MULTISCALE MODELING OF SOLIDIFICATION OF MULTI-COMPONENT ALLOYS

A Dissertation

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

by

Lijian Tan

August 2007

MULTISCALE MODELING OF SOLIDIFICATION OF MULTI-COMPONENT ALLOYS

Lijian Tan, Ph.D.

Cornell University 2007

Modeling solidification in the micro-scale is computationally intensive. To overcome this difficulty, a method combining features of front-tracking methods and fixed-domain methods is developed. To explicitly track the interface growth and shape of the solidifying crystals, a front-tracking approach based on the level set method is implemented. To easily model the heat and momentum transport, a fixed-domain method is implemented assuming a diffused freezing front where the liquid fraction is defined in terms of the level set function. The fixed-domain approach, by avoiding the explicit application of essential boundary conditions on the freezing front, leads to an energy conserving methodology that is not sensitive to the mesh size. Techniques including fast marching, narrow band computing and adaptive meshing are utilized to speed up computations. The model is used to investigate various phenomena in solidification including two- and three-dimensional dendrite growth of pure material and alloys, eutectic and peritectic solidification, convection effects on crystal and dendrite growth, planar/cellular/dendritic transition, interaction between multiple dendrites, columnar/equiaxed transition and etc.

Interaction between thousands or even millions of crystals gives the overall behavior of the solidification process and defines the properties of the final product. A multiscale model based on a database approach is developed to investigate alloy solidification. Appropriate assumptions are introduced to describe the behavior of macroscopic temperature, macroscopic concentration, liquid volume fraction and microstructure features. These assumptions lead to a macroscale model with two unknown functions: liquid volume fraction and microstructure features. These functions are computed using information from microscale solutions of selected problems. A computationally efficient model, which is different from the microscale and macroscale models, is utilized to find relevant sample problems. The microscale solution of the relevant sample problems is then utilized to evaluate the two unknown functions (liquid volume fraction and microstructure features) in the macroscale model. The temperature solution of the macroscale model is further used to improve the estimation of the liquid volume fraction and microstructure features. Interpolation is utilized in the feature space to greatly reduce the number of required sample problems. The efficiency of the proposed multiscale framework is demonstrated with numerical examples that consider a large number of crystals. A computationally intensive fully-resolved microscale analysis is also performed to evaluate the accuracy of the multiscale framework.

BIOGRAPHICAL SKETCH

The author was born in the city of Xiangxiang, Hunan province, China, in October 1978. After completing his high school education from the YiZhong high school in the city of ShuangFeng, the author was admitted into the Bachelor's program in Mechanical Engineering at Tsinghua University in 1996, from where he received his Bachelor's degree in 2000, and Master's degree in 2003. In May 2003, the author entered the doctoral program at the Sibley School of Mechanical and Aerospace Engineering, Cornell University.

This thesis is dedicated to my wife Wenyan, my parents Zhiqiang and Sujun, and my sister Liyong for their constant support and encouragement.

ACKNOWLEDGEMENTS

I would like to thank my advisor, Professor Nicholas Zabaras, for his constant support, motivation and guidance over the last 4 years. I would also like to thank Professors Subrata Mukherjee, Stephen Vavasis and Doug James for serving on my special committee and for their encouragement and suggestions during the course of this work.

The financial support for this project was provided by the Department of Energy (grant DE-FC07-ID0214396). I would like to thank the Sibley School of Mechanical and Aerospace Engineering for having supported me through a teaching assistantship for part of my study at Cornell. The computing for this project was performed on the windows cluster in CTC (Cornell Theory Center), and the MPDC (Materials Process Design and Control Laboratory) linux cluster.

Most of the computer codes associated with this project were developed based on two open source scientific computation packages `PETSc and ParMetis`. I would like to acknowledge the effort of their developers. I also acknowledge the support I received from Deep Samanta and Baskar Ganapathysubramanian, with whom I closely worked during the course of this project. Finally, I would like to thank all members in the group, especially Jingbo Wang, for their help in the last four years.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

1.1 Microscale models for alloy solidification

Simulation of solidification and crystal growth processes has been of interest for a number of years not only because of its scientific value in understanding pattern formation in nature but also because of its importance in many technological applications. The classical Stefan problem with a sharp interface is well accepted for modelling the solidification of pure materials and alloys. Front-tracking techniques have been widely used for such sharp front solidification processes. Recently these techniques have been used successfully to reproduce the complex dendritic structure in crystal growth in undercooled melts including effects such as liquid trapping, tip-splitting, side branching and coarsening [29, 2, 74, 75, 3] Successful three-dimensional front-tracking implementations including the effects of melt flow have been presented [3]. The advantages of front-tracking methodologies lie in their ability to directly enforce the freezing interface temperature (Gibbs-Thomson) relation and energy balance (Stefan condition). Unfortunately, many of the current implementations of these conditions do not allow global energy conservation even though they may satisfy the Stefan condition pointwise. Such formulations appear to result in high sensitivity to mesh size and orientation. Moreover, the complexity for handling interfaces in all possible solidification conditions such as advancing/receding, merging/splitting and interface unit normal vector/curvature computations limit the applicability of front-tracking methods to single solid phase systems.

During the last two decades, significant progress has been made in the simulation of microstructure evolution in solidification processes using phase-field methods [23, 8, 48, 30]. These approaches, by considering a diffused-interface and a fixed-grid, avoid the need for applying temperature boundary conditions on the moving interface. A review of recent progress in phase field methods as applied to solidification processes is given in [8]. The basic idea of the phase-field method is to employ a phase-field variable Φ that varies smoothly from zero to unity between the two phases over the diffused-interface region, which has a small but numerically resolvable thickness. The phase field variable serves to distribute the interfacial forces over the diffused freezing region. It is governed by a phase-field control equation derived from the thermodynamics of phase transition [8]. Important physical mechanisms, such as curvature, anisotropy and kinetics effects, are implicitly incorporated in the phase-field control equation. This leads to many computational advantages. For example, one does not need to compute interfacial geometric quantities such as interface curvature and outward normal vector. The phase field method can be shown to reduce to the standard sharp interface formulation in the limit of vanishing interface thickness [40]. The quality of the solution deteriorates with increasing interface thickness. This necessitates the grid spacing to be of the order of or smaller than the interfacial thickness. It has been shown that the interface thickness must be smaller than the capillary length for the solution to converge to the sharp interface limit [71, 67]. Improved asymptotic coefficients have been derived for the thin-interface limit of the phase field equations which facilitate the use of a coarser grid [31]. Phase-field analysis for unequal solid/liquid diffusivities leads to computational models which require finer grid resolution and hence lead to slower computational performance [1]. One of the drawbacks of the phase field method is the significant computational effort required, especially when investigating dendritic growth in the presence of convection and multiple array dendritic growth. However, the techniques of adaptive gridding and parallel computing have resulted in assuaging this drawback to some extent [70, 49]. Another drawback of the phase field methods is related to the large number of parameters involved in the solution of the evolution equations. These parameters are difficult to determine for accurate physical crystal growth simulation of real world materials.

The level set method is an alternative method to handle the sharp interface front directly and to avoid the asymptotic analysis needed in phase field models [56, 43]. It has been shown to be a promising mathematical tool for tracking the interface with low computational cost. It is widely used in various applications such as two-phase flow, crack propagation, computer vision and image processing. In this method, interfacial geometric quantities such as curvature and outward normal can be easily calculated using the level set variable ϕ. The method was first applied to Stefan problems in [12]. Level set calculations for dendritic growth were reported in [32]. In these works [12, 32], the heat flux at nodes near the interface is interpolated in order to calculate the interface velocity via the Stefan condition. During this interpolation, the temperature on the interface is fixed at the equilibrium temperature defined from the Gibbs-Thomson relation. However, like front-tracking methods, the direct application of temperature boundary conditions on the interface and the computation of heat fluxes from the temperature nodal values usually lead to energy conservation issues associated with the discretization error [26]. This may result in large variation of the computed crystal shapes if meshes of different sizes and orientations are used.

Interaction between thousands or even millions of crystals gives the overall behavior of the solidification process and defines the properties of the final product. The cellular automata technique [20, 33, 50, 68] is suitable for studying the interaction between multiple dendrites especially in three-dimensions (3D), because its computational requirement (memory and time) is low in comparison to other numerical methods. Cellular automata is widely used in many areas including biological systems and highway traffic modeling. In this technique, a collection of 'colored' cells on a grid of specified shape evolves through a number of discrete time steps according to a set of rules based on the states of neighboring cells. Cellular automata was first introduced into solidification systems in [51] with a growth kinetics model, which can reflect the growth along preferred directions (e.g. $< 100 >$ crystallographic orientations) during dendrite development. The basic idea of cellular automata for solidification is to mimic interface propagation by capturing nearby liquid cells to the solid body according to a certain criterion. In recent advances of this method, cellular automata is coupled with finite differences (CA-FD) [33, 68] or the finite element (CA-FE) [20] by using a criterion based on numerical solution of temperature or solute concentration from finite differences or the finite element method. Curvature effects can also be taken into account [33]. Cellular automata for solidification is relatively easier to implement and requires less computational resources than most other numerical methods, including front-tracking, phase field method, and level set method. However, it has some deficiencies in its accuracy due to its discrete nature. For example, it is pointed out in [20] that cellular automata methods have a tendency to bias the results by introducing an anisotropy associated with the network of cells or sites. Although corrections can be introduced [20] to circumvent this problem, independence of the cellular automata results on the numerical grid size and mesh orientation is rarely demonstrated. It is also pointed out in [5] that cellular automata lacks the ability to accurately take into account the surface tension anisotropy effect, which is of great importance in dendritic growth.

Although cellular automata is computationally efficient, issues of accuracy are not sufficiently addressed by the cellular automata method. Many researchers are making effort to use the more complicated and more expensive methods (phase field method, level set method and front-tracking method) to study the interaction between multiple crystal for better accuracy. Reference [74] uses a front-tracking method to study the interaction between multiple crystals with the same initial perturbation in the seeds. References [12, 22] demonstrate the ability of using the level set method to capture the growth and merging of multiple dendrites with different initial perturbations. However, in these studies [74, 12, 22], nucleation is not modeled. Only recently, reference [5] gives the first phase field study of interaction between tens of crystals by fully utilizing symmetry in 2D with modeling of nucleation. Interaction between hundreds or even thousands of randomly nucleated crystals with a convergence study has not yet been demonstrated in the literature. In all of the above studies, convergence and mesh orientation independence are not demonstrated. More importantly, the randomness of crystal orientation is not considered in these studies.

The primary reason for the lack of progress in studying crystal interaction with phase field method, level set method or front-tracking method is the huge computation cost. In fact, obtaining a converged solution for a single crystal independent of mesh orientation itself is a nontrivial task due to the existence of a moving interface during the solidification process [76, 3, 8, 32, 22, 61]. This point is particularly brought into context in [48], where it took approximately 10 CPU hours on a Sun UltraSPARC 2200 workstation to simulate the growth of a single crystal at dimensionless undercooling 0.55 by utilizing the improved phase field method [31] and adaptive meshing technique. On the other hand, since level set and other front tracking methods explicitly track the interface, they do not suffer from this particular problem. The major problems plaguing these explicit tracking techniques are energy conservation issues. This arises primarily due to the direct application of Gibbs-Thomson relation on the interface. These set of problems can only be resolved by using a fine mesh. As demonstrated in [29, 12, 32], a typical 2D mesh size to obtain a converged solution using the front-tracking method or level set method is of the size about 400×400. In essence, both phase field and level-set methods are bottle-necked by (different) issues that make realistic multiple dendrite simulation (while demonstrating convergence and mesh independence of the solution) a computationally daunting task requiring huge meshes and consequently enormous computational resources.

The method presented here combines features of front-tracking and fixed domain methods. The level set method is implemented to allow a direct calculation of the growth and shape of the solidifying crystals. An energy conserving implementation of the level set method is used without the need to apply the temperature boundary condition explicitly at the freezing interface. To easily model the heat and momentum transport, a fixed domain method is implemented assuming a diffused freezing front where the liquid fraction is defined in terms of the level set function. The fixed domain approach, by avoiding the need to explicitly apply essential boundary conditions on the freezing front, leads to an energy conserving methodology that is not sensitive to the mesh size or orientation. The present implementation is similar to that of the phase field method, except that the 'phase field variable' Φ is no longer governed by the phase-field control equation but is defined using the level set (signed distance) variable ϕ. For systems with multiple solid phases, phase boundaries are tracked by solving the multi-phase level set equations. The melt flow is modelled using equal-order velocity-pressure interpolation that has been shown to lead to better convergence rates and accuracy [72] than the fractional step method commonly used in dendritic solidification models. The present methodology will be shown to be computationally efficient and accurate for both two- and three-dimensional problems. It will be demonstrated that the methodology performs very well in benchmark dendritic growth problems including growth under low-undercooling conditions for which other numerical approaches do not work [46]. The interaction between multiple dendrites can be studied numerically with demonstration of convergence, which will serve as the foundation for multi-scale modeling of solidification processes.

1.2 Multi-scale models for alloy solidification

Investigating the interaction and growth of crystals in the microscale is computationally very intensive, whereas macroscale models lack accuracy since they are based on a large number of simplifications. Multiscale modeling by coupling macroscopic and microscopic models allows us to take advantage of both the efficiency of the macroscopic models and the accuracy of the microscopic models. There are mainly three types of multiscale techniques for solidification used in the literature: (1) resolving microscale details in an efficient way, such as using an adaptive mesh refinement [49, 60, 41, 63] or an adaptive moving grid [19], since the main complexity in the microscale arises from a moving solid-liquid interface, (2) performing analytical studies [69] or simple numerical computation [16] in the microscale and passing the required information to the macroscale computation, and (3) designing multiscale algorithms driven by microscopic numerical solution data, e.g. regression fit [68] and subgrid modeling [50].

Adaptive mesh refinement is often coupled with a phase field method to simulate the growth of 2D and 3D dendrites. It exploits the fact that the phase field variable, temperature and concentration fields vary significantly only near the solid-liquid interface. Similarly to adaptive mesh refinement, a moving mesh strategy is utilized in [19] to make grid points spatially adaptive in the physical domain, while maintaining a structured grid in the computational domain for the Fourier spectral implementation. Although, adaptive techniques can greatly reduce the degrees of freedom for the problem of interest, their application is often limited to the study of single crystals [49, 19] or of a small number of dendrites/crystals [60] because of the required small grid spacing for convergence to the sharp interface limit when using the phase field method. By combining both features of front tracking and fixed domain methods in [63], this thesis work will demonstrate the ability to simulate interaction between hundreds of randomly nucleated crystals with convergence studies by utilizing adaptive mesh refinement, the level set method and parallel computation as a part of this thesis work. Despite this improvement, all of these adaptive techniques [49, 60, 41, 63] are microscale models in nature. Thus with current computing resources, they are not very practical for investigating interaction between far more than hundreds of crystals.

