motion together with topological changes. Topological changes here, means that the connection between the cells changes, whereas normal mesh motion are just points changing position.

```
$FOAM_SRC/dynamicFvMesh/
|-- dynamicFvMesh
|-- dynamicInkJetFvMesh
|-- dynamicMotionSolverFvMesh
|-- dynamicRefineFvMesh
|-- include
|-- lnInclude
|-- Make
|-- solidBodyMotionFvMesh
+-- staticFvMesh
```

```
$FOAM_SRC/topoChangerFvMesh/
|-- linearValveFvMesh
|-- linearValveLayersFvMesh
|-- lnInclude
|-- Make
|-- mixerFvMesh
|-- movingConeTopoFvMesh
|-- rawTopoChangerFvMesh
+-- topoChangerFvMesh
```

FIGURE 3.1: Directory Structure of the dynamicFvMesh and topoChangerFvMesh libraries

Our simulations does not involve topological changes of any kind. Thereby making our interests focused to `dynamicFvMesh`.

The most basic one is the `dynamicMotionSolverFvMesh` class which just solves the equations for mesh motion and move cells according to a diffusivity model. The `dynamicInkJetFvMesh` class is is specifically designed for the pumping system of an inkjet injector. The `solidBodyMotionFvMesh` moves the mesh according to a

prescribed motion function. The `dynamicRefineFvMesh` class provides adaptive mesh refinement during run-time. It marks a cell for refinement if the cell value of a user specified field is within a certain range. If the a cell is marked for refinement it calls the mesh cutter `hexRef8` and split the cell into eight new cells. This is also known as oct-tree refinement.

3.2.1 `dynamicRefineFvMesh` Class

The `dynamicRefineFvMesh` refines the cell based on the value of the field, i.e. the regions in space having value within a specified range would be refined to a set value. With refinement occurring at only one specified range, this prevents having different levels of refinement for different ranges. While using `dynamicRefineFvMesh` does provide improvement over static meshing, in an ideal setting one wishes to have multiple refinement regions, with multiple refinement levels. This was achieved by using the open-sourced class `dynamicInterfaceRefineFvMesh` class developed by Holzmann [51].

3.2.2 `dynamicInterfaceRefineFvMesh` Class

The `dynamicInterfaceRefineFvMesh` can be used to set refinement level for two independent ranges. This allows additional flexibility of setting very fine mesh at the interface. In the bulk the meshing near the interface is kept higher than the meshing far away from the interface. This is done by introducing buffer layers, which adds a set number of layer usually between 2-5 while transitioning from one refinement level to the other. In addition to that in meshes with more than two refinement levels it is important to ensure a smooth transition between the different levels in order to sufficiently decrease discretization errors due to mesh skewness at refinement transitions and to provide a buffer between two refinement levels for the computed flow to adapt to the new mesh level.

3.2.3 Additional computational cost due to re-meshing

Creating new points in the mesh and interpolating all the fields to the new mesh is a computationally heavy process, thereby it is imperative to consider the trade-off involved. Re-meshing at every timestep is a redundant, time consuming process. However, determination of the timestep interval for which re-meshing would be necessary to avoid affecting the accuracy cannot be known apriori. To accomplish this we set up a dynamic feedback loop using the phase-field order parameter bound which is bound between 0 and 1, and the number of new cells created. To persevere the accuracy of the solution we check the *max* and *min* of the order parameter at every time step and enforce meshing whenever it goes slightly beyond 1 or below 0. In addition to that we check number of new cells created in the last refinement. Smaller the number, higher the refinement interval imposed.

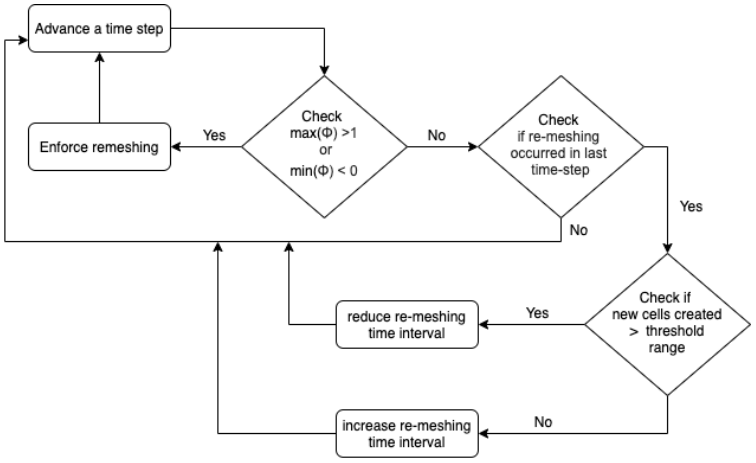

FIGURE 3.2: Flow chart for minimizing the re-meshing computational load

3.2.4 Load Balancing

In OpenFOAM parallelization is based on domain decomposition. The simulation domain is decomposed into sub-domains, each being assigned to a different processor. When adaptive mesh refinemesnt(AMR) is used in parallel, the processor with the largest load becomes a bottleneck when processed data is required by other waiting

processors. Several methods to estimate the load balance can be thought of. Here we use the file size generated by each of the files. The larger the range of max and min file sizes from processor, greater is the load imbalance. Due to unavailability of any dynamic load balancing(DLB) classes in OpenFOAM, we repurpose an inbuilt domain decomposition class in OpenFOAM named `scotch`. Scotch decomposition obviates the need of any geometrical parameter input, and attempts to minimize the number of processor boundaries. This allows us to minimize the standard deviation of number of cells per processor. It is also important to mention that load balancing involves pausing and restarting the simulation (creating the mesh, and allocating all the cells to processors) and thereby involves a fairly high CPU Time. Therefore, we load balance only when the imbalance is greater than 10%

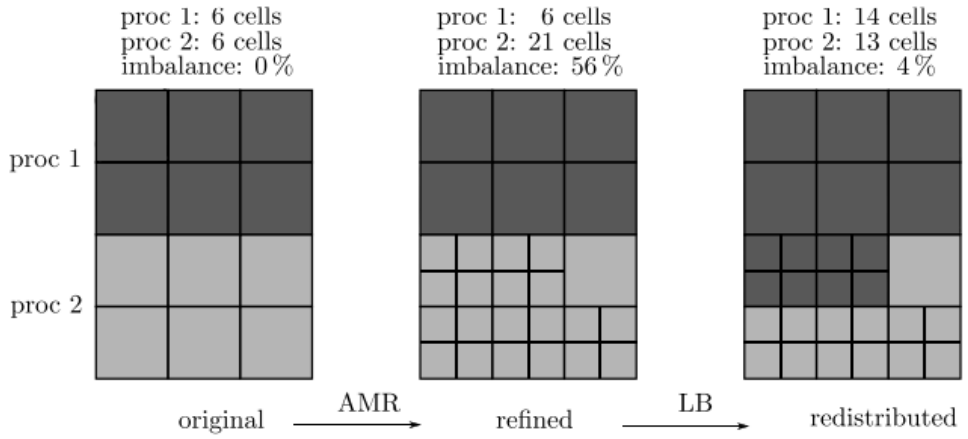

FIGURE 3.3: Representation of load-balancing

3.3 Results: Benchmarking

To benchmark the accuracy and effectiveness of the solver developed we run a set of free dendrite microstructure simulation. The R_{tip} and V_{tip} predicts the accuracy, and the CPU time predicts the gain in computational efficiency. The comparison is made between solver developed herein and phase-Field solver developed by R.K. Singh [52] using identical systems. The Phase field model used has been described

in chapter 2. However, these simulations are done at small undercooling, hence the low interface velocity and in the presence of anti trapping current.