An alternative multiscale technique performs analytical study in the microscale with certain simplifications to provide microstructural information to the macroscale model. Reference [69] models the solidification system as solid phase, inter-dendritic and extra-dendritic liquid phases. The macroscopic transport equations for these three phases are derived using volume averaging technique and closed by supplementary relations, which are obtained from analysis in the microscale. In [69], there is no numerical computation performed at the microscale. But in some other studies (e.g. [16]), the microscopic problems are not avoided completely. Assuming periodic distribution of crystals, numerical computation of a single crystal growth is carried out for every point of a macroscopic grid to provide information for the macroscale computation. Although numerical computation is reduced, the assumption of periodicity is physically unrealistic. This limits its application to only equiaxed growth. Without the ability to capture important physics in the microscale (e.g. randomness in nucleation and crystal orientation), both of these methods [69, 16] are phenomenological in nature regarding their prediction for example of the microstructure type and size.

Multiscale algorithms driven by data have the ability to study interaction between thousands or even millions of crystals. The microscale model tries to accurately capture important physics. Computation with the microscale model is then used to provide necessary data for macroscale computation. For example, a particular form of predictive equation with unknown parameters is assumed in [68] to describe pore formation.

Microscale computations are used to obtain data for regression fit of the predictive equation, which is further used for macroscale computation. A disadvantage of this method is that the predictive equation has a simple form with only a few unknown parameters. In [50], the idea of subgrid modeling is used for multiscale modeling of solidification. The computation domain is divided into a number of small sub-domains. The microscale model is applied for each sub-domain to pass information of volume fraction to the macroscale model. By dividing the whole domain into small sub-domains, the computational time can be greatly reduced. However, this method is still computationally very intensive. Qualitatively, similar macroscopic conditions would lead to similar microstructure. For example, the microstructure at the corners or at the center of the domain may be similar. In subgrid modeling, this similarity is not taken account into the model to reduce the computational effort. Moreover, results of microscale computation for one problem cannot be applied to another problem. In this thesis work, the similarity between microscale computations is quantitatively explored.

The multiscale method used in this thesis work falls into the heterogeneous multiscale method (HMM) framework [15], which aims at designing combined macroscopic-microscopic computational methods that are much more efficient than solving the full-microscopic model and at the same time lead to a desired level of accuracy. The idea of building a database based on results from microscopic simulation of selected problems and using this database for multiscale computation is very straightforward and has been widely used in many research areas. The database approach has not yet been fully explored for solidification processes. The complex nature of solidification process causes difficulty in finding relevant problems to build the database, and selecting results of relevant problems from the database to solve the problem of interest. The emphasis of multi-scale modeling in this thesis work is on addressing some simple algorithms for handling these difficulties.

The organization of this thesis is as follows: In Chapter 2, the micro-scale model for dendritic solidification of pure materials with combined features of front-tracking and fixed-domain methods is introduced. In Chapter 3, the micro-scale model is extended to multi-component alloys. In Chapter 4, the model is further extended to model the interaction of a large number of crystals. In Chapter 5, a multi-scale framework based on database approach is discussed. Finally, in Chapter 6, conclusions of this thesis work and suggestions for future research are summarized.

Chapter 2

Level set simulation of dendritic solidification for pure materials

2.1 Definition of the Stefan problem

Let us consider the classical Stefan problem with a sharp interface including the effects of melt flow. The liquid phase of a pure material at initial temperature $T_0^\ell(x, y)$ is assumed to occupy a region Ω_0^ℓ. This liquid phase is either alone ($\Omega_0^\ell = \Omega$) or in an unstable equilibrium with the neighboring solid phase Ω_0^s ($\Omega = \Omega_0^\ell \bigcup \Omega_0^s$) at initial temperature $T_0^s(x, y)$. In the latter case, the regions Ω_0^ℓ and Ω_0^s are separated by the solid/liquid interface Γ_0. The superscripts ℓ and s are used here to denote quantities corresponding to the liquid and solid phases, respectively, whereas the subscript 0 is used to denote quantities at time $t = 0$. It is assumed in this thesis work that solidification starts at time $t = 0$. The domains Ω^ℓ and Ω^s are time-dependent and the solid-liquid interface Γ is moving with normal velocity V. The normal $\boldsymbol{n}$ to the interface Γ is defined as pointing away from the solid region Ω^s. The domain $\Omega = \Omega^\ell \bigcup \Omega^s$ containing these two phases and its external boundary $\partial\Omega$ are assumed constant (time-independent). Figure 2.1 presents a typical schematic of the problems considered.

In this chapter, constant thermo-physical and transport properties, including viscosity μ, density ρ, thermal conductivity k, heat capacity c and latent heat L is assumed. The melt flow is assumed to be a laminar flow caused by temperature-induced density variations (Boussinesq flow). Following standard notation, the governing equations in the presence of fluid flow are given as follows:

$$\rho_s c_s \frac{\partial T(\boldsymbol{x}, t)}{\partial t} = k_s \nabla^2 T(\boldsymbol{x}, t), \quad \boldsymbol{x} \in \Omega^s, \tag{2.1}$$

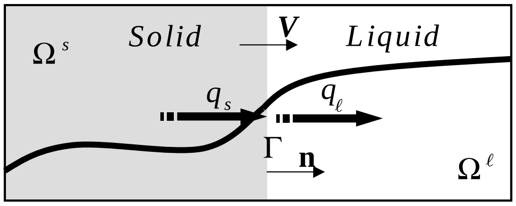


Figure 2.1: Schematic of the sharp interface model of the solidification of a pure material (Stefan problem). The heat fluxes q_s and q_l are defined on the freezing front.

$$\begin{aligned} \rho_l c_l \left(\frac{\partial T(\boldsymbol{x}, t)}{\partial t} + \boldsymbol{v} \cdot \nabla T(\boldsymbol{x}, t) \right) &= k_l \nabla^2 T(\boldsymbol{x}, t), \ \ \boldsymbol{x} \in \Omega^\ell, && (2.2) \\ \rho_l \left(\frac{\partial \boldsymbol{v}}{\partial t} + \nabla \boldsymbol{v}(\boldsymbol{x}, t) \boldsymbol{v}(\boldsymbol{x}, t) \right) &= -\nabla p(\boldsymbol{x}, t) \boldsymbol{I} + \\ \nabla \cdot \mu \left[\nabla \boldsymbol{v}(\boldsymbol{x}, t) + (\nabla \boldsymbol{v}(\boldsymbol{x}, t))^{\mathrm{T}} \right] &+ \boldsymbol{b}, \ \ \boldsymbol{x} \in \Omega^\ell, && (2.3) \\ \nabla \cdot \boldsymbol{v}(\boldsymbol{x}, t) &= 0, \ \ \boldsymbol{x} \in \Omega^\ell, && (2.4) \end{aligned}$$

where $\boldsymbol{v}$ is the melt flow velocity and $\boldsymbol{b}$ is the buoyancy body force.

The temperature on the interface Γ denoted as T_I is equal to the equilibrium temperature T^*. This equilibrium temperature is given from the Gibbs-Thomson relation as follows:

$$T^* = T_m + \varepsilon_c \kappa + \varepsilon_V V, \tag{2.5}$$

where T_m is the pure material melting temperature, κ is the curvature of the freezing interface, and ε_c and ε_V are the curvature and kinetic undercooling coefficients, respectively.

The motion of the interface velocity V is dictated by the classical Stefan equation (energy balance at the freezing front) as follows:

$$\rho_s L V = [q] = q_s - q_l, \tag{2.6}$$

where [q] denotes the jump of the heat flux $q = k \nabla T \cdot \boldsymbol{n}$ at the freezing front.

To describe the interface, we construct a field ϕ such that at any time t, the interface is equal to the zero level set of ϕ, i.e.

$$\Gamma(t) = \{\boldsymbol{x} \in \Omega \ : \ \phi(\boldsymbol{x}, t) = 0\}\,. \tag{2.7}$$

Initially, ϕ is set equal to the signed distance function from the interface Γ_0,

$$\phi(\boldsymbol{x}, 0) = \begin{cases} +d(\boldsymbol{x}, 0), & \boldsymbol{x} \in \Omega_0^\ell, \\ 0, & \boldsymbol{x} \in \Gamma_0, \\ -d(\boldsymbol{x}, 0), & \boldsymbol{x} \in \Omega_0^s, \end{cases} \tag{2.8}$$

where $d(\boldsymbol{x}, 0)$ is the normal distance of a point $\boldsymbol{x}$ from the interface.

The idea behind the level set method is to move ϕ with the correct speed V at the interface which is extracted from Eq. (2.6). The interface position is thus implicitly stored in ϕ. The equation of motion governing ϕ is given as follows:

$$\phi_t + V|\nabla\phi| = 0. \tag{2.9}$$

This equation moves ϕ with correct speed at the interface so that Γ will always be equal to the zero level set of ϕ [56].

With the above introduction of the level set function, we will re-write the Stefan condition for the classical Stefan problem as follows:

$$\rho_s L V = [q], \ \ q_s = \lim_{\phi \to 0^-} q, \ \ \ q_l = \lim_{\phi \to 0^+} q, \tag{2.10}$$

where the notation 0^+ and 0^- is used here for notational simplification to denote the values of ϕ as we approach the freezing front from the solid and liquid sides, respectively.

Equations (2.1)-(2.5) and (2.10) together with appropriate initial temperature and velocity conditions, boundary thermal conditions on Γ and the no-slip condition in all solid boundaries completely define what we here refer to as the *Stefan problem*. Note that in the literature the classical Stefan problem does not include melt flow. These effects have been included herein to allow us to study their importance in crystal growth of pure materials in undercooled melts.

2.2 Introducing the extended Stefan problem

Many of the difficulties in the implementation of the Stefan problem defined earlier are related to the application of the essential boundary condition given in Eq. (2.5). Front-tracking techniques attempt to apply this condition directly to the moving front, often leading to schemes that are not energy conserving. On the other hand, phase field methods consider a diffused-interface model in an attempt to avoid a direct application of this condition. However, phase field models require proper parameter selection through an asymptotic analysis in order to model the sharp interface solidification problem.

To take advantage of the front-tracking capability of the level set methods and of the ability of phase field methods to avoid directly applying Eq. (2.5), we introduce in this section the *extended Stefan problem* that uses features of both methods. Figure 2.2 shows a schematic of the new problem.

Assumption 1: We assume that solidification occurs in a diffused zone of width $2w$ that is symmetric around $\phi = 0$ (see Fig. 2.2). The half-width w is not related with the underlying physics of the solidification of a pure material (which usually happens on a thickness of the order of atomic distances). In this work, the half-width w is selected based on the size of the grid used in the discretization of the problem.

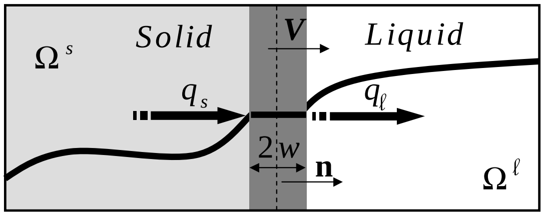


Figure 2.2: Schematic of the diffused-interface model of the solidification of a pure material (extended Stefan problem). The heat fluxes q_s and q_l are here defined at distances $\pm w$ from the actual sharp freezing front.

A diffused-interface model will be used in the solution of the energy and momentum equations. Following ideas similar to those in diffused-interface models, let us define the function $\Phi(\boldsymbol{x}, t)$ as follows:

$$\Phi(\boldsymbol{x}, t) = \begin{cases} 1, & \phi(\boldsymbol{x}, t) > w, \\ 0, & \phi(\boldsymbol{x}, t) < -w, \\ \phi(\boldsymbol{x}, t)/_{2w} + 0.5, & \phi(\boldsymbol{x}, t) \in [-w, w]. \end{cases} \tag{2.11}$$

Since solidification occurs in a diffused-interface, following a volume-averaging approach as in [72], we can write the energy equation (applicable in the whole domain Ω) as follows:

$$\begin{aligned} [\rho_l \Phi c_l + \rho_s (1 - \Phi) c_s] \frac{\partial T}{\partial t} + \rho_l c_l \boldsymbol{v} \cdot \nabla T &= \nabla \cdot (k \nabla T) \\ &- \rho_s [L + (c_s - c_l)(T - T_m)] \dot{\Phi}, \end{aligned} \tag{2.12}$$

where Φ plays the role of the phase field variable in phase field models and $k \equiv k_l \Phi + k_s (1 - \Phi)$. Similarly, the volume-averaging momentum and continuity equations take the following forms:

$$\begin{aligned} \frac{\partial (\rho \boldsymbol{v})}{\partial t} + \nabla \cdot \left(\frac{\rho^2}{\rho_l} \frac{\boldsymbol{v}\boldsymbol{v}}{\Phi}\right) = &- \nabla p + \frac{p}{\Phi} \nabla \Phi + \nabla \cdot \left[\mu\left(\nabla\left(\frac{\rho}{\rho_l} \boldsymbol{v}\right) + \nabla^T\left(\frac{\rho}{\rho_l} \boldsymbol{v}\right)\right)\right] \\ &- \mu \frac{(1 - \Phi)^2}{\Phi^2} \frac{\rho}{\rho_l} \frac{\boldsymbol{v}}{K_0} + \Phi \rho_l g \boldsymbol{e}_{\boldsymbol{g}}, \end{aligned} \tag{2.13}$$

$$\nabla \cdot \boldsymbol{v}(\boldsymbol{x}, t) = 0, \tag{2.14}$$

where $\rho \equiv \rho_l \Phi + \rho_s (1 - \Phi)$, $\boldsymbol{e_g}$ is the unit vector in the direction of gravity, g is the gravity constant and $\boldsymbol{v}$ from now on denotes the volume-averaged velocity equal to $\Phi \boldsymbol{v}_l$ [72]. For simplicity of the model, we assume that the solid/liquid densities and specific heats are the same but we allow for different conductivities.

The Kozeny-Carman approximation for the permeability $K(\Phi)$ has been used directly in Eq. (2.13) with K_0 denoting the permeability constant. Note that the governing conservation Eqs. (2.12), (2.13) and (2.14) are applicable in the whole domain Ω.

The transport equations for the extended Stefan problem are now defined from Eqs. (2.12)-(2.14) together with appropriate initial temperature and velocity conditions, boundary thermal conditions on Γ and the no-slip condition. Note that these equations completely define the temperature and melt flow, but since they are based on a volume-averaging formulation, they do not directly involve the motion of the interface zone. The freezing front motion will be computed by the level set function as it will be discussed below.

To compute the front velocity V (velocity of the interface $\phi = 0$) we cannot simply utilize Eq. (2.10), which is based on the sharp front model. Instead, a modified energy balance is needed at the freezing zone as follows (see Fig. 2.2):

$$2\rho_s \tilde{c} w \frac{\Delta \bar{T}_I}{\Delta t} = \rho_s L V + (q_l - q_s), \quad q_s = \lim_{\phi \to -w^-} q, \quad q_l = \lim_{\phi \to w^+} q, \tag{2.15}$$

where $\bar{T}_I$ is the average temperature within the diffused-interface, and $\Delta \bar{T}_I$ is the increase of $\bar{T}_I$ at the time interval Δt. Also,

$$\tilde{c} \equiv 0.5(1 + \frac{\rho_l c_l}{\rho_s c_s}) c_s, \tag{2.16}$$

and $w^+, -w^-$ are defined following similar notation to that introduced earlier for Eq. (2.10). The term $2\rho_s \tilde{c} w \Delta \bar{T}_I$ on the left hand side of Eq. (2.15) is the energy change in the diffused-interface due to change of temperature. Also note that the temperature on the zero level set denoted as T_I is conceptually different from $\bar{T}_I$, which is the average temperature within the diffused mushy zone. Since the temperature varies only slightly within the thin interface, in this work we will use T_I to approximate $\bar{T}_I$. The temperature T_I can be easily computed from interpolating temperature to zero level set. However, in order to maintain the generality of the formulation, we will maintain the notation $\bar{T}_I$ for the average temperature within the diffused-interface.

The Gibbs-Thomson constraint $T_I = T^*$ with T^* given in Eq. (2.5) ensures that the interface temperature is the equilibrium temperature at each instant. To explicitly enforce this constraint, various methods like a penalty method or a Lagrange multiplier approach can be used. In this work, we take an alternative approach. Since the numerical simulation provides solutions only at discrete time levels, we ensure that the interface temperature approaches the equilibrium temperature at these time levels.

Assumption 2: The mean interface temperature $\bar{T}_I$ in the freezing zone of width $2w$ is allowed to vary from the equilibrium temperature T^* in such a way that

$$\frac{\bar{T}_I}{dt} = -k_N(\bar{T}_I - T^*), \tag{2.17}$$

where k_N controls the rate with which $\bar{T}_I$ is designed to approach the desired equilibrium temperature.

Assuming for example that the equilibrium temperature remains constant, integration of Eq. (2.17) leads to the following:

$$\bar{T}_I(t) - T^* = (\bar{T}_I(0) - T^*) \exp(-k_N t). \tag{2.18}$$

A relaxation time that controls how fast $\bar{T}_I$ approaches T^* can then be defined as $\tau =$

$\frac{1}{k_N}$. The relaxation parameter τ (or equivalently k_N) is selected such that the interface reaches the desired equilibrium temperature exponentially fast. An explicit integration of Eq. (2.17) over the time step Δt results in the following:

$$\bar{T}_I(t_n) - T^* \approx (\bar{T}_I(t_{n-1}) - T^*)(1 - k_N \Delta t), \tag{2.19}$$

where the subscripts refer to time levels ($n = 1, 2, \ldots$). To make sure that the interface temperature stably converges to the equilibrium temperature, we require the following:

$$|1 - k_N \Delta t| < 1 \Rightarrow \Delta t < \frac{2}{k_N} = 2\tau. \tag{2.20}$$

If we further require that $\bar{T}_I - T^*$ does not change sign as it approaches zero ($T^* = \bar{T}_I$), then $\Delta t \leq \tau$. In our algorithms, for a given Δt, we select $k_N = \frac{1}{\Delta t}$ so that the condition $\Delta t \leq \tau$ is satisfied automatically. The selection of Δt will be discussed in Section 2.3.

Substitution of the constraint of Eq. (2.17) to Eq. (2.15) results in the following modified energy balance at the freezing front:

$$V = \frac{[q]}{\rho_s L} + \frac{2\tilde{c}w}{L} k_N (T^* - \bar{T}_I), \; q_s = \lim_{\phi \to -w^-} q, \;\; q_l = \lim_{\phi \to w^+} q, \tag{2.21}$$

where the heat fluxes q_l and q_s are computed at the boundaries of the control volume. Note that in the above formulation, we use the average temperature $\bar{T}_I$ instead of the temperature T_I on the exact interface ($\phi = 0$). Equation (2.21) will be used as the energy balance at the diffused-interface and will allow us to accurately compute the interface velocity V. Note that the above scheme is only to ensure that the interface temperature relaxes to the equilibrium temperature in a given time step.

Remark 1: Note that Eq. (2.17) is not used directly in the analysis but it is embedded in the modified Stefan condition given in Eq. (2.21). To compute a finite front velocity when $k_N \to \infty$, Eq. (2.21) requires that $T_I = T^*$ in which case it is simplified to Eq. (2.10). It can be shown that, using Eq. (2.21), the numerical scheme for the thermal

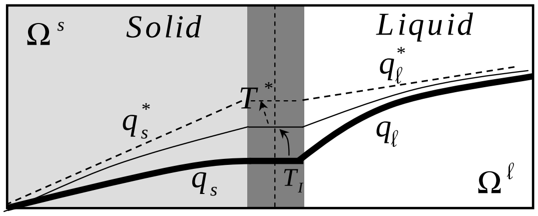


Figure 2.3: Schematic of applying $T_I = T^*$ with the correction of Eq. (2.21).

problem of Eq. (2.12) leads to a discretized form of Eq. (2.17) thus weakly enforcing $T_I = T^*$. The choices $w = \Delta x$ and $k_N = 1/\Delta t$ are sufficient to ensure satisfaction of both Eqs. (2.5) and (2.10) and lead to an accurate estimate of the front velocity V.

Remark 2: From a numerical point of view, the second term on the right hand side of Eq. (2.21) can be thought of as the constraint $T^* - T_I = 0$ numerically enforced via a Lagrange multiplier method. A graphical demonstration of an iterative process for the satisfaction of the constraint $T^* - T_I = 0$ is given in Fig. 2.3. In this figure, we assume that the temperature field away from the freezing zone remains the same during iterations. Figure 2.3 shows the iterative process as T_I approaches T^* from below. This iterative procedure for computing V was not needed in the calculations reported in Section 3.4 and one step calculation was sufficient to evaluate V within the desired accuracy.

Remark 3: The volume-averaging based energy Eq. (2.12) weakly accounts for the Stefan condition on the freezing front by treating its contribution as a source term. This is typically the case with phase field and other diffused-interface models. The key element of the present formulation is that the phase variable Φ (liquid volume fraction) is defined in terms of the level set function ϕ that is used to track the freezing front explicitly. In classical volume-averaging models of solidification, Φ is defined based on thermodynamic update formulas using the computed temperature and equilibrium temperature at each point within the diffused-interface. This last calculation is embedded in the calculation of ϕ.

Remark 4: Curvature or kinetic undercooling effects play a significant role in solidification through the Gibbs-Thomson relation Eq. (2.5). In the traditional phase field method, an asymptotic analysis is required to determine simulation parameters from the values of ε_c and ε_V used to model sharp front solidification. The present level set method solves the extended Stefan problem directly without any need for an asymptotic analysis.

2.3 The level set method

In this work, we consider finite difference approximations for the level set function calculation and volume-averaging based stabilized finite element techniques for modelling the thermal and fluid flow problems to capitalize on recent advances in the implementation of the level set method with finite difference techniques. Only structured grids are considered in this work. For the finite element discretization, four-node bilinear elements (in 2D) and eight-node brick trilinear elements (in 3D) are used.

Let ϕ be a signed distance variable (minimum distance to the interface between the two phases) satisfying $|\nabla\phi| = 1$. Then the normal direction of the interface (pointing from the solid to the liquid) is calculated as follows:

$$\boldsymbol{n} = \frac{\nabla\phi}{|\nabla\phi|}. \tag{2.22}$$

The curvature κ of the interface in terms of ϕ is computed as discussed in [43]. Equation (2.9) is solved for the level set function in a narrow band near the interface. For this extension, the interface velocity is calculated from the extended Stefan condition

Eq. (2.21). Details of numerical schemes for the solution of Eq. (2.9) can be found in [56, 44].

After an update according to the level set Eq. (2.9), ϕ does not in general remain a signed distance function. It is thus necessary for re-initialization where the following equation is iterated until reaching steady-state [12]:

$$\phi_t = \frac{\phi_0}{\sqrt{\phi_0^2 + \varepsilon^2}}(1 - |\nabla\phi|), \tag{2.23}$$

where ϕ_0 is the initial level set value to be re-initialized. After ϕ reaches steady-state, $|\nabla\phi| = 1$, i.e. ϕ is a signed distance. The parameter ε in Eq. (2.23) takes some small value and is needed for the formulation to remain well-posed as $\phi \to 0$. We use $\varepsilon = 2\Delta x$ in our calculations [12]. The time step in this re-initialization process is taken as $\Delta t = \frac{\Delta x}{5}$ and the number of iterations in the re-initialization process is taken to be $200\lambda_{CFL}$, where λ_{CFL} is the CFL condition coefficient ranging from 0 to 1. Note that the solid/liquid interface is advanced within a time step with a distance $\lambda_{CFL}\Delta x$. Thus based on the current re-initialization process, the more distance the interface is advanced forward within a time step, the more iterations will be required to rebuild the signed distance. Unless it is otherwise stated, the CFL coefficient in all the examples of Section 3.4 is selected as $\lambda_{CFL} = 0.3$.

2.4 Energy conserving level set method

The level set method has been successfully applied to Stefan problems in [12, 32, 22]. In this work, we apply the level set method to the extended Stefan problem. Because the Dirichlet temperature boundary condition is not applied directly on the interface, we can use energy conserving numerical schemes for the implementation of the heat transfer problem [26]. While the front-tracking method in [74] and the level set methods in [12, 32, 22] are all analytically energy conserving, when applying Dirichlet temperature boundary condition on the interface, numerically they do not conserve energy. For the method presented in this work (as well as in phase field methods), energy is not only conserved analytically but also numerically. This is the reason we refer to the present methodology as an ‘energy conserving level set method’.

2.4.1 Stability analysis: Selection of the time step and k_N

To simulate sharp-front solidification, one will theoretically be required to consider a very high value for k_N. Such a choice will of course lead to prohibitively small time steps via the stability condition $\Delta t \leq \tau = 1/k_N$. In this work, our choice of Δt is based on the CFL condition for the level set function calculation i.e.

$$\Delta t \leq \lambda_{CFL} \frac{\Delta x}{V_{\max}}, \tag{2.24}$$

where λ_{CFL} is the CFL coefficient [43], and $V_{\max}$ is the maximum interface nodal velocity.

The subsequent choice of k_N is such that $\Delta t \leq 1/k_N$ to allow the interface temperature T_I to asymptotically converge to the equilibrium temperature T^*. As discussed earlier, our selection of k_N that satisfies the above condition is $k_N = \frac{1}{\Delta t}$.

In summary, the scheme to select Δt and k_N is the following:

1. Choose λ_{CFL} between 0 and 1. In the 3D diffusion crystal growth under low-undercooling conditions example examined in Section 2.6.5, λ_{CFL} is selected as 0.1. In all other examples, we use $\lambda_{CFL} = 0.3$.
2. Select a time step size as $\Delta t = \lambda_{CFL} \frac{\Delta x}{V_{\max}^{n-1}}$, where $V_{\max}^{n-1}$ is the maximum interface nodal velocity at the previous time level.

3. Select $k_N = \frac{1}{\Delta t}$.
4. Calculate the interface velocity according to Eq. (2.21) (see Section 2.4.2)
5. Use the level set method to update ϕ.

2.4.2 Interface velocity calculation

In the level set method, the interface velocity V should be defined on the whole domain (or a narrow band near the interface). In the present algorithm, V is first computed on the nodes near the interface (depicted as empty circles in Fig. 2.4) using Eq. (2.21). A node is marked as being near the interface if at least one of its neighboring nodes has a different sign of ϕ. Equation (2.21) involves heat fluxes q_s and q_l, equilibrium temperature T^* and average temperature within the diffused interface $\bar{T}_I$. All these variables are computed on the nodes near the interface to obtain V using the methodology discussed in Sections 2.4.2 and 2.4.2. After V is computed on these nodes, it is extended to other nodes using the algorithm discussed in Section 2.4.2.

Calculation of the heat flux jump

We use the following two steps to compute q_s and q_l at the nodes near the interface:

1. For nodes with $\phi \leq -w$, one computes $q_s = k_s \nabla T \cdot \boldsymbol{n}$, where ∇T is the temperature gradient approximated using central differencing or upwind differencing. Similarly, for nodes with $\phi \geq w$, one computes $q_l = k_l \nabla T \cdot \boldsymbol{n}$.
2. Solving $\frac{\partial q_s}{\partial t} + \boldsymbol{n} \cdot \nabla q_s = 0$ in the region $\phi \in [-w, w]$ using the values of q_s in the region $\phi \leq -w$ as boundary condition extrapolates q_s in the normal interface direction. Similarly, solving $\frac{\partial q_l}{\partial t} - \boldsymbol{n} \cdot \nabla q_l = 0$ in the region $\phi \in [-w, w]$ using

the values of q_l in the region $\phi \geq w$ as boundary condition extrapolates q_l in the opposite normal direction to the interface. This one-way extrapolation method is referred to as the 'ghost fluid method' in [43, 17].

Note that after the second step above, q_s and q_l are computed on all nodes belonging to the region $\phi \in [-w, w]$. Since all nodes near the interface (empty circles in Fig. 2.4) are included in this region, q_s and q_l are now computed on all nodes near the interface.

Interpolation of the interface temperature

The equilibrium temperature T^* can be computed from the Gibbs-Thomson relation as follows:

$$T^* = T_m + \varepsilon_c \kappa + \varepsilon_V V^{n-1}. \tag{2.25}$$

As discussed earlier in Section 3.2, we will use T_I (temperature at $\phi = 0$) to approximate $\bar{T}_I$ (average temperature within the diffused-interface). However, in general, the nodes near the interface will not satisfy $\phi = 0$. So interpolation is necessary to obtain T_I. This can be easily computed using the following equation:

$$T_I = T - (\nabla T \cdot \boldsymbol{n})\phi, \tag{2.26}$$

where ∇T is calculated using simple differencing techniques.

Extending the interface velocity away from the interface

With q_s, q_l, T^* and T_I computed on the nodes near the interface, the interface velocity can be calculated at these nodes using Eq. (2.21). One can then perform a two-way extrapolation to extend the interface velocity to every other nodal point as described in [43, 77]. In the present simulation, we use a slightly faster method defined by the following steps (see Fig. 2.4):

1. Extend the interface velocity to band within Δx away from the interface. For all the points within the band Δx but not near the interface (for example, at point P in Fig. 2.4), draw a circle with radius $1.2\Delta x$, and calculate the average interface velocity for all the points near the interface and within this circle as its velocity. For example, $V_P = \frac{V_Q+V_W}{2}$.
2. Extend the interface velocity to a band within $2\Delta x$ away from the interface. For all the points within the band $2\Delta x$ and outside the band Δx, draw a circle with radius $1.2\Delta x$, and calculate the average interface velocity for all the points within band Δx and within this circle as its velocity.