TABLE 3.1: Thermophysical and Simulation data

Slope liquidus (m_l)	0.45
Relaxation coefficient (ω)	1.687
Surface energy (γ)	1
Interface width parameter (ϵ)	48
Composition of solid (c_{Sol})	0.78
Composition of liquid (c_{Liq})	0.45
Equilibrium composition (c_{eq})	0.50
Anti trapping coefficient(j)	0.35355
Diffusivity in liquid (D_l)	1
Strength of anisotropy (δ)	0.02
Melting Temp (T_0)	1

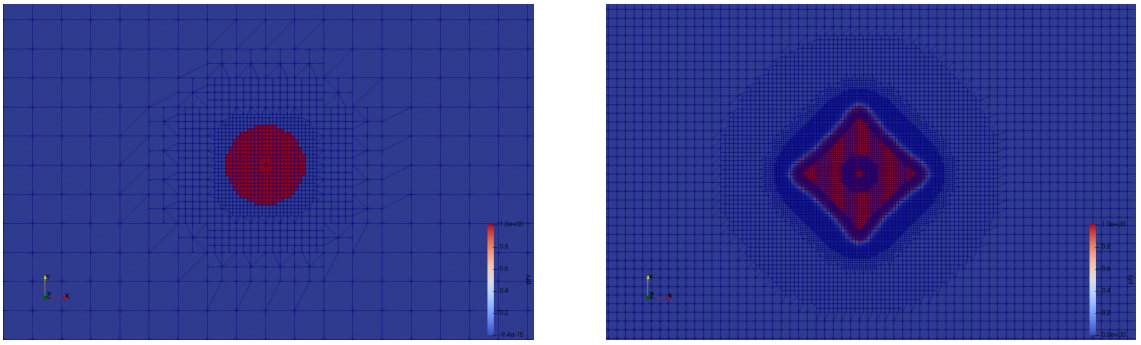

FIGURE 3.4: Progression of adaptive meshing (left figure at smaller scale for representing the meshing clarity

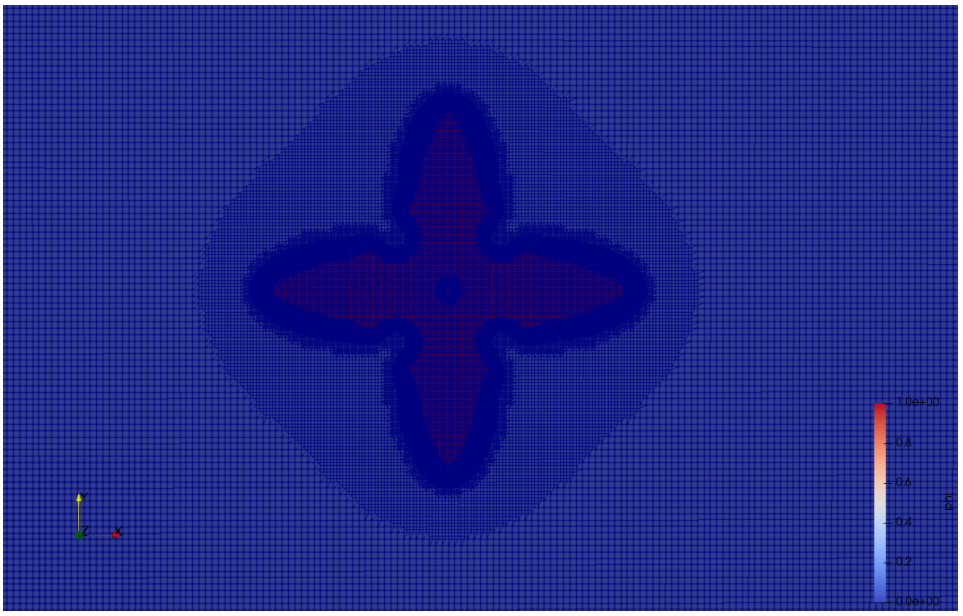

FIGURE 3.5: Zoomed out view of meshing

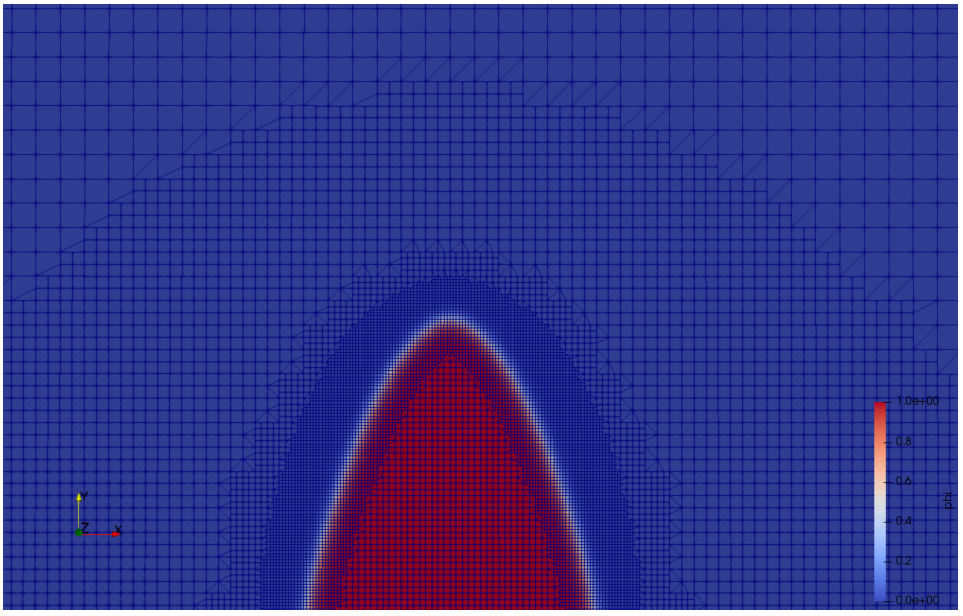

FIGURE 3.6: Meshing near the dendrite tip

3.4 Discussion

As can seen from Table 3.2 there is absolutely no loss of accuracy when using AMR, the radius being exactly same in both cases. In addition to that there is an extremely high degree of computational time gained. A free dendrite which was taking about a