Using this method, one can extend the interface velocity $n\Delta x$ away from the interface with just n iterations. In the simulations of Section 3.4, the narrow band used to extend the velocity for solving the level set equation was $3\Delta x$ on each side of the interface. Although accuracy is sacrificed, this method is faster than the ghost fluid method, which is constrained by the CFL condition. After extension of the interface velocity, the velocity needed for the level set calculation can be defined within a narrow band. As an alternative method, one can use the fast-marching method to extend the interface velocity using a heap data structure to achieve time complexity of only $O(NlogN)$, where N is the number of nodes where the velocity is extended [56].

2.4.3 Incorporating melt convection

It has been shown from numerical simulations [2, 3, 6, 66, 35] and experiments [36] that fluid flow has an important effect on crystal growth. There are a variety of ways to

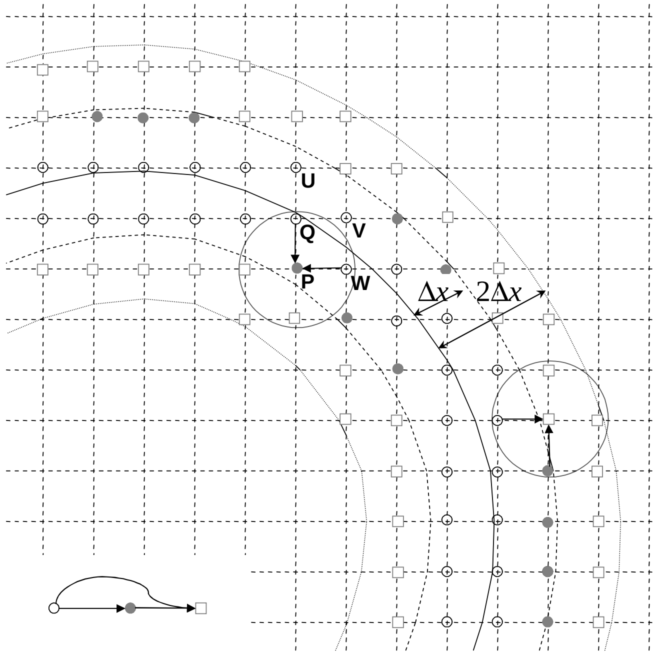


Figure 2.4: Extending the interface velocity away from the freezing interface. Note that the velocity V is first computed at the points depicted with empty circles, then at the solid circle points and finally at the points shown with squares.

incorporate convection using the phase-field method or front-tracking methods. In [66], the solid is treated as a highly-viscous liquid by letting the viscosity depend on the phase field variable in the standard Navier-Stokes equations. In [6], the no-slip condition between the melt and the solid was realized via a drag resistivity in the diffused interface region. In [2], the Navier-Stokes equations are solved in two steps (with the first step considering only the advection and viscous terms and a trial pressure, and the second step considering only the pressure gradient). In these two steps, the unprojected velocity and the pressure gradient are multiplied by an index function to set the velocity in the solid to zero.

In this work, we treat the diffused-interface as a narrow ‘mushy zone’. Volume-averaging is then applied to the whole region. The velocity in the solid region is set to zero, so that no-slip condition is applied at the solid/liquid interface. The formulation is briefly summarized below with more details provided in [72]. The flow equations are first re-cast in dimensionless form as follows:

$$\begin{aligned}\frac{\partial \boldsymbol{v}(\boldsymbol{x},t)}{\partial t}+\nabla\cdot\left(\frac{\boldsymbol{v}(\boldsymbol{x},t)\boldsymbol{v}(\boldsymbol{x},t)}{\Phi}\right) &= -\nabla p(\boldsymbol{x},t)+\frac{p(\boldsymbol{x},t)}{\Phi}\nabla\Phi \\ &\quad +\nabla\cdot[Pr(\ \nabla\boldsymbol{v}(\boldsymbol{x},t)+(\nabla\boldsymbol{v}(\boldsymbol{x},t))^T\ )] \\ &\quad -\frac{(1-\Phi)^2}{\Phi^2}\frac{Pr}{Da}\boldsymbol{v}(\boldsymbol{x},t)-\Phi Pr Ra_T\ \theta(\boldsymbol{x},t)\ \boldsymbol{e_g},\end{aligned}$$

where Pr is the Prandtl number defined as $\frac{\nu_l}{\alpha_l}$, Da is the Darcy number defined as $\frac{\alpha_l}{D_l}$ and Ra_T is the thermal Rayleigh number defined as $\beta_T|g|(T_0-T_m)L^3/\nu_l\alpha_l$. The function spaces $S_{\boldsymbol{v}}$ and S_p are then introduced as follows:

$$\begin{aligned}S_{\boldsymbol{v}} &\equiv \{\boldsymbol{v}|\boldsymbol{v}\in L_2^{nsd}, \mathrm{div}\boldsymbol{v}\in L_2, \boldsymbol{v}=0 \text{ on } \partial\Omega\}, \\ S_p &\equiv \{p|p\in L_2, \int_\Omega p d\Omega = 0\}.\end{aligned}$$

The classical Galerkin formulation for the flow problem can then be stated as: Find $\boldsymbol{V}\equiv\{\boldsymbol{v},p\}\in S_{\boldsymbol{v}}\times S_p$ such that for all $\boldsymbol{W}=\{\boldsymbol{w},q\}\in S_{\boldsymbol{v}}\times S_p$ $B(\boldsymbol{W},\boldsymbol{V})=L(\boldsymbol{W})$ holds, where

$$\begin{aligned}B(\boldsymbol{W},\boldsymbol{V}) &= \int_\Omega \boldsymbol{w}\cdot\left(\frac{\partial\boldsymbol{v}}{\partial t}+\boldsymbol{v}\cdot\nabla\left(\frac{\boldsymbol{v}}{\Phi}\right)+\frac{(1-\Phi)^2}{\Phi^2}\frac{Pr}{Da}\boldsymbol{v}\right)d\Omega-\int_\Omega p\nabla\cdot\boldsymbol{w}d\Omega \\ &\quad +\int_\Omega Pr\nabla\boldsymbol{w}\cdot(\nabla\boldsymbol{v}+\nabla\boldsymbol{v}^T)d\Omega+\int_\Omega q\nabla\cdot\boldsymbol{v}d\Omega, \\ L(\boldsymbol{W}) &= \int_\Omega\frac{p}{\Phi}\nabla\Phi\cdot\boldsymbol{w}d\Omega-\int_\Omega\boldsymbol{w}\cdot\Phi Pr Ra_T\theta\boldsymbol{e_g}d\Omega.\end{aligned}$$

In the finite element implementation of the Navier-Stokes equations, stabilizing techniques are needed to accommodate equal-order interpolation velocity-pressure elements. A stabilized FEM technique for porous media flows is presented in [72] and is briefly discussed below for completeness. After introducing a modified pressure space S'_p as follows:

$$S'_p \overset{\text{def}}{=} \{p|p\in H^1(\Omega), \int_\Omega p d\Omega = 0\}, \tag{2.27}$$

the stabilized weak form is the following: Find $\boldsymbol{V} = \{\boldsymbol{v}, p\} \in S_{\boldsymbol{v}} \times S'_p$ such that for all $\boldsymbol{W} = \{\boldsymbol{w}, q\} \in S_{\boldsymbol{v}} \times S'_p$ the following holds:

$$B_{\text{stab}}(\boldsymbol{W}, \boldsymbol{V}) = L_{\text{stab}}(\boldsymbol{W}), \tag{2.28}$$

with:

$$\begin{aligned} B_{\text{stab}}(\boldsymbol{W}, \boldsymbol{V}) &= B(\boldsymbol{W}, \boldsymbol{V}) + \int_\Omega \mathcal{F}(\boldsymbol{v}, p) \cdot \mathcal{G}(\boldsymbol{w}, q) d\Omega + \int_\Omega \tau_5 \nabla \cdot \boldsymbol{v} \nabla \cdot \boldsymbol{w} d\Omega, & (2.29) \\ L_{\text{stab}}(\boldsymbol{W}) &= L(\boldsymbol{W}) + \int_\Omega \{\frac{p}{\Phi} \nabla \Phi - \Phi \, PrRa_T \, \theta \boldsymbol{e_g}\} \cdot \mathcal{G}(\boldsymbol{w}, q) d\Omega, & (2.30) \end{aligned}$$

where $\mathcal{F}$ and $\mathcal{G}$ are defined as:

$$\begin{aligned} \mathcal{F}(\boldsymbol{v}, p) &= \frac{\partial \boldsymbol{v}}{\partial t} + \boldsymbol{v}_* \cdot \nabla \left(\frac{\boldsymbol{v}}{\Phi} \right) + \nabla p + \frac{(1-\Phi)^2}{\Phi^2} \frac{Pr}{Da} \boldsymbol{v} - Pr \nabla^2 \boldsymbol{v}, & (2.31) \\ \mathcal{G}(\boldsymbol{w}, q) &= \tau_1 \boldsymbol{v}_* \cdot \nabla \left(\frac{\boldsymbol{w}}{\Phi} \right) - \tau_2 \frac{(1-\Phi)^2}{\Phi^2} \frac{Pr}{Da} \boldsymbol{w} - \tau_3 Pr \nabla^2 \boldsymbol{w} + \tau_4 \nabla q, & (2.32) \end{aligned}$$

with $\boldsymbol{v}_*$ a divergence-free velocity, which in the implementation of Eq. (2.28) at a given time is usually taken as the known velocity at the previous time step. The particular values of the parameters $\tau_1, \ldots, \tau_5$ used in this work are given in [72]. Four-node bilinear finite elements (in 2D) and eight-node brick trilinear finite elements (in 3D) were used for both velocity and pressure interpolations. In closing, we note that in the problems examined in Section 3.4, $Ra_T = 0$ and the flow is induced by inlet velocity conditions.

2.5 Summary of the algorithm

A finite difference scheme is used for the level set calculations so that higher-order accuracy (third-order WENO scheme in space and third-order Runge-Kutta in time) can be achieved [44]. The same structured grid is used for both finite difference approximations in the level set calculations and the finite element approximations of the heat and flow problems. A summary of the overall algorithm is provided below.

1. Update level set variable ϕ.
    (a) Copy ϕ to ϕ^{n-1}.
    (b) Copy V to V^{n-1}.
    (c) Determine the time step Δt and extended velocity V
2. Copy Φ to Φ^{n-1}. Update Φ using ϕ according to Eq. (2.11).
3. Solve the heat equation utilizing Φ^{n-1} and Φ. A fully implicit scheme is used in these calculations. The tolerance for the residual vector is taken as $||\boldsymbol{b}||_2 \leq 10^{-5}$. If no fluid flow effects are incorporated, the discretized equations are only solved once.
4. Use ϕ to construct an artificial mushy zone and solve the fluid flow equations with tolerance of residual vector $||\boldsymbol{b}||_2 \leq 10^{-5}$, permeability $K_0 = 10^{-5}$, and $Ra_T = 0$.
5. Set $t = t_{n-1} + \Delta t$. Return to step 1.

2.6 Numerical examples

2.6.1 Solidification in a corner

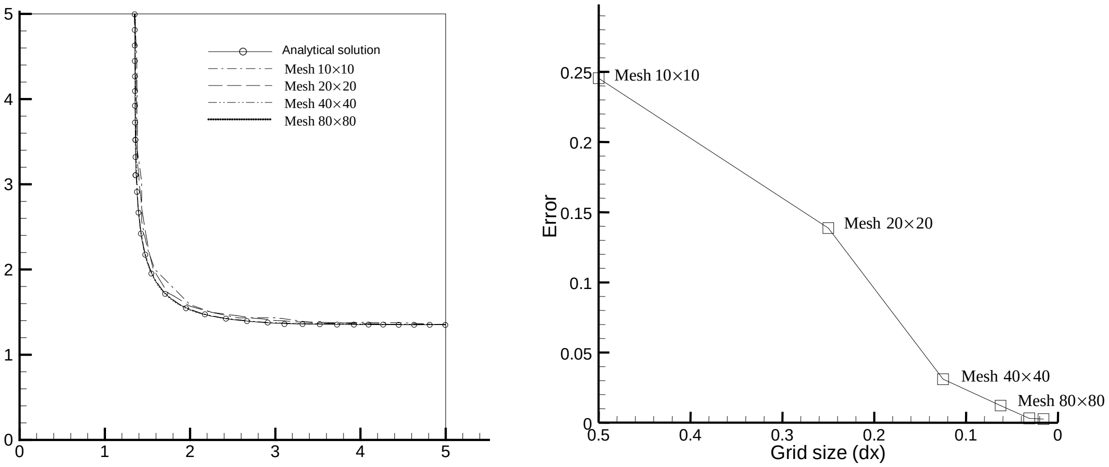
This example is the solidification of a pure material in an infinite corner region with $k_s = k_l = 1$, $c_s = c_l = 1$, $L = 0.25$ and $\rho = 1$. The melting and initial temperatures are $T_m = 0$ and $T_{in} = 0.3$, respectively. A constant temperature condition $T_0 = -1$ is applied to the two boundary sides of the region. The analytical solution for the non-dimensionless interface position is given in [52].

To simulate this infinite corner Stefan problem, we use a domain of 5×5 discretized with a quadrilateral grid. At the left side and bottom sides of the domain, the temperature is kept at T_0, whereas the top and right sides are assumed to be adiabatic. This is only an approximation of the original problem with a solution that at early times should compare well with the analytical solution of the infinite corner problem.

We considered grids of different sizes. Figure 2.5(a) shows that the numerical solution converges to the analytical solution very well. In Fig. 2.5(b), we define the error as the maximum distance of the calculated interface position from the analytical solution. This is computed by (1) finding all the elements cut by the zero level set, (2) interpolating points which are on element edges and satisfy $\phi = 0$, and (3) calculating distance of the interpolated points to the analytical solution, which is discretized into 100 points. As shown in Fig. 2.5(b), the error drops almost quadratically ($\approx 2.4(\Delta x)^2$) with the grid size Δx.

2.6.2 Growing of a circle and a sphere in an undercooled melt

The dynamical evolution away from an unstable steady state was studied in [27, 10]. It was found that under conditions favoring rapid solidification in 2D, the radius of the growing circle satisfies $R(t) \propto t$. In 3D, the radius of the growing sphere satisfies $R(t) \propto \sqrt{t}$. Using the presented numerical scheme, we simulated the growth of an initial seed with radius 1 with initial temperature 0 located in the middle of the domain $[-5, 5]^{\mathrm{nsd}}$, where the number of spatial dimensions nsd$=2$ for the growing circle example and nsd$=3$ for growing sphere. The rest of the domain is at initial temperature $T_0 = -0.5$.



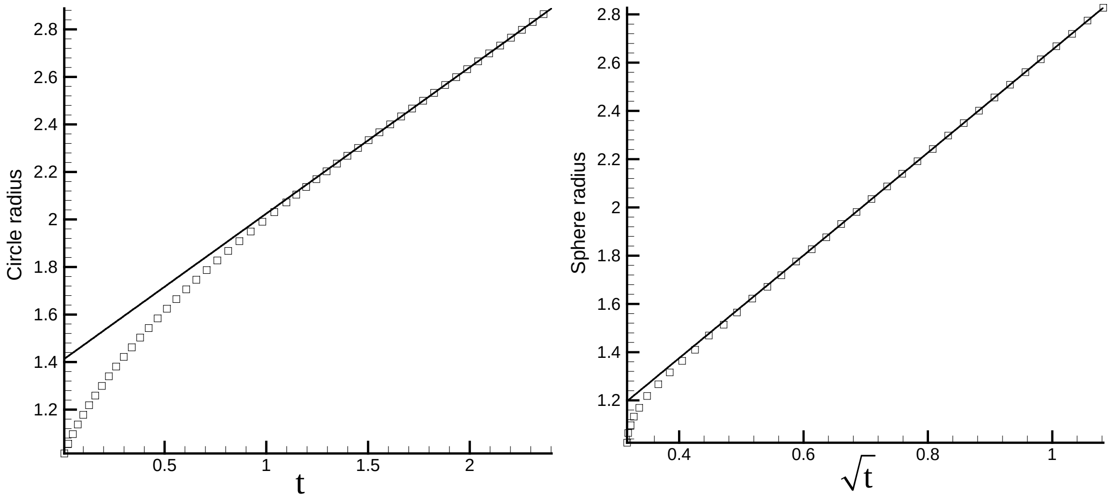
(a) (b)

Figure 2.5: Convergence study of the infinite corner problem (time 0.9) (a) front position using various mesh sizes (b) maximum distance of the calculated interface from the analytical solution versus grid size.

The solid/liquid interface is always at temperature $T_m = 0$ without surface tension or kinetic undercooling. A constant temperature boundary condition $T = -0.5$ is applied at the boundary of the domain. Other parameters in the calculation are $k_s = k_l = 1$, $c_s = c_l = 1$, $L = 1$ and $\rho = 1$. We use a 100×100 mesh for the 2D simulation and $100 \times 100 \times 100$ mesh for the 3D simulation. Our results shown in Fig. 2.6 verify that after a "burn-in" period, the radius of the solidifying circle grows linearly with time in 2D and the radius of the solidifying sphere in 3D grows linearly with the square root of time.

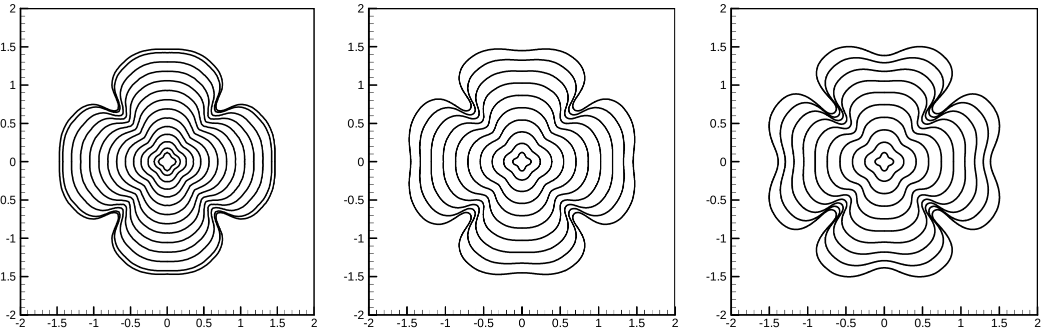
2.6.3 Crystal growth in an undercooled melt: Effects of anisotropy and surface tension

This example was originally addressed in [29] using a front-tracking method and re-examined in [12] using an implementation of the level set method. The material param-



(a) Growing circle (b) Growing sphere

Figure 2.6: Radius of a solidifying circle and sphere in an undercooled melt.



(a) Mesh 100×100 (b) Mesh 200×200 (c) Mesh 400×400

Figure 2.7: Crystal growth in the presence of surface tension.

eters defining the problem are $k_s = k_l = 1$, $c_s = c_l = 1$, $L = 1$, $\rho = 1$ and $T_m = 0$. On the freezing interface Γ we consider the classical Gibbs-Thomson relation given in Eq. (2.5) with $\varepsilon_c = \varepsilon_V = -0.002$. The computational domain is taken as $[-2, 2] \times [-2, 2]$. Insulated boundary conditions are considered at all sides of the two-dimensional domain.

At time zero, we consider a small solid seed in the middle of the computational domain. Its geometry is described as follows:

$$x(s) \quad = \quad (R + P\cos(8\pi s))\cos(2\pi s), \tag{2.33}$$

$$y(s) \;=\; (R + P\cos(8\pi s))\sin(2\pi s), \tag{2.34}$$

where $R = 0.1$ and $P = 0.02$. The initial temperature of the seed is taken as 0 and the initial temperature of the undercooled melt as -0.5. In the implementation of this example, we considered three different finite element meshes (100×100, 200×200 and 400×400). These studies allow us to investigate and report on the mesh-dependency of the results obtained with the present methodology.

For these calculations $\lambda_{CFL} = 0.3$. To accelerate our simulation, the time step size is adjusted automatically according to the CFL condition thus the lines shown in Fig. 2.7 are not evenly separated with time. The growth results shown in [29, 12] are in increments of 0.04 up to a final time of 0.8. Notice in Fig. 2.7 that at time 0.8, a steady-state has been reached and as expected the area of the computed solid region is exactly half of the total area of the domain. However, the area of the solid region at steady-state (time 0.8) is significantly different for the morphologies with low mesh resolution 100×100 comparing with high mesh resolution 400×400 in both [29] and [12]. The reason is that when Dirichlet boundary condition is applied at the solid/liquid interface, discretization error leads to high mesh-dependency. In our algorithm, we avoided applying Dirichlet boundary conditions on the interface so that energy conservation is satisfied leading to mesh-insensitive results.

We note that the grid refinement results shown in Fig. 2.7 compare well with those given in [12]. The morphologies obtained in [29] using a front-tracking technique appear to have a much higher mesh-dependency than the results reported here. The difference in published results [29, 74, 12, 76] for this problem suggest that its solution is highly sensitive to perturbations during the solution process and that the problem is indeed a non-trivial one.

The above simulations were repeated but without the effects of surface tension, i.e.

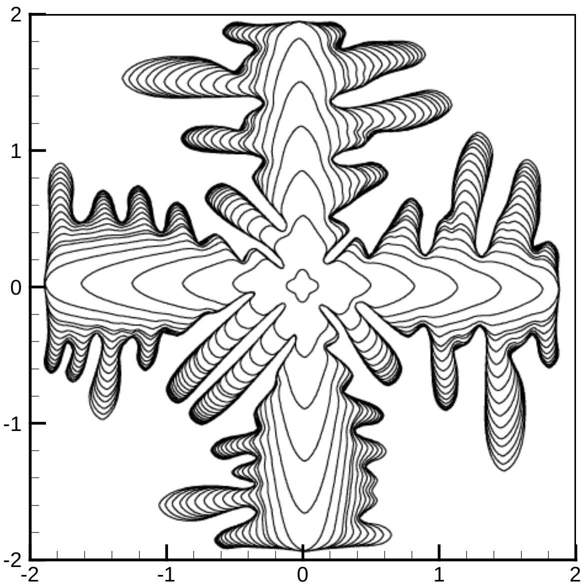


Figure 2.8: Unstable crystal growth without curvature effect using a mesh of 400× 400.

with $\varepsilon_c = 0$ and all other conditions in Eq. (2.5) as before. Figure 2.8 shows the results of this simulation for a mesh size 400×400. Comparing with the results in Fig. 2.7, it is seen that the crystal as expected is growing in a much more unstable mode.

Six-fold symmetric growth

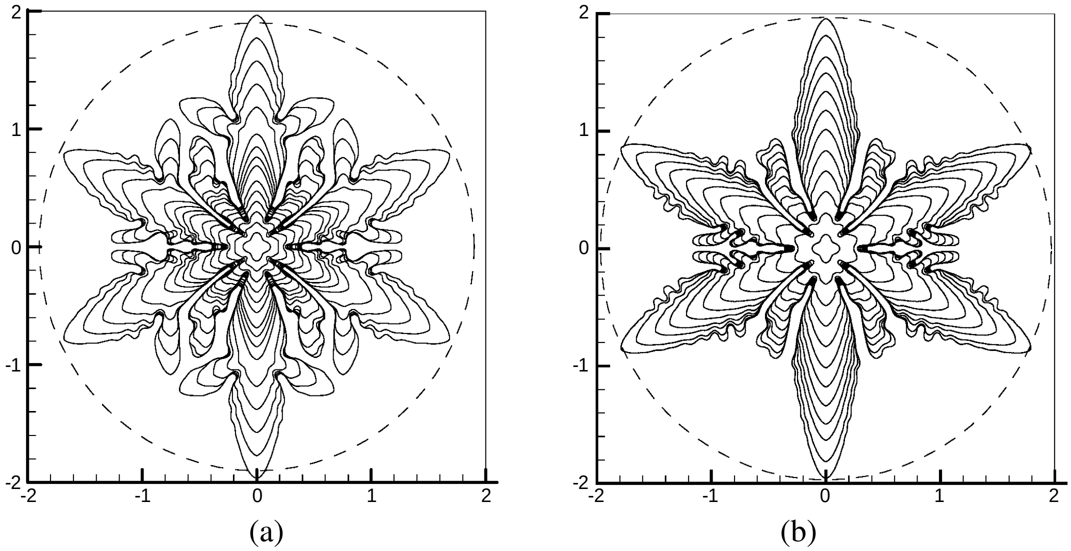
We also computed the solution to a crystal growth problem under anisotropy with six-fold symmetry examined previously in [76]. The problem definition is similar to the earlier example.

At time zero, a small solid seed is put in the middle of the computational domain $[-2, 2] \times [-2, 2]$. Its geometry is described from Eqs. (2.33)-(2.34). The initial temperature of the seed is taken as 0 and the initial temperature of the undercooled melt as -0.8. The kinematic undercooling coefficient is constant with value $\varepsilon_v = -0.001$. The surface tension (curvature undercooling coefficient) is specified by the following anisotropic model with six-fold symmetry:

$$\varepsilon_c = -0.001\{1.0 + 0.4[\tfrac{8}{3}\sin^4 3(\theta - \pi/2) - 1.0]\}$$

All other material properties are normalized as 1.

In this example, the initial seed has a four-fold symmetry while ε_c has a six-fold symmetry. This difference allows us to study how initial perturbations affect the crystal growth. Using a grid of 400 × 400 and 800 × 800 and at final time 0.036, we obtain the crystal interfaces shown in Fig. 2.9.



(a) (b)

Figure 2.9: Evolution of four-fold initial seed with six-fold surface tension anisotropy (a) Mesh 400 × 400 (b) Mesh 800 × 800.

From Fig. 2.9, we can conclude that the primary dendrite arms are determined by the growth mechanism (anisotropy), while the initial perturbations only affect the formation of secondary dendritic arms.

As pointed out in [76], the formed secondary dendritic arms are different for coarse and fine grid simulations (see Figs. 2.9 (a) and (b)), while the primary dendrite tips are growing with the same velocity for both grids (they all reach the computational boundary at time about 0.036). However, note that in the work of [76] using a front tracking method with markers, the total time for the crystal to reach the computational boundary varied from 0.035 with grid 800 × 800 to 0.045 with grid 400 × 400. The apparent improvement provided by the present methodology may be due to its energy conserving nature.

2.6.4 Two-dimensional steady-state dendritic growth: Comparison with solvability theory

Steady-state features of dendritic growth have been studied extensively using phase-field models [48, 30] and the level set method [32]. These calculations have been in good agreement with the predictions of microscopic solvability theory. We will show here that an excellent agreement is also obtained using the developed energy conserving level set method. In the problem considered, the equilibrium temperature in the freezing interface Γ incorporates the effects of anisotropy as

$$T^* = -d_0(1 - 15\epsilon \cos 4\theta)\kappa, \tag{2.35}$$

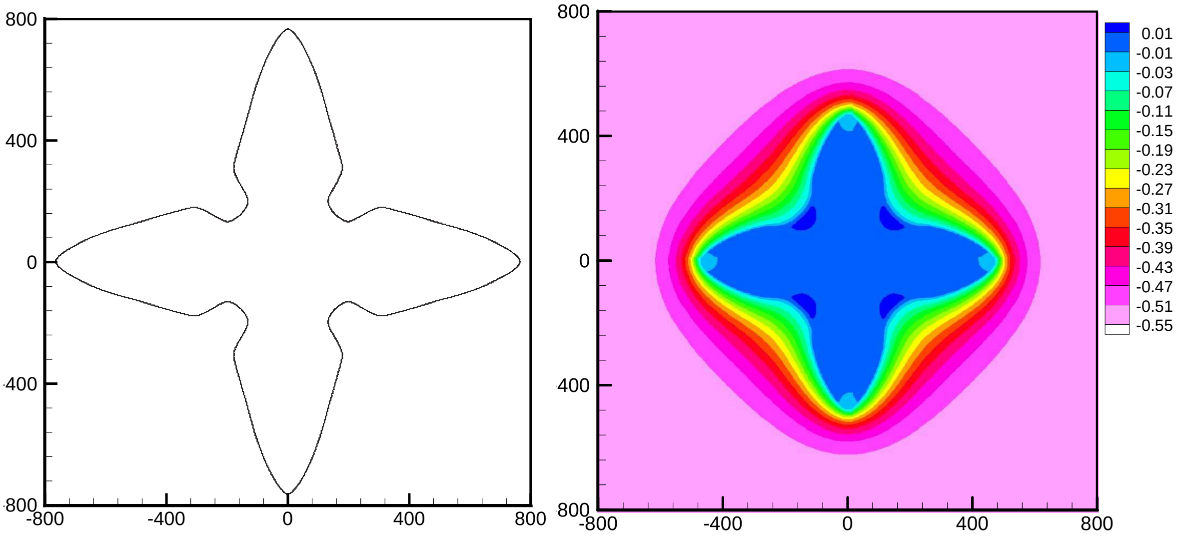
with θ the angle between the outward normal and the x direction, $d_0 = 0.5$ and $\epsilon = 0.05$. The initial liquid temperature and boundary temperature (thus the undercooling considered) is $\Delta = 0.55$. The remaining material parameters are selected as $k_s = k_l = 1$, $c_s = c_l = 1$, $L = 1$ and $\rho = 1$. The obtained results are plotted using a normalized velocity $\tilde{V} = \frac{\rho c d_0 V}{k}$, a normalized position $\tilde{x} = \frac{x}{d_0}, \tilde{y} = \frac{y}{d_0}$, and dimensionless time $\tilde{t} = \frac{k}{\rho c d_0^2} t$. These dimensionless variables are also used in all of the following examples.