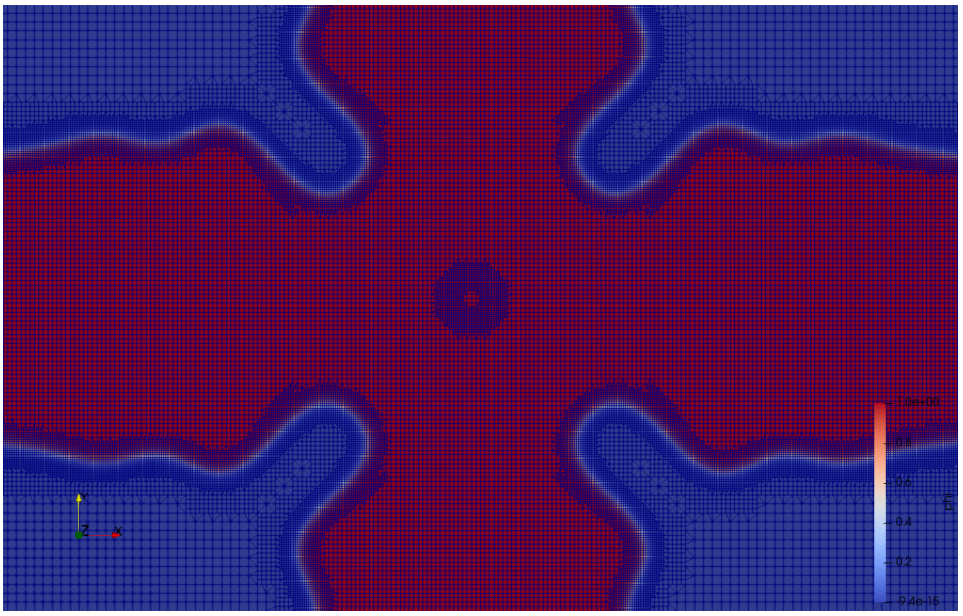

FIGURE 3.7: Meshing interaction near two dendrite arm

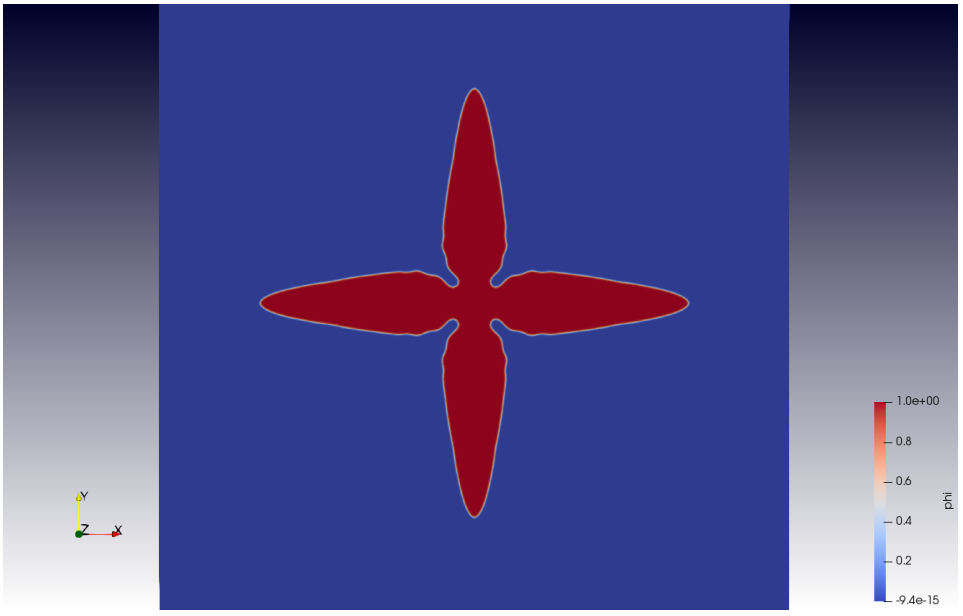

FIGURE 3.8: Phase field profile without meshing lines

day in HPC cluster, can be done under 3 hours in a desktop computer. This opens the possibility of doing of doing many large-scale which was otherwise computationally intensive to perform. This is also significantly promising to reduce the computational cost for our rapid solidification simulations.

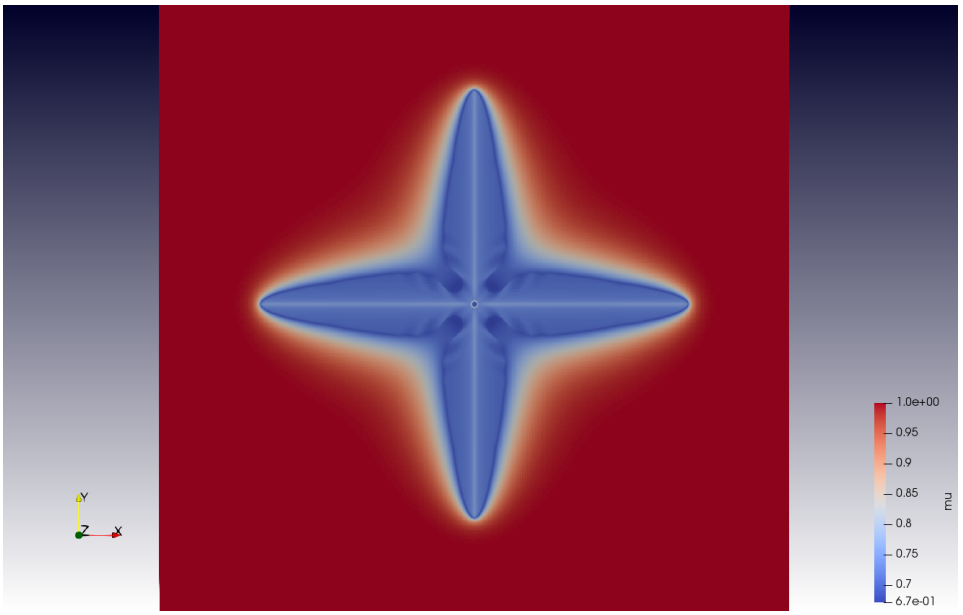

FIGURE 3.9: Chemical potential without meshing lines

TABLE 3.2: Comparison of R_{Tip}

ΔT	R_{Tip} without AMR	R_{Tip} with AMR
0.065	230	230
0.07	188	188
0.075	146	146

TABLE 3.3: Comparison of CPU time with and without AMR

ΔT	96 Core CPU time without AMR	6 core CPU time with load-balanced AMR
0.065	69842 s	9331 s
0.07	67412 s	9103 s
0.075	64793 s	8466 s

Chapter 4

Revisiting Mullins Sekerka Instability analysis - modifications and insights

4.1 Introduction to Interface Stability analysis

Tthe classical Interface stability criterion is derived for low solutal peclet numbers, making it insufficient in the case of rapid solidification. The most notable extension to this was made by Kurz et. al [34] to include the Aziz's [1] velocity dependence of the partition coefficient for rapid solidificaiton. However, that velocity dependence has evolved through literature most notably through a series of works by P. Galenko and Sobolev [2, 53]. In this chapter, we derive the interface stability criterion in for rapid solidification, and make relevant modifications to include Galenko's form of partition coefficient. We also study its implications on the dendritic microstructures forming during rapid solidification.

4.2 High velocity interface stability analysis

To keep the results general we need to solve for both solute and heat diffusion. Here the solute diffusion in the solid phase is neglected because of low diffusivity.

$$\frac{\partial^2 C}{\partial x^2} + \frac{\partial^2 C}{\partial z^2} + \frac{V}{D}\frac{\partial C}{\partial z} = 0 \tag{4.1}$$

$$\frac{\partial^2 T_l}{\partial x^2} + \frac{\partial^2 T_l}{\partial z^2} + \frac{V}{a_l}\frac{\partial T_l}{\partial z} = 0 \tag{4.2}$$

$$\frac{\partial^2 T_s}{\partial x^2} + \frac{\partial^2 T_s}{\partial z^2} + \frac{V}{a_s}\frac{\partial T_s}{\partial z} = 0 \tag{4.3}$$

Here D is the liquid phase solute diffusivity, and a_s and a_l are the thermal diffusivities for solid and liquid respectively. Here the growth direction is z. The general solutions for the above equations are as follows

$$C = C_0 - \frac{DG_c}{V}\exp\left[-\frac{V_z}{D}\right] \tag{4.4}$$

$$T_l = T_0 + \frac{G_l a_{[1}}{V}\left[1 - \exp\left(-\frac{V_z}{a_l}\right)\right] \tag{4.5}$$

$$T_s = T_0 + \frac{G_s a_s}{V}\left[1 - \exp\left(-\frac{V_z}{a_s}\right)\right] \tag{4.6}$$

where G_c, G_l, and G_s are solute and temperature gradients (liquid and solid) respectively at the plane interface. Note that C_0 is the concentration at infinity, but T_0 is temperature at the interface.

According to perturbation theory the disturbances/ noise at the interface can be assumed of a sinusoidal form.

$$C = C_0 - \frac{DG_c}{V}\exp\left(-\frac{V_z}{D}\right) + A\delta\sin\omega x\exp\left(-b_c z\right)) \tag{4.7}$$

$$T_l = T_0 + \frac{G_l a_l}{V}\left[1 - \exp\left(-\frac{V_z}{a_l}\right)\right] + B\delta\sin\omega x\exp\left(-b_l z\right) \tag{4.8}$$

$$T_s = T_0 + \frac{G_s a_s}{V}\left[1 - \exp\left(-\frac{V_z}{a_s}\right)\right] + R\delta\sin\omega x\exp\left(-b_s z\right) \tag{4.9}$$

where A, B, R, b_c, b_l, and b_s have to be determined by applying the various physical constraints.

b_c, b_l, and b_s can be determined by inserting the derivatives of the individual equations into each of the equations (details are presented in Appendix A)

$$b_c = \frac{V}{2D} + \left[\left(\frac{V}{2D}\right)^2 + \omega^2\right]^{1/2} \tag{4.10}$$

$$b_l = \frac{V}{2a_l} + \left[\left(\frac{V}{2a_l}\right)^2 + \omega^2\right]^{1/2} \tag{4.11}$$

$$b_s = -\frac{V}{2a_s} + \left[\left(\frac{V}{2a_s}\right)^2 + \omega^2\right]^{1/2} \tag{4.12}$$

To find the value of R we use the fact that at the interface the value of temperature for both solid and liquid is equal. We also use a shorthand $S = \delta\sin\omega x = z$

$$\begin{aligned} &T_0 + \tfrac{G_l a_l}{V}\left[1 - \exp\left(-\tfrac{VS}{a_l}\right)\right] + B\,\mathrm{Sexp}\left(-b_c S\right) = \\ &= T_0 + \tfrac{G_s a_s}{V}\left[1 - \exp\left(-\tfrac{VS}{a_s}\right)\right] + R\,\mathrm{Sexp}\left(-b_s S\right) \end{aligned} \tag{4.13}$$

since δ is very small, we can expand in terms of $O(\delta)$ to get $\exp(-\delta \sin \omega x) \approx 1 - \delta \sin \omega x$

This gives us

$$R = (G_l - G_s) + B \tag{4.14}$$

The argument we use next is that the temperature at the front of the perturbed interface is given by the total undercooling.

$$T^+ = T_f + mC^+ - \Gamma\omega^2 S \tag{4.15}$$

Here T^+ is the temperature at the front of the perturbed interface, and T_f is the melting point. As a general rule hereon quantities with superscript $_+$ are calculated at the front of the perturbed interface.

Joining Equation 4.7, 4.13 and 4.15 we get

$$\begin{aligned} &T_0 + \tfrac{G_l a_l}{V}\left[1 - \exp\left(-\tfrac{VS}{a_l}\right)\right] + B\,\mathrm{Sexp}\left(-b_l S\right) = \\ &\quad = T_f + m\left[C_0 - \tfrac{DG_c}{V}\exp\left(-\tfrac{VS}{D}\right) + AS - \exp\left(-b_c S\right)\right] - \Gamma\omega^2 S \end{aligned} \tag{4.16}$$

The smallness of δ allows us to taylor expand the exponentials in the above equation, which gives us

$$T_0 + G_l S + BS = T_f + mC_0 - \frac{mDG_c}{V} + mG_c S + mAS - \Gamma\omega^2 S \tag{4.17}$$

Using $T_0 = T_f + mC_0 - mDG_c/V$, one can write,

$$G_l S + BS = mG_c S + mAS - \Gamma\omega^2 S \tag{4.18}$$

Cancelling S throughout, we get,

$$B = mG_c + mA - \Gamma\omega^2 - G_l \tag{4.19}$$

Using Stefan's condition, and using the fact that the solute and heat equations should correspond to the same growth rate

$$\frac{\kappa_s G_s^+ - \kappa_l G_l^+}{\Delta h_f} = \frac{DG_c^+}{(k-1)C^+} \tag{4.20}$$

κ_s and κ_l are solid and liquid thermal conductivities respectively

The following equations can be derived by taking the derivatives of the equation 4.7, 4.8 and 4.9 with the growth direction z respectively and substituting the corresponding gradient in the LHS

$$G_c^+ = G_c - \frac{G_c}{D} VS - b_c AS \tag{4.21}$$

$$G_l^+ = G_l - \frac{G_l}{a_l} VS - b_l BS \tag{4.22}$$

$$G_s^+ = G_s - \frac{G_s}{a_s} VS - b_s RS \tag{4.23}$$

Substituting S for z in equation 4.7 and expanding for $O(\delta)$ we get

$$C^+ = C_0 - \frac{DG_c}{V} + G_c S + AS \tag{4.24}$$

Substituting equations 4.21, 4.22, 4.23, 4.24 in equation 4.20

$$n_1 + n_2 VS + (\kappa_l b_l B - \kappa_s b_s R)\, S = \frac{\Delta h_f}{(k-1)} \cdot \frac{DG_c - G_c VS - Db_c AS}{C^* + G_c S + AS} \tag{4.25}$$

where

$$n_1 = (\kappa_s G_s - \kappa_l G_l)\,, \quad n_2 = \left(\frac{\kappa_l G_l}{a_l} - \frac{\kappa_s G_s}{a_s}\right)$$

Multiplying throughout by the RHS denominator and ignoring S^2 terms we get

$$\begin{aligned} n_1 C^* + n_1 G_c S + n_1 AS + n_2 VC^* S + (\kappa_l b_l B - \kappa_s b_s R)\, C^* S = \\ = \frac{\Delta h_f}{k-1}\left(DG_c - G_c VS - Db_c AS\right) \end{aligned} \tag{4.26}$$

Using $n_1 C^+ = \frac{\Delta h_f}{k-1} DG_c$ from equation 4.20 and dividing throughout by S we get,

$$n_1 G_c + n_1 A + n_2 VC^* + (\kappa_l b_l B - \kappa_s b_s R)\, C^* = \frac{\Delta h_f}{p}\left(G_c V + Db_c A\right) \tag{4.27}$$

where $p = 1 - k$

By substituting for R from equation 4.14 and B from equation 4.19

$$A = \frac{-n_1\left(\frac{VC^*}{D} + G_c\right) - n_2 VC^* - n_3 C^*\left(\phi - \omega^2\Gamma\right) + \kappa_s b_s\left(G_l - G_s\right) C^*}{n_1 + n_3 m C^* + n_1\left(\frac{C^*}{G_c}\right) b_c} \tag{4.28}$$

where $n_3 = \kappa_l b_1 - \kappa_s b_s$ and $\phi = mG_c - G_l$

This concludes us with finding the unknowns.