Our results shown in Figs. 2.10(a)-(d) obtained with a mesh of 800 × 800 compare fairly well with the numerical results obtained using the phase field method [48, 30] and the level set method [32]. From the plot of the computed temperature field in Fig. 2.10b, we can observe that there is an undercooling at the dendrite tips due to positive curvature, whereas at places with a negative curvature, the melt temperature is greater than zero (the melting point). In these simulations, the interface velocity is extended away from the interface to a distance of $3\Delta x$ on each side for solving the level set equation. At other places, the interface velocity is taken to be zero. From Fig. 2.10d, we can see that the interface velocity achieves a maximum at the dendrite tips. This value is about 0.016 for the results obtained using a mesh of 400×400 and 0.017 for the results using a mesh of 800×800. According to the solvability theory, the steady-state velocity at the dendrite tips is 0.017 [32].

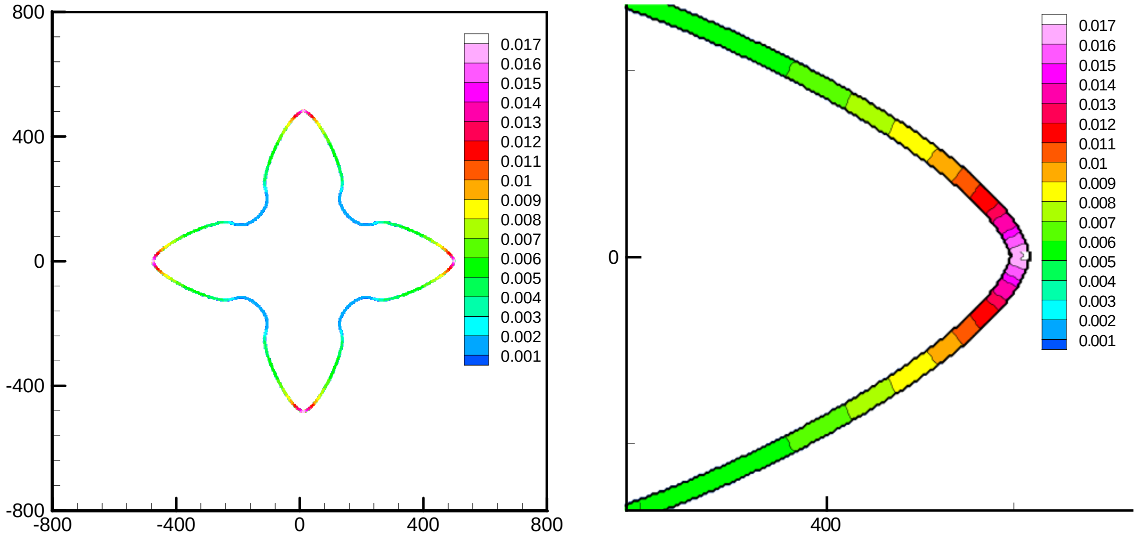
The dendrite tip velocity is found to be very sensitive to the degree of undercooling. Using a slightly higher undercooling of 0.65, the steady-state dendrite tip velocity increases to about 0.047 as shown in Fig. 2.11. For the results in this figure, we have increased the computational domain to 1200×1200 using a mesh of 600×600. In this figure, we can find that the dendrite tip velocity settles to a value 0.017 at dimensionless time 22000. In the earlier reported case of a domain of 800×800, the thermal boundary layer eventually interacts with the system boundary and the asymptotic nature of the solution is lost.

Finally, in order to demonstrate the ability of the present method to enforce weakly the interface temperature condition, we also documented the equilibrium temperature at the dendrite tips for this example with undercooling 0.55 (Fig. 2.12).

While as discussed in Remark 2, an iterative process may be needed in general to enforce the interface temperature condition, in this example an explicit non-iterative process was sufficient. The equilibrium temperature falls in the range of $[-0.042, -0.002]$, which only varies about $0.04/0.55 = 7.2\%$ of the total undercooling. This small varia-



(a) Final crystal shape (b) Temperature field



(c) Growth velocity (d) Velocity near tip

Figure 2.10: Crystal growth at undercooling Δ = 0.55.

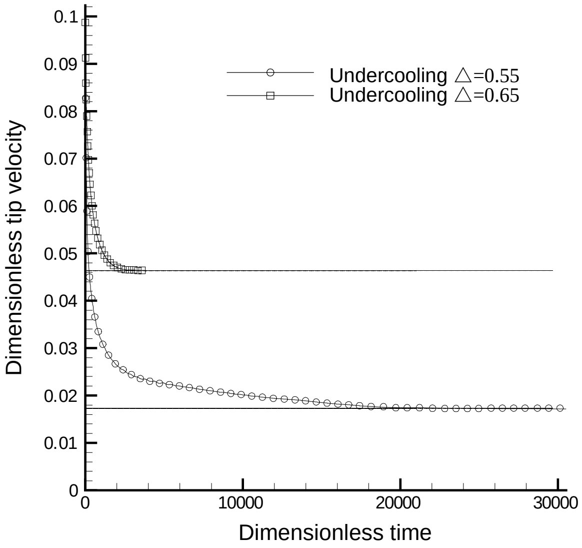


Figure 2.11: Time evolution of the dimensionless dendrite tip velocity for various undercoolings.

tion of the equilibrium temperature did not require the use of an iterative process such as the one shown in Fig. 2.3.

It has been pointed out in [32] that phase-field asymptotics for unequal solid/liquid diffusivities lead to computationally inconvenient forms and require extra grid resolution. The level set method by its nature avoids this difficulty. In order to demonstrate the ability of the present methodology to model unequal diffusivities, we calculated the same 2D crystal growth case using various diffusivities as shown in Fig. 2.13. From this figure, we can see that an increase of the liquid diffusivity tends to make the dendrite tips sharper, while an increase of the solid diffusivity would only make the dendrite tips a little bit fatter. A change of the liquid diffusivity was shown to affect the growth pattern more than a change of the solid diffusivity.

Off-axis solidification growth

In order to demonstrate that the presented algorithm works properly for off-axis

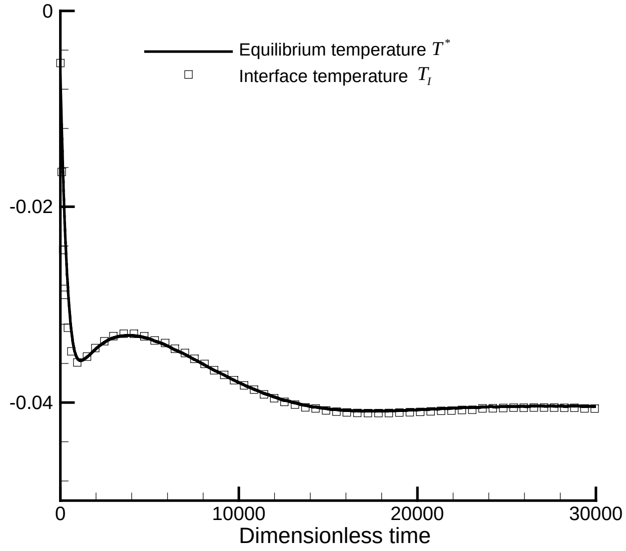


Figure 2.12: Equilibrium temperature at dendrite tips of the 2D dendritic growth with undercooling 0.55.

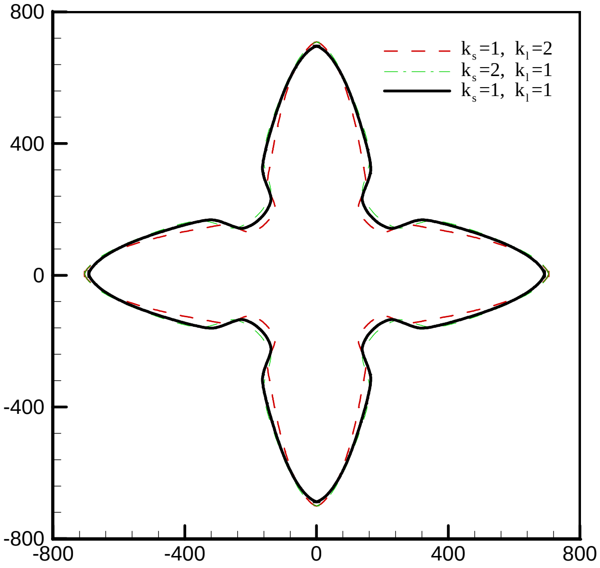
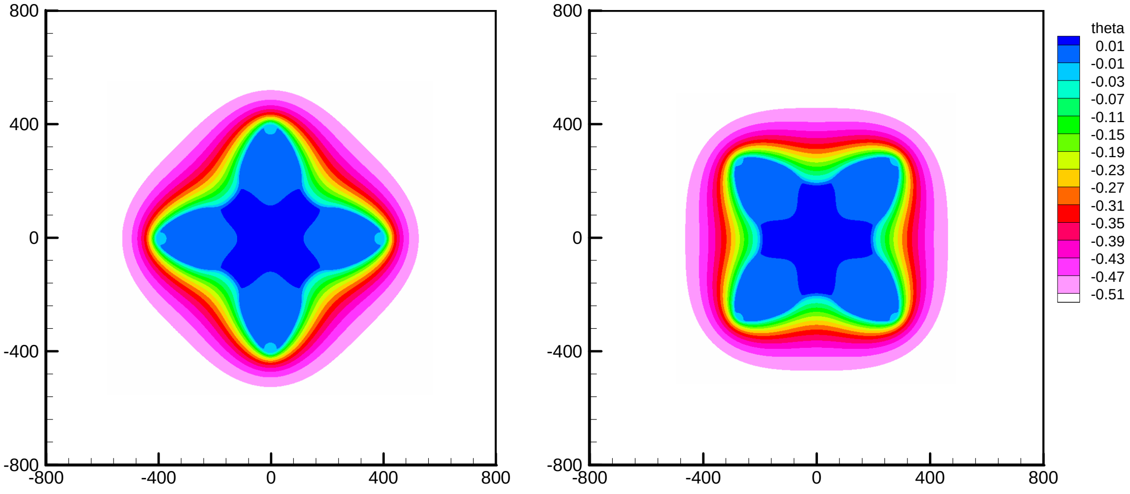


Figure 2.13: Dendrite growth with unequal solid/liquid conductivities.

growth, we recomputed the above example with a rotated surface tension anisotropy as follows:

$$T^* = -d_0\left(1 - 15\epsilon\cos(4(\theta + \frac{\pi}{4}))\right)\kappa. \tag{2.36}$$

The temperature field (at time about 10000) obtained by the original anisotropy and rotated anisotropy are shown in Figs. 2.14 (a) and (b). From Figs. 2.14 (a) and (b), we can conclude that the crystal growth is not affected by the grid orientation.



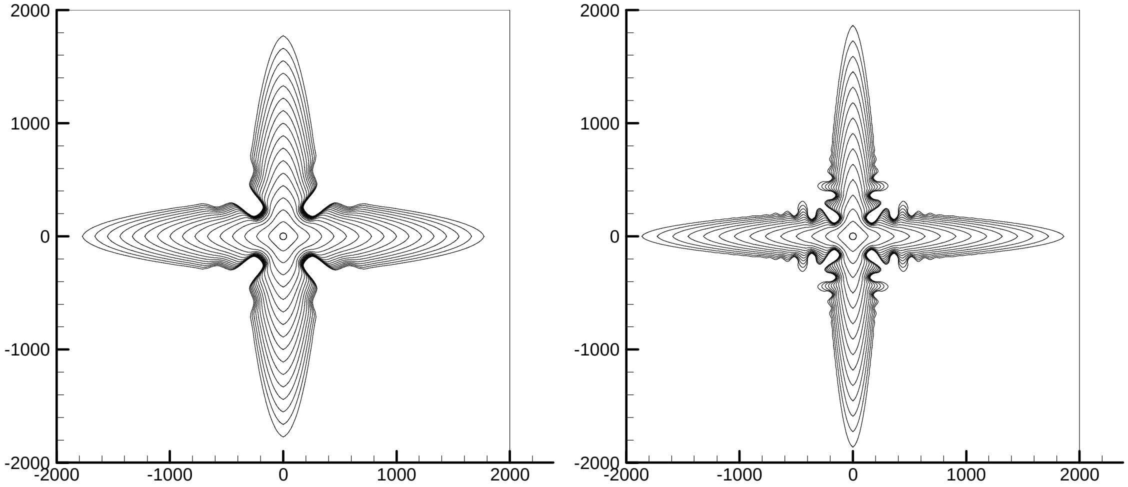
(a) (b)

Figure 2.14: Temperature field for the 2D solvability case with undercooling 0.55 (a) normal surface tension anisotropy (b) rotated surface tension anisotropy.

Formation of secondary dendrites

The formation of secondary dendrites is of great interest to many researchers, e.g. the mechanical properties are related with secondary dendrite arm spacing. In the solvability problem examined earlier with undercooling 0.55 and a computational domain of 800 × 800, no secondary arms were formed as shown in Fig. 2.10. In order to capture the formation of secondary dendritic arms, the computational domain is now increased to 2000 × 2000 using a grid of 800 × 800. When the tips are about 800 away from the center, the dendrite shapes are exactly the same as in Fig. 2.10(a). After that, small perturbations starts growing (see Fig. 2.15(a)). However, these perturbations do not evolve into secondary dendrites. This is due to the particular undercooling considered.

Indeed, we performed another run with the same increased domain and an undercooling of 0.80. In this case, the perturbations eventually evolved into secondary dendrites as shown in Fig. 2.15(b). Calculations with such increased computational domain are given in [30, 49], however only in the adaptive finite element calculations of [49] secondary arms are predicted.



(a) $\Delta = 0.55$ (b) $\Delta = 0.80$

Figure 2.15: Front position evolution for the solvability problem using an enlarged domain and two-different undercoolings.

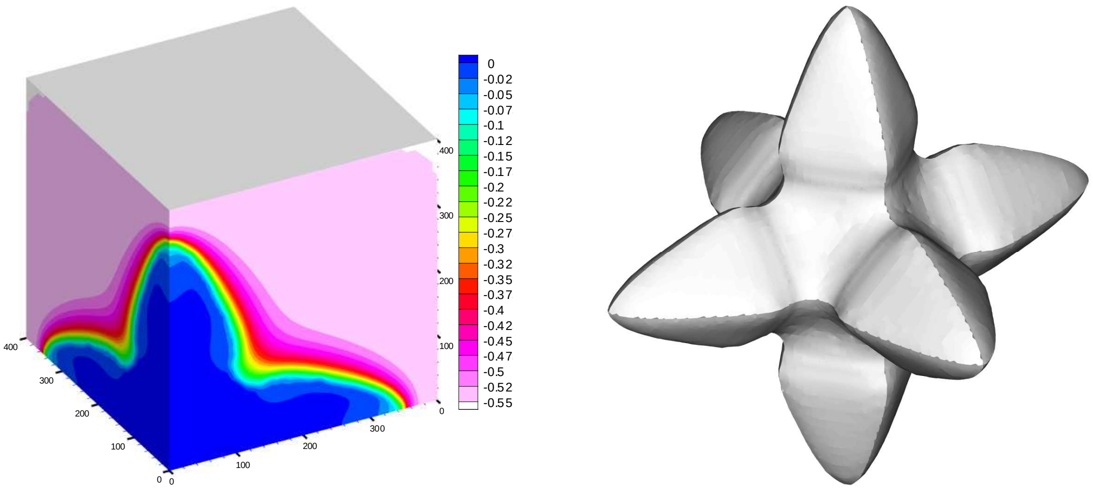
2.6.5 Three-dimensional dendritic growth

High-undercooling case

We herein examine the 3D case of the earlier solvability example using a mesh of $60 \times 60 \times 60$ with an undercooling of 0.55. To incorporate anisotropy in this 3D solidification growth example, we consider the following equilibrium temperature [3]:

$$T^* = -d_0(1 - \epsilon(4(n_1^4 + n_2^4 + n_3^4) - 3))\kappa, \quad \text{on } \Gamma, \tag{2.37}$$

where $d_0 = 0.5$, $\epsilon = 0.05$ and n_1, n_2, n_3 are the components of the normal unit vector



(a) Temperature field (b) Crystal shape

Figure 2.16: Temperature field and crystal shape at time $t = 105$ for 3D crystal growth at an undercooling $\Delta = 0.55$.

along the x, y, z axes, respectively, calculated at each point on the freezing interface. The whole domain considered is $[-400, +400]^3$. Using symmetry, only $1/8$ of the whole domain is calculated. Using an IBM T41 with a 1.7GHz CPU and 512 MB memory, the calculation time for the crystal to reach the state shown in Fig. 3.27 is about 10 hours. In this case, the undercooling is relatively large, so that the temperature field extends spatially as shown in Fig. 3.27.

Low-undercooling

It was observed that at low-undercoolings, the length scale to model the underlying transport processes is several orders of magnitude the tip radius [46]. However, the mesh step size Δx should be based on the tip radius in order to model the dendrite shape. This requires a very fine 3D grid, and substantially increases the computational difficulty for such problems. The following case with an undercooling of 0.05 is first presented in [46] to illustrate the computational power of the so called '*multiscale random-walk algo-*

rithm' to address previously computationally unreachable range of low-undercoolings. Other parameters considered are the following [46]:

$$T^* = -d_0(1-3\epsilon)[1-4\epsilon(n_1^4+n_2^4+n_3^4)/(1-3\epsilon)]\kappa, \text{ on } \Gamma, \tag{2.38}$$

where $d_0 = 1$ and $\epsilon = 0.025$. Again using symmetry, we take a computation domain of $[0, +20000]^3$ with mesh 120^3 using 2 V1 nodes in the Cornell CTC center (with 4×500 MHz CPU on each node). Domain decomposition using 8 domains (processes) is used in this calculation.

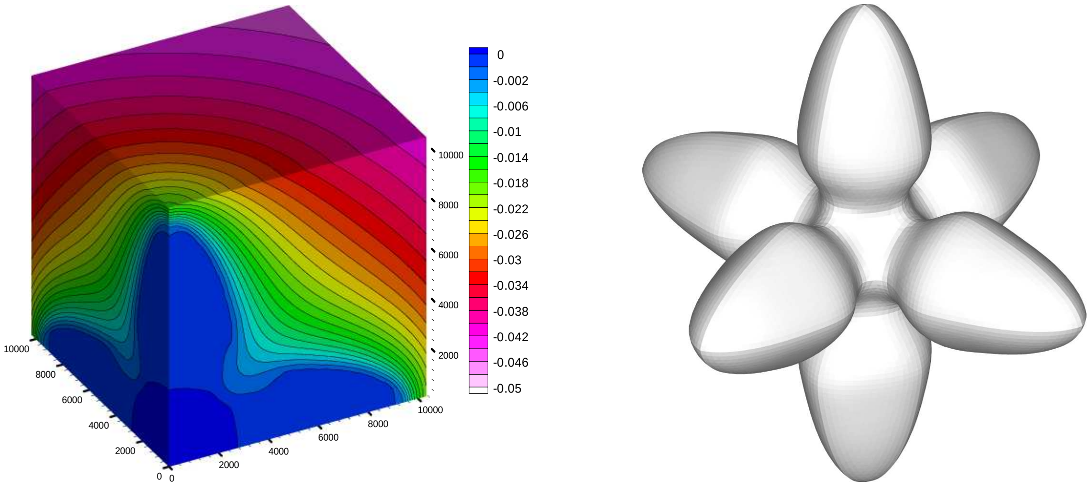


Figure 2.17: Temperature field and crystal shape at time $t = 2.4 \times 10^8$ for 3D crystal growth at an undercooling $\Delta = 0.05$.

For this low-undercooling case, the interface velocity is very small compared to the high-undercooling case. A smaller CFL coefficient of $\lambda_{CFL} = 0.1$ is thus used in this case to reduce the computation time step. The temperature field on the first process and the crystal shape is shown in Fig. 2.17 with domain $[0, +10000]^3$ using a mesh of 60^3. Since the other 7 processes provide only little additional information on temperature, the results of those processes are not shown here. The dimensionless tip velocity and tip radius as a function of time are shown in Fig. 2.18. These results are very close to those reported in [46]. The steady-state tip velocity and dendrite tip radius agree

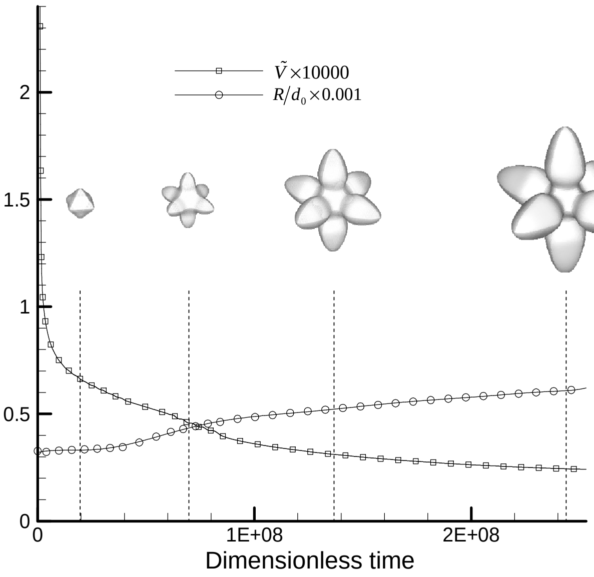


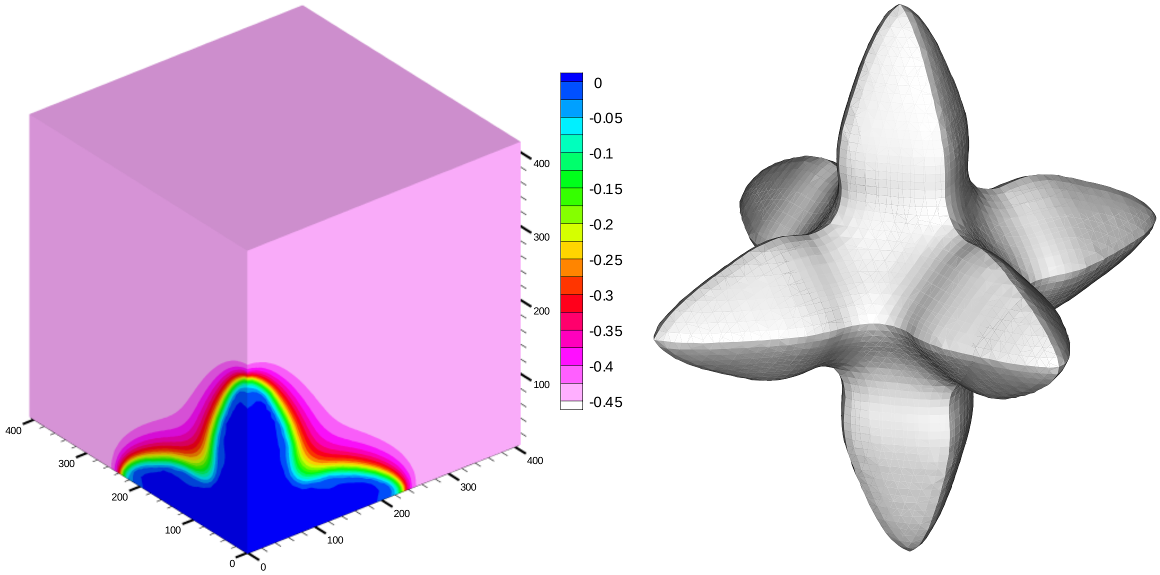
Figure 2.18: Time evolution of the dimensionless tip velocity and dimensionless tip radius for 3D crystal growth at an undercooling $\Delta = 0.05$.

well with values reported in [46]. This demonstrates that the present methodology can successfully be applied to undercoolings as low as 0.05 thus bridging the disparity of length scales for modelling the tip radius and the thermal boundary layer. This ability is important since there is direct experimental relevance in this order of undercoolings [46].

We also considered the equilibrium condition in Eq. (2.38) but with an undercooling of 0.45 and $\epsilon = 0.04$. This case was examined earlier in [28]. The obtained results in Fig. 2.19 show that the predicted dendrite shapes compare fairly well with those reported in [28] using the phase field method.

2.6.6 Two-dimensional crystal growth with convection

Beckermann et al. [6] presented the first calculations of dendritic growth in the presence of convection with a diffused-interface using the phase-field method. An implementation of dendritic growth with fluid flow using a front-tracking method is given in [3].



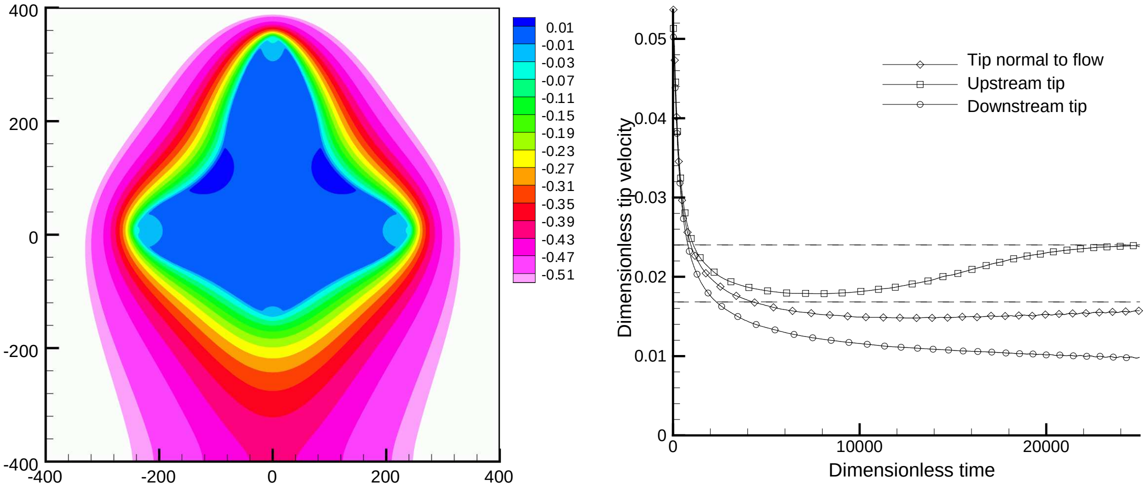
(a) Temperature field

(b) Crystal shape

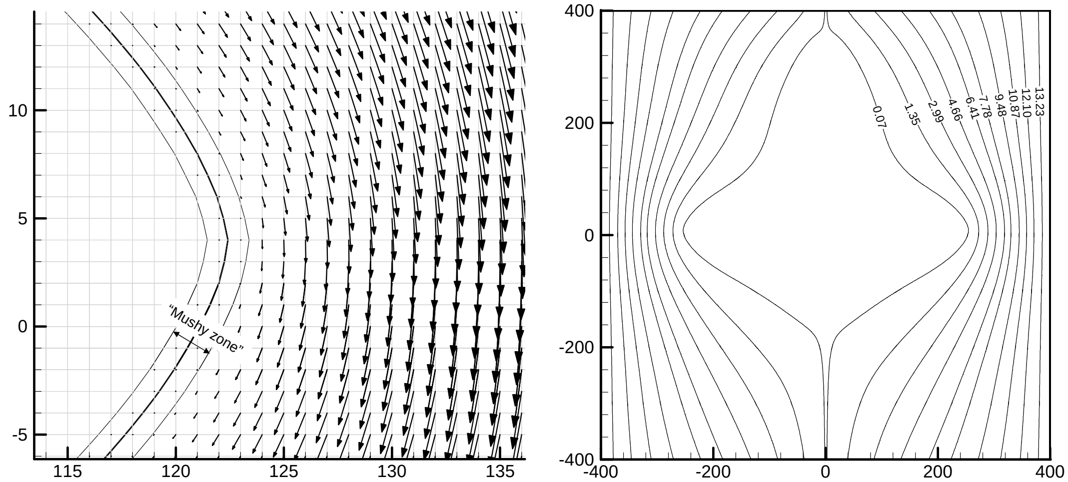
Figure 2.19: Temperature field and crystal shape at time $t = 320$ for 3D crystal growth at an undercooling $\Delta = 0.45$. The predicted 3D crystal growth compares well with that obtained in [28] using the phase field method.

The 2D case studied here is from [6]. The initial temperature is $T_{in} = -0.55$ and the inlet velocity $V_{inlet} = 0.035$ with Pr= 23.1. The remaining conditions are the same as those reported in the 2D solvability theory case.

The computed results are summarized in Fig. 2.20. The dendrite ‘tilting’ shown in Fig. 2.20(a) is due to the fact that the heat fluxes are higher on the upstream side than on the downstream side. The growth patterns predicted here compare fairly well with those in [6]. As shown in Fig. 2.20(a), we only calculated the solution on the right half of the domain $[0, 400]\times[-400, 400]$ using a mesh of 200×400. The left half is mirrored to show the results. Domain decomposition using 8 domains is also used in this calculation. The actual grid in each processor includes two rows of additional grid points in each outer surface to facilitate the communication of data at the boundary. This calculation was performed in the Cornell CTC supercomputer using 4 VII nodes (2×2.4 GHz CPU



(a) 2D Temperature field (b) 2D growth tip velocity



(c) Flow through perpendicular tip (d) Streamline contours

Figure 2.20: Two-dimensional crystal growth with convection.

on each node) within about 5 hours. A schematic of the typical dendrite growth in the presence of fluid flow is shown in Fig. 2.20(a). The dimensionless velocity at the tips is shown in Fig. 2.20(b).