Without any loss of generality we can assume the velocity of the perturbed interface of the form

$$V + \dot{\delta} \sin \omega y \tag{4.29}$$

This is equal to the velocity from the heat conservation equation 4.20 . Inserting the perturbed interface values from equation 4.21 and 4.22 in 4.20 we get

$$\begin{aligned} V + \dot{\delta} \sin \omega y = \\ = \frac{(\kappa_s G_s - \kappa_l G_l) + \left(\frac{\kappa_l G_l}{a_l} - \frac{\kappa_s G_s}{a}\right) V \delta \sin \omega y + (\kappa_l b_l B - \kappa_s b_s R) \delta \sin \omega y}{\Delta h_f} \end{aligned} \tag{4.30}$$

This can be thought to be split into 2 part, perturbed and unperturbed part such that

$$V = \frac{\kappa_s G_s - \kappa_l G_l}{\Delta h_f} \tag{4.31}$$

and,

$$\dot{\delta} \sin \omega y = \frac{n_2 V \delta \sin \omega y + (\kappa_l b_l B - \kappa_s b_s R) \delta \sin \omega y}{\Delta h_f} \tag{4.32}$$

which gives

$$\frac{\dot{\delta}}{\delta} = \frac{n_2 V + \kappa_l b_l B - \kappa_s b_s R}{\Delta h_f} \tag{4.33}$$

The quantity $\frac{\dot{\delta}}{\delta}$ determines the stability of the interface. Any positive value of $\frac{\dot{\delta}}{\delta}$ would mean that the disturbances are amplified

Substituting for B and R we get,

$$\frac{\dot{\delta}}{\delta} = \frac{n_2 V - \kappa_s b_s \left(G_l - G_s\right) + \left(\kappa_l b_l - \kappa_s b_s\right) \left(\phi - \omega^2 \Gamma + mA\right)}{\Delta h_f} \tag{4.34}$$

At marginal stability i.e. at $\frac{\dot{\delta}}{\delta} = 0$ we can write equation 4.34 as

$$-\Gamma\omega^2 - [\bar{\kappa}_l G_l \xi_l + \bar{\kappa}_s G_s \xi_s] + m G_c \xi_c = 0 \tag{4.35}$$

where

$$\begin{aligned}
\xi_l &= [b_l - (V/a_l)] \,/\, [\bar{\kappa}_l b_l + \bar{\kappa}_s b_s] \\
\xi_s &= [b_s - (V/a_s)] \,/\, [\bar{\kappa}_l b_l + \bar{\kappa}_s b_s] \\
\xi_c &= \left[b_c - \tfrac{V}{D}\right] \,/\, \left[b_c - \tfrac{Vp}{D}\right] \\
\bar{\kappa}_l &= \kappa_l / \left(\kappa_l + \kappa_s\right) \\
\bar{\kappa}_s &= \kappa_s / \left(\kappa_l + \kappa_s\right)
\end{aligned} \tag{4.36}$$

For a situation of no thermal gradients we have,

$$-\Gamma\omega^2 + m G_c \xi_c = 0 \tag{4.37}$$

Using $b_c \sim V/D + (D/V)\omega^2$ we get

$$\xi_c = \left[1 - \left(1 + 4D^2\omega^2/V^2\right)^{1/2}\right] \,/\, \left[1 - 2k - \left(1 + 4D^2\omega^2/V^2\right)^{1/2}\right] \tag{4.38}$$

And by inserting Peclet numbers we get,

$$\xi_c = 1 - 2k(V) / \left\{\left[1 + (2\pi/P_c(V))^2\right]^{1/2} - 1 + 2k(V)\right\} \tag{4.39}$$

A simple way to see the effectiveness take $\lim_{V\to\infty} \xi_c$,

i.e. at rapid solidification rates or when the partition coefficient is 1.

For disturbances to be amplified,

$$-\Gamma\omega^2 + m G_c \xi_c > 0 \tag{4.40}$$

Since,

$$\lim_{V\to\infty} \xi_c = 0 \tag{4.41}$$

and, $\Gamma\omega^2 > 0$, since Γ cannot be negative, the interface is unconditionally stable. The interesting part here is to understand the behaviour of $\xi_c(V)$, and for the value of what velocity Equation 4.37 holds, and to find that critical velocity. It is also important to mention that in presence of thermal gradients the critical velocity is similar. It is just a interplay of more factors for stability.

$$F(V) = -\Gamma\omega^2 + mG_c\xi_c \tag{4.42}$$

Plotting F(V) with ballpark values gives,

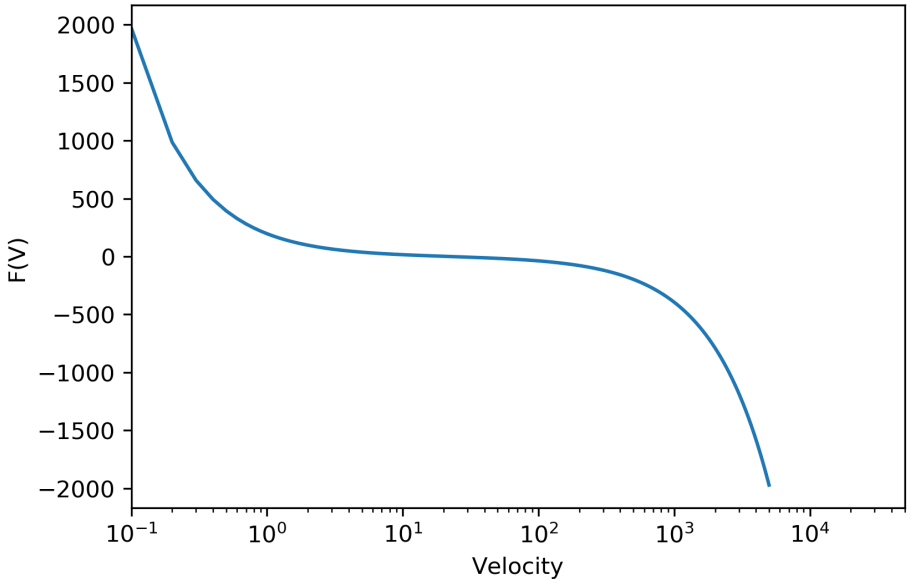

FIGURE 4.1: Variation of F(V) with velocity

As can be seen there is a transition zone which matches with the typical rapid solidification velocities

Galenko and Sobolev in a series of studies introduced a new partition coefficient for high velocity solidification using hyperbolic diffusion.

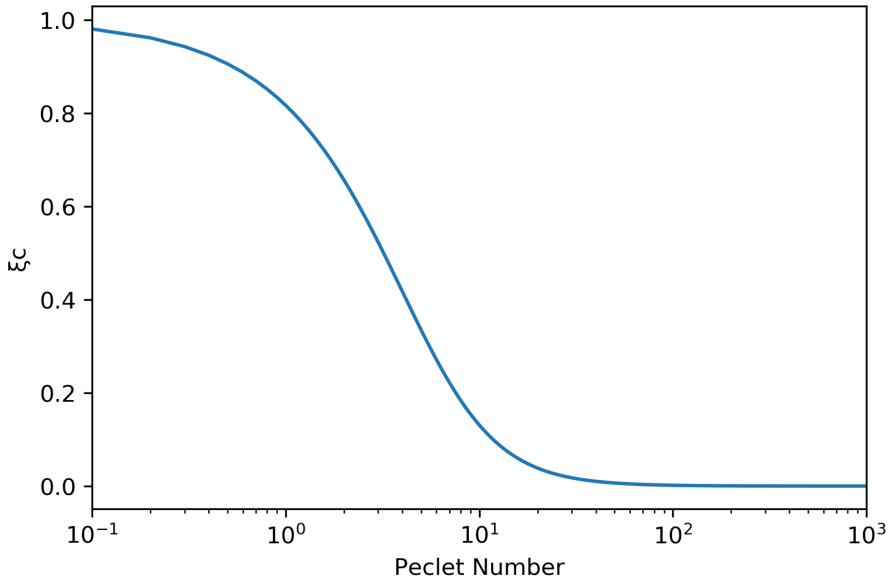

FIGURE 4.2: $\xi_c(Pe)$ with peclet number

$$\begin{gathered} k(V) = \frac{k_e\left(1-V^2/V_D^2\right)+V/V_{DI}}{1-V^2/V_D^2+V/V_{DI}}, \quad V < V_D \\ k(V) = 1, \quad V \geqslant V_D \end{gathered} \tag{4.43}$$

This hyperbolic diffusion can be used to modify the $\xi_c(V)$

$$\begin{gathered} \bar{\xi_c} = 1 - 2k(V)/\left\{\left[1 + (2\pi/P_c(V))^2 \times (1 - \tfrac{V^2}{V_D^2})\right]^{1/2} - 1 + 2k(V)\right\}, \quad V < V_D \\ \bar{\xi_c} = 0, \quad V \geqslant V_D \end{gathered} \tag{4.44}$$

The main difference of $\bar{\xi_c}$ is that the value of it goes to 0 at a finite velocity, as opposed to ξ_c, for which it happens at when $\lim V \to \infty$

4.3 Discussion

As shown in figure 4.1, there is a transition from instable to stable interface at a finite velocity which is dependent on ξ_c. However, ξ_c is derived using the classical diffuion. Galenko and co-workers [2] have proposed the use of hyperbolic diffusion to incorporate the correct behaviour of solute trapping. This form has been used

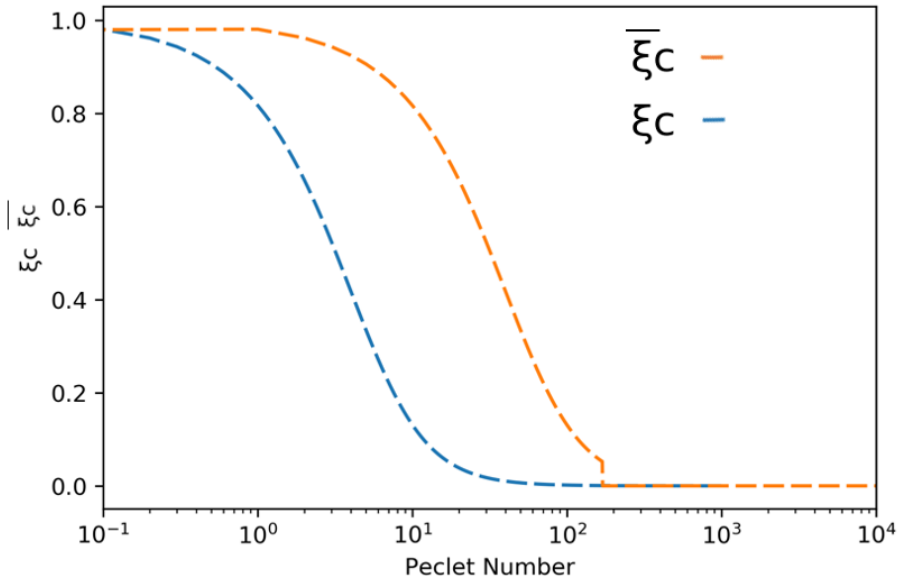

FIGURE 4.3: Comparison with $\bar{\xi}_c$ and ξ_c

to modify the ξ_c to $\bar{\xi}_c$ and the comparison has been presented. This theoritical analysis gives an insight into the fact that interface instability causing dendritic microstructures takes place at only upto certain driving forces. This analysis can also be used to analytically compute the R_{Tip} using the marginal stability and Ivantsov solution $\bar{\xi}_c$.

Chapter 5

Conclusions, Planned work and Future Scope of work

5.1 Conclusions

In Chapter 2, we studied transition from diffusion controlled growth in the grand potential phase field model, and showed how the transition happens at higher driving forces.

In Chapter 3, we developed and benchmarked the Adaptive Mesh refinement(AMR) based phase field solver developed in OpenFOAM. We showed that the AMR based solver is much faster than traditional static mesh solvers.

In Chapter 4, we derived the instability analysis for higher driving forces both for classical and hyperbolic diffusion, and compared and contrasted the two. The velocity ranges for which the interface is unstable could be determined using this analysis.

5.2 Planned work

The pandemic unprecedentedly halted the flow of work. The 6 weeks of lost work was supposed to be the composed of determining the tip selection of free dendrite using the developed solvers. It was further planned to run simulations in the presence of a thermal gradients.

5.3 Problems faced

The starting problem faced was the learning curve associated with OpenFOAM. Familiarizing with OpenFOAM takes sometime, and developing solvers takes a fair amount of expertise with the huge toolbox. There were also problems associated with version mismatch and installing OpenFOAM in HPCs.

Working remotely slowed everything down. The data files ranged in tens of GBs. Transferring them remotely, and processing took time. This was amplified was unavailability of good computing power at home.

5.4 Future Scope of work

In addition to performing the planned work, the future scope of work would be venturing into simulation of other microstructures like eutectic microstructures. Another direction of work would be to study rapid solification in multi-component alloys, and not be restricted to binary alloys.

Appendix A

AMR Phase-Field Solver

A.1 main.c

```
/*---------------------------------------------------------------------------*\
  =========                 |
  \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox
   \\    /   O peration     |
    \\  /    A nd           | Copyright (C) 2011-2018 OpenFOAM Foundation
     \\/     M anipulation  |
-------------------------------------------------------------------------------
License
    This file is part of OpenFOAM.

    OpenFOAM is free software: you can redistribute it and/or modify it
    under the terms of the GNU General Public License as published by
    the Free Software Foundation, either version 3 of the License, or
    (at your option) any later version.

    OpenFOAM is distributed in the hope that it will be useful, but WITHOUT
    ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or
    FITNESS FOR A PARTICULAR PURPOSE.  See the GNU General Public License
    for more details.

    You should have received a copy of the GNU General Public License
    along with OpenFOAM.  If not, see <http://www.gnu.org/licenses/>.

Application
```

```
    laplacianFoam

Description
    Solves a simple Laplace equation, e.g. for thermal diffusion in a solid.

\*---------------------------------------------------------------------------*/

#include "fvCFD.H"
//#include "fvOptions.H"
#include "simpleControl.H"
#include "Random.H"
#include "dynamicFvMesh.H"
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

int main(int argc, char *argv[])
{
    #include "setRootCase.H"

    #include "createTime.H"
    #include "createDynamicFvMesh.H"

    simpleControl simple(mesh);

    #include "createFields.H"

    // * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

    Info<< "\nCalculating temperature distribution\n" << endl;

   volVectorField q=dimx*fvc::grad(phi);
   volScalarField magq = 0.0*phi;
   volScalarField T = 0.0*phi + initial;
   volScalarField sumLHS = 0.0*phi;
   volVectorField q_4 = 0.0*q;
   volVectorField q_6 = 0.0*q;
   volScalarField ac_01 =0.0*phi;
   volVectorField dAdq01= phi*vector(0,0,0);
   volVectorField dadgradPhi=q*0.0;

    while (simple.loop(runTime))
    {
        Info<< "Time = " << runTime.timeName() << nl << endl;

          //  T = initial*(1/dimx)*mesh.C();
            T = G*( (1/dimx)* mesh.C().component(vector::X) - v*runTime.value()) + initial;
```

```
          while (simple.correctNonOrthogonal())
        {


          label curTimeIndex = mesh.time().timeIndex();

        for (int i=0; i<30; i++)
        {
                if(curTimeIndex == 0)
                                {
                                    #include "preAlan.H"
                                }

        }

                mesh.update();
                #include "alphaEqn.H"



        }



    if (runTime.writeTime())
    {
      runTime.write();
    }
        Info<< "ExecutionTime = " << runTime.elapsedCpuTime() << " s"
            << "  ClockTime = " << runTime.elapsedClockTime() << " s"
            << nl << endl;
        runTime.write();
    }

    Info<< "End\n" << endl;

    return 0;
}
```

A.2 alphaeqn.H

```
Random obj(1);
const scalar randNumber(obj.scalar01());
#include "dAdgradPhi_mod.H"
fvScalarMatrix alphaEqn
(


omega*epsilon*dimt*fvm::ddt(phi)     //alpha=1 liquid phase
                ==
                2.0*epsilon*gamma*dimx*dimx*fvm::laplacian(ac_01*ac_01,phi) +

                2*gamma*epsilon*dimx*fvc::div(dadgradPhi) -

                18.0*(gamma/epsilon)*(phi)*(1-phi)*(1-2.0*phi) + B*(c_Sol-c_Liq)*(mu
                - (2*A*c_eq - 2*A*(T0-T)/m_1))*30.0*phi*phi*(1.0-phi)*(1.0-phi) +

                6*noise_mag*phi*(1.0-phi)*phi*(1-phi)*randNumber


);

alphaEqn.solve();

volVectorField n=dimx*fvc::grad(phi)/(1E-20+mag(dimx*fvc::grad(phi)));
fvScalarMatrix muEqn
(
    (0.5)*(1/A)*dimt*fvm::ddt(mu) ==

    diff_Liq*0.5*(1/A)*dimx*dimx*fvm::laplacian((1-phi),mu) - (c_Sol-c_Liq)*
    dimt*fvc::ddt(phi)*30.0*phi*phi*(1.0-phi)*(1.0-phi) -

    anti_trap*epsilon*(c_Sol-c_Liq)*dimx*fvc::div((n*dimt*fvc::ddt(phi)))
);

muEqn.solve();



    Info<< "Min/max phi:" << min(phi).value() << ' '
    << max(phi).value() << endl;
```

Appendix B

Code to find Analytical tip radius

B.1 Matlab code

```
m=0.45;
global c0;
c0=0.5; k=0.78125/0.5 Tm=1; k1=0.45/0.56; global Delta T
for Delta T=0.065 term=@lgk; pe0=0.003;
y=fsolve(term,pe0);
Iv=sqrt(pi*y)*exp(y)*(1-erf(sqrt(y))); R=k1*((-m*c0*Iv*(k-1))/(1-Iv*(1-k))+Delta T)  (-1)
end
function y=lgk(pe)
m=0.45;
global c0;
k=0.78125/0.5;
global Delta T; y=-m*c0*(1-k)*(pe/(2*pi*pi)+sqrt(pi*pe)*exp(pe)*(1-erf(sqrt(pe))))
/(Delta T*(1-(1-k)*sqrt(pi*pe)*exp(pe)*(1-erf(sqrt(pe)))))-1 end
```

Appendix C

Script to find interface velocity

C.1 bash code

```
#!/bin/bash
#get-velocity
filename='alpha'
starttime=10
endtime=150
deltaT=1
startline=22
val=0
for (( i=$1; i<=$2; i=i + $3 ))

do

#reconstructPar -time $i > reconstruct.log

#for i in 10 20 30


cd processor0/0.$i
            n=0
            while read line;
            do
  #echo $line
            n=$((n+1))
            if [ $n -ge $startline ];
```

```
                        then
                        num=$line
                        diff=‘echo "scale=14;0.5-$num" | bc‘
                        #echo $diff
                        if [ 1 -eq "$(echo "${diff} > ${val}" | bc)" ]
                                then
                                diff1=‘echo "scale=14;$lastnum-$num" | bc‘
                                diff2=‘echo "scale=14;$lastnum-0.5" | bc‘
                                pos=‘echo "scale=7;$n+($diff2/$diff1)-22" | bc‘
                                break
                                else
                                lastnum=$num
                        fi
            fi
done < $filename
#echo $i $pos $diff1 $diff2 $diff $num $lastnum $n
echo -e 0.$i ’ \t ’ $pos
cd ../..
#rm -r $i
```

Bibliography

[1] M.J. Aziz. *An atomistic model of solute trapping*, pages 113–117. National Bureau of Standards, Gaithersburg, MD, 1982.

[2] P. K. Galenko, E. V. Abramova, D. Jou, D. A. Danilov, V. G. Lebedev, and D. M. Herlach. Solute trapping in rapid solidification of a binary dilute system: A phase-field study. *Physical Review E*, 84(4), October 2011.

[3] Tomohiro Takaki. Phase-field modeling and simulations of dendrite growth. *ISIJ International*, 54(2):437–444, 2014.

[4] R. Trivedi and W. Kurz. Dendritic growth. *International Materials Reviews*, 39(2):49–74, 1994.

[5] A.A. Wheeler, B.T. Murray, and R.J. Schaefer. Computation of dendrites using a phase field model. *Physica D: Nonlinear Phenomena*, 66(1):243 – 262, 1993.

[6] R. Willnecker, D. M. Herlach, and B. Feuerbacher. Evidence of nonequilibrium processes in rapid solidification of undercooled metals. *Phys. Rev. Lett.*, 62:2707–2710, Jun 1989.

[7] R. Trivedi and W. Kurz. *Theory of Microstructural Development During Rapid Solidification*, pages 260–267. Springer Netherlands, Dordrecht, 1986.

[8] E. J. Lavernia, J. D. Ayers, and T. S. Srivatsan. Rapid solidification processing with specific application to aluminium alloys. *International Materials Reviews*, 37(1):1–44, 1992.

[9] S. J. Savage and F. H. Froes. Production of rapidly solidified metals and alloys. *JOM*, 36(4):20–33, April 1984.

[10] Long-Qing Chen. Phase-field models for microstructure evolution. *Annual Review of Materials Research*, 32(1):113–140, August 2002.

[11] W. J. Boettinger, J. A. Warren, C. Beckermann, and A. Karma. Phase-field simulation of solidification. *Annual Review of Materials Research*, 32(1):163–194, August 2002.

[12] Ingo Steinbach. Phase-field models in materials science. *Modelling and Simulation in Materials Science and Engineering*, 17(7):073001, jul 2009.

[13] Alfonso Fuggetta. Open source software—an evaluation. *Journal of Systems and Software*, 66(1):77–90, April 2003.

[14] W. W. Mullins and R. F. Sekerka. Morphological stability of a particle growing by diffusion or heat flow. *Journal of Applied Physics*, 34(2):323–329, February 1963.

[15] J. Langer. Les houches session xlvi, edited by j. souletie, j. vannenimus, and r. stora northholland, amsterdam, 1987!, pp. 629–711. *Chance and Matter*.

[16] David A. Kessler, Joel Koplik, and Herbert Levine. Pattern selection in fingered growth phenomena. *Advances in Physics*, 37(3):255–339, June 1988.

[17] David A. Kessler, Joel Koplik, and Herbert Levine. Geometrical models of interface evolution. II. numerical simulation. *Physical Review A*, 30(6):3161–3174, December 1984.

[18] E. Ben-Jacob, Nigel Goldenfeld, B. G. Kotliar, and J. S. Langer. Pattern selection in dendritic solidification. *Physical Review Letters*, 53(22):2110–2113, November 1984.

[19] Alain Karma and Wouter-Jan Rappel. Phase-field method for computationally efficient modeling of solidification with arbitrary interface kinetics. *Phys. Rev. E*, 53:R3017–R3020, Apr 1996.

[20] Nikolas Provatas, Nigel Goldenfeld, and Jonathan Dantzig. Efficient computation of dendritic microstructures using adaptive mesh refinement. *Phys. Rev. Lett.*, 80:3308–3311, Apr 1998.

[21] Yung-Tae Kim, Nigel Goldenfeld, and Jonathan Dantzig. Computation of dendritic microstructures using a level set method. *Phys. Rev. E*, 62:2471–2474, Aug 2000.

[22] Sebastian Gurevich, Alain Karma, Mathis Plapp, and Rohit Trivedi. Phase-field study of three-dimensional steady-state growth shapes in directional solidification. *Phys. Rev. E*, 81:011603, Jan 2010.

[23] G. Ivantsov. Ptemperature field around a spherical, cylindrical, and needle-shaped crystal, growing in a pre-cooled melt, temperature field around a spherical, cylindrical, and needle-shaped crystal, growing in a pre-cooled melt transl. into english of temperaturnuye pole vokrug sharoobraznogo tsilindricheskogo i igloobrazno go kristalla rastushchego v pereokhlazhdennom rasplave"doklady, akademiya nauk ssr, vol. 58, 1947, pp. 567–569.

[24] S.-C. Huang and M.E. Glicksman. Overview 12: Fundamentals of dendritic solidification—i. steady-state tip growth. *Acta Metallurgica*, 29(5):701–715, May 1981.

[25] M.E. Glicksman. Free dendritic growth. *Materials Science and Engineering*, 65(1):45–55, July 1984.

[26] G Horvay and J.W Cahn. Dendritic and spheroidal growth. *Acta Metallurgica*, 9(7):695–705, July 1961.

[27] William Oldfield. Computer model studies of dendritic growth. *Materials Science and Engineering*, 11(4):211–218, April 1973.

[28] J.S Langer and H Müller-Krumbhaar. Theory of dendritic growth—i. elements of a stability analysis. *Acta Metallurgica*, 26(11):1681–1687, November 1978.

[29] A. Barbieri and J. S. Langer. Predictions of dendritic growth rates in the linearized solvability theory. *Phys. Rev. A*, 39:5314–5325, May 1989.

[30] LANGER JS. Dendritic solidification of dilute solutions, 1980.

[31] Alain Karma and J. S. Langer. Impurity effects in dendritic solidification. *Phys. Rev. A*, 30:3147–3155, Dec 1984.

[32] J. Lipton, M.E. Glicksman, and W. Kurz. Dendritic growth into undercooled alloy metals. *Materials Science and Engineering*, 65(1):57–63, July 1984.

[33] J. Lipton, M. E. Glicksman, and W. Kurz. Equiaxed dendrite growth in alloys at small supercooling. *Metallurgical Transactions A*, 18(2):341–345, February 1987.

[34] W. Kurz, B. Giovanola, and R. Trivedi. Theory of microstructural development during rapid solidification. *Acta Metallurgica*, 34(5):823–830, May 1986.

[35] M. Bobadilla, J. Lacaze, and G. Lesoult. Influence des conditions de solidification sur le déroulement de la solidification des aciers inoxydables austénitiques. *Journal of Crystal Growth*, 89(4):531–544, July 1988.

[36] M. Rappaz, S. A. David, J. M. Vitek, and L. A. Boatner. Analysis of solidification microstructures in fe-ni-cr single-crystal welds. *Metallurgical Transactions A*, 21(6):1767–1782, June 1990.

[37] Ryo Kobayashi. Modeling and numerical simulations of dendritic crystal growth. *Physica D: Nonlinear Phenomena*, 63(3-4):410–423, March 1993.

[38] J.A. Warren and W.J. Boettinger. Prediction of dendritic growth and microsegregation patterns in a binary alloy using the phase-field method. *Acta Metallurgica et Materialia*, 43(2):689–703, February 1995.

[39] Alain Karma and Wouter-Jan Rappel. Quantitative phase-field modeling of dendritic growth in two and three dimensions. *Phys. Rev. E*, 57:4323–4349, Apr 1998.

[40] W. Losert, D. A. Stillman, H. Z. Cummins, P. Kopczyński, W.-J. Rappel, and A. Karma. Selection of doublet cellular patterns in directional solidification through spatially periodic perturbations. *Phys. Rev. E*, 58:7492–7506, Dec 1998.

[41] Yung-Tae Kim, Nikolas Provatas, Nigel Goldenfeld, and Jonathan Dantzig. Universal dynamics of phase-field models for dendritic growth. *Phys. Rev. E*, 59:R2546–R2549, Mar 1999.

[42] Abhik Choudhury and Britta Nestler. Grand-potential formulation for multicomponent phase transformations combined with thin-interface asymptotics of the double-obstacle potential. *Phys. Rev. E*, 85:021602, Feb 2012.

[43] Bikramjit Karmakar. On the phase field modelling of rapid solidification, bachelor's thesis, 2019.

[44] William J. Boettinger and James A. Warren. The phase-field method: simulation of alloy dendritic solidification during recalescence. *Metallurgical and Materials Transactions A*, 27(3):657–669, March 1996.

[45] B. Wei and D.M. Herlach. Dendrite growth during rapid solidification of co-sb alloys. *Materials Science and Engineering: A*, 226-228:799–803, June 1997.

[46] H.K.D.H. Bhadeshia. Some difficulties in the theory of diffusion-controlled growth in substitutionally alloyed steels. *Current Opinion in Solid State and Materials Science*, 20(6):396–400, December 2016.

[47] N. A. Ahmad, A. A. Wheeler, W. J. Boettinger, and G. B. McFadden. Solute trapping and solute drag in a phase-field model of rapid solidification. *Phys. Rev. E*, 58:3436–3450, Sep 1998.

[48] Alain Karma. Phase-field formulation for quantitative modeling of alloy solidification. *Phys. Rev. Lett.*, 87:115701, Aug 2001.

[49] Openfoam: Api guide: dynamicfvmesh class reference. `https://www.openfoam.com/documentation/guides/latest/api/classFoam_1_1dynamicFvMesh.html`. (Accessed on 06/22/2020).

[50] Openfoam: Api guide: topochangerfvmesh class reference. `https://www.openfoam.com/documentation/guides/latest/api/classFoam_1_1topoChangerFvMesh.html`. (Accessed on 06/22/2020).

[51] shor-ty / dynamicinterfacerefinefvmesh — bitbucket. `https://bitbucket.org/shor-ty/dynamicinterfacerefinefvmesh/src/master/`. (Accessed on 06/22/2020).

[52] Ravi Kumar Singh. Openfoam for solving material science problems, 2019.

[53] Peter Galenko and Sergei Sobolev. Local nonequilibrium effect on undercooling in rapid solidification of alloys. *Phys. Rev. E*, 55:343–352, Jan 1997.