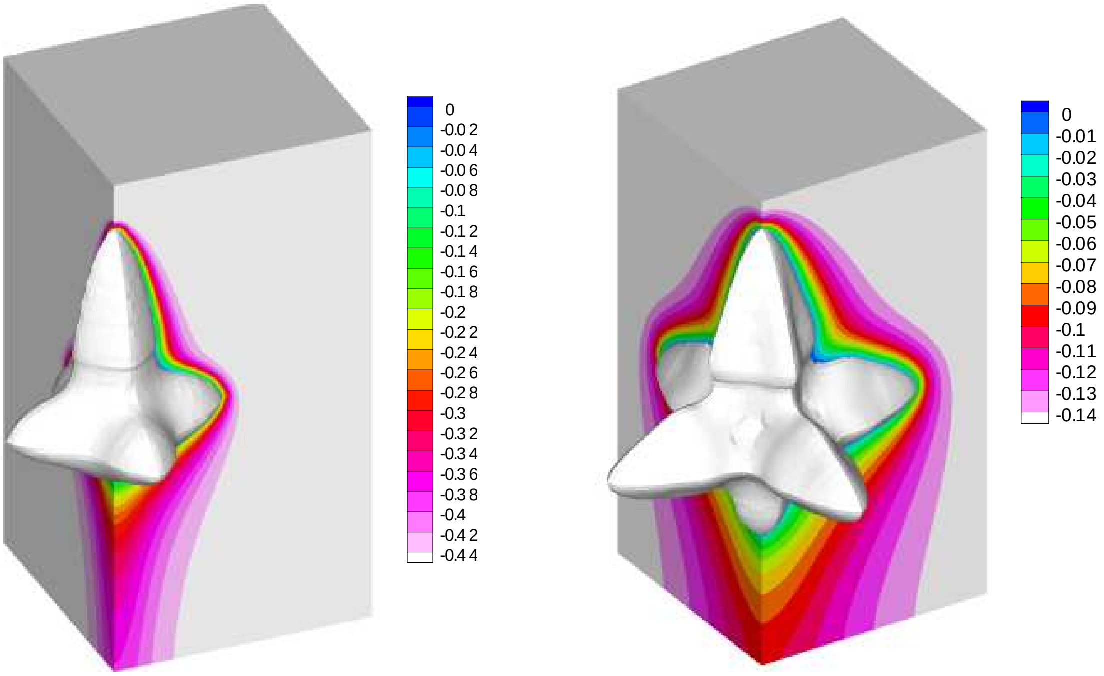
The flow passing through the perpendicular tip is shown in Fig. 2.20(c), while the streamline contours are shown in Fig. 2.20(d). As shown in Fig. 2.20(c), an artificial 'mushy zone' with width $2\Delta x$ is assumed. We use a permeability constant $K_0 = 10^{-5}$

so that the velocity is almost zero within this zone, as if the no-slip condition is applied at the boundary between the mushy-zone and the melt region. Since the half-width of the mushy zone is only Δx, the no-slip condition is applied on the position $\phi = \Delta x$. In our calculation, the mushy zone is identified using $\phi \in [-\Delta x, +\Delta x]$. One can also change position of the artificial mushy zone according to the signed distance variable ϕ, e.g. $\phi \in [-\Delta x, 0]$ or $\phi \in [-2\Delta x, 0]$, so that the no-slip condition can be exactly applied at $\phi = 0$. Comparing with the phase-field method [6] where special modelling of the interfacial stress term was used, it can be seen that the present method does not require an asymptotic analysis. Moreover, since the present method is a whole-domain method using volume-averaging, it is easy to implement and accurate when coupling heat transfer with fluid flow [72]. As it has been pointed out in [72], the volume-averaging model with stabilized FEM formulation converges nearly quadratically. Other methods such as the fractional step method do not show such rates of convergence.

Remark 5: Treating the diffused-interface as a porous medium [6] with the Kozeny-Carman approximation for the permeability is of no physical significance in the present calculations. It is simply a numerical tool for applying the no-slip boundary condition on the growing crystal using a fixed-grid. A variant viscosity in the diffused-interface can be applied as well [66]. In our calculations, we found that both of these methods lead to nearly identical results with velocity of very small magnitude within the mushy zone.

2.6.7 Three-dimensional crystal growth with convection

Two 3D examples are examined with high- and low-undercooling. These examples have been examined earlier in [28] and [3], respectively. For the high-undercooling

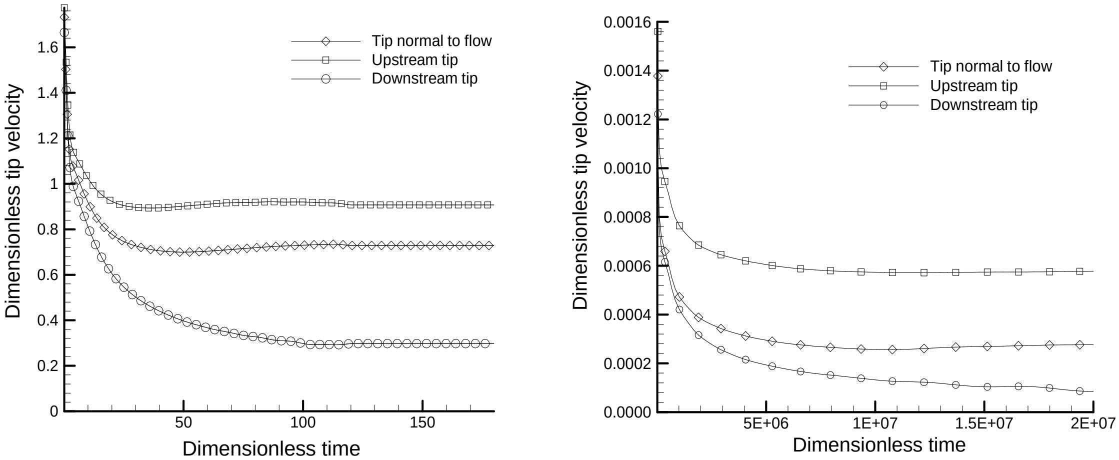


(a) (b)

Figure 2.21: Crystal shape and temperature field for 3D growth in the presence of fluid flow (a) $\Delta = 0.45$, $\epsilon = 0.04$, $V_{inlet} = 1$, $Pr = 23.1$ at time $t = 163$ (b) $\Delta = 0.15$, $\epsilon = 0.3$, $V_{inlet} = 0.002$, $Pr = 1.0$ at time $t = 1.89 \times 10^7$.

case, the undercooling is 0.45 and the inlet velocity is specified as 1.0. The equilibrium temperature on the interface is given from Eq. (2.38) where $d_0 = 1$ and $\epsilon = 0.04$. The computational domain is taken as $[0, 200] \times [0, 200] \times [-200, 200]$ with mesh size $60 \times 60 \times 120$. For the low-undercooling case, the undercooling is 0.15, and the inlet velocity is 0.002. The equilibrium temperature on the interface is specified by Eq. (2.37) where $d_0 = 1$ and $\epsilon = 0.3$. The computational domain is taken as $[0, 12500] \times [0, 12500] \times [-12500, 12500]$ with mesh size $60 \times 60 \times 120$.

The temperature field and crystal shape for both cases are shown in Fig. 2.21. The computed 3D growth pattern is similar to the 2D growth pattern except that there are four perpendicular arms in 3D. Comparing Figs. 2.22(a) and 2.22(b), one can also observe that the crystal growth velocity is greatly affected by the degree of undercooling.



(a) (b)

Figure 2.22: Dendrite tip velocity evolution for 3D crystal growth in the presence of flow (a) $\Delta = 0.45$, $\epsilon = 0.04$, $V_{inlet} = 1$, $Pr = 23.1$ (b) $\Delta = 0.15$, $\epsilon = 0.3$, $V_{inlet} = 0.002$, $Pr = 1.0$.

The dendrite tip velocity differs by an order of about 10^2 for the two undercoolings considered. These results compare very well with those reported in [3, 28].

2.7 Conclusions

A method combining features of front-tracking and fixed domain methods is presented to model dendritic solidification of pure materials. Some of the key features of the presented method include (a) the use of a fixed-grid simulation for heat and momentum transfer, (b) energy conservation by avoiding the explicit application of temperature condition on the freezing front, (c) avoiding the direct application of the no-slip condition on the freezing front, (d) symmetric discretization of the heat equation thus avoiding the problems addressed in [32] and (e) automatic time step selection. The method is substantially simpler to implement relative to front tracking or phase field models.

Chapter 3

Extension to multi-component alloys

3.1 Sharp interface model for the solidification of multi-component alloys

Let us consider the solidification of an alloy system with n components that results in N phases. Each phase α is assumed to occupy a region Ω^α with temperature T^α and concentration C_i^α, for each component i. Each pair of these regions, Ω^α and Ω^β, are separated by the interface $\Gamma^{\alpha\beta}$, which is $\emptyset$ when the two phases are not neighboring each other. The phase domain Ω^α is time-dependent and its boundary Γ^α, which is a union of interfaces with all other phases ($\Gamma^\alpha = \bigcup_{\gamma\neq\alpha} \Gamma^{\alpha\gamma}$), is moving with normal velocity V^α. The unit normal vector $\boldsymbol{n}^\alpha$ is defined as pointing away from the phase region Ω^α. The total domain $\Omega = \bigcup_\alpha \Omega^\alpha$ containing all phases and its external boundary $\partial\Omega$ is assumed to be time-independent. On the interface of two phases $\Gamma^{\alpha\beta}$, which is a part of both Γ^α and Γ^β, the normal growth velocities and unit normal vectors are related as $V^\alpha = -V^\beta$ and $\boldsymbol{n}^\alpha = -\boldsymbol{n}^\beta$, respectively. In this work, we use the subscript i to denote different species with $i = 1$ indicating the major component and $i = 2, 3, \ldots, n$ the minor components. The superscript α is used to denote different phases with $\alpha = 1$ referring to the liquid phase and $\alpha = 2, 3, \ldots, N$ the various solid phases. To emphasize the differences between the solid phases and the liquid phase, we also use s to denote different solid phases with $s = 2, 3, \ldots, N$ and ℓ to denote the liquid phase with $\ell = 1$.

To allow us to concentrate on key aspects of the solidification of multi-component alloy systems, the following assumptions are introduced for the transport of heat, momentum and solute:

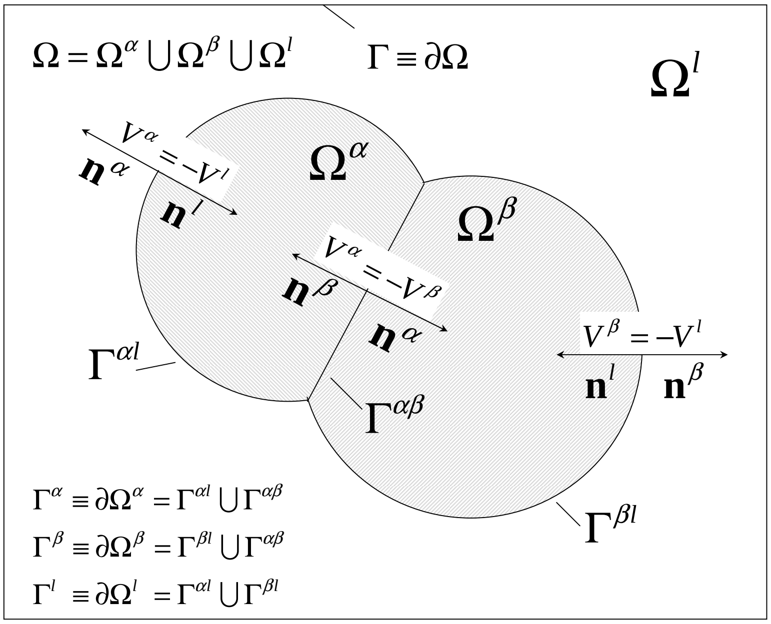


Figure 3.1: Schematic of the solidification of a multi-component alloy system with multiple solid phases (here phases α and β).

1. Constant density ρ, constant thermo-physical and transport properties for each phase including thermal conductivities k^α, heat capacities c^α, latent heats of transformation of the liquid phase ℓ to each solid phase s (denoted as L^s), partition coefficients of the liquid phase ℓ and solid phase s with respect to component i (denoted as $k_{p_i}^s$), solutal diffusion coefficients D_i^ℓ, and viscosity μ^ℓ.
2. Zero solute diffusivity in all solid phases, D_i^s=0, and negligible solid-solid phase transformation, $V^\alpha = 0$ on $\Gamma^{\alpha\beta}$, $\forall\ \alpha,\beta \neq \ell$.

Following standard notation, the governing equations in the presence of fluid flow are given as follows:

$$\nabla \cdot \boldsymbol{v}(\boldsymbol{x}, t) = 0, \quad \boldsymbol{x} \in \Omega^\ell, \tag{3.1}$$

$$\rho\left(\frac{\partial \boldsymbol{v}(\boldsymbol{x}, t)}{\partial t} + \nabla \boldsymbol{v}(\boldsymbol{x}, t)\boldsymbol{v}(\boldsymbol{x}, t)\right) = -\nabla p(\boldsymbol{x}, t)\boldsymbol{I} + \nabla \cdot \mu\left[\nabla \boldsymbol{v}(\boldsymbol{x}, t) + (\nabla \boldsymbol{v}(\boldsymbol{x}, t))^{\mathrm{T}}\right] + \boldsymbol{b}, \quad \boldsymbol{x} \in \Omega^\ell, \tag{3.2}$$

$$\rho c^s \frac{\partial T(\boldsymbol{x},t)}{\partial t} = k^s \nabla^2 T(\boldsymbol{x},t), \;\; \boldsymbol{x} \in \Omega^s, \; s = 2,3,\ldots,N, \tag{3.3}$$

$$\rho c^\ell \left(\frac{\partial T(\boldsymbol{x},t)}{\partial t} + \boldsymbol{v} \cdot \nabla T(\boldsymbol{x},t) \right) = k^\ell \nabla^2 T(\boldsymbol{x},t), \;\; \boldsymbol{x} \in \Omega^\ell, \tag{3.4}$$

$$\frac{\partial C_i^\ell(\boldsymbol{x},t)}{\partial t} + \boldsymbol{v} \cdot \nabla C_i^\ell(\boldsymbol{x},t) = D_i^\ell \nabla^2 C_i^\ell(\boldsymbol{x},t), \;\; \boldsymbol{x} \in \Omega^\ell, \; i = 2,3,\ldots,n, \tag{3.5}$$

where $\boldsymbol{v}$ is the melt flow velocity and $\boldsymbol{b}$ is the buoyancy body force.

The temperature on the interface $\Gamma^{s\ell}$ (denoted as $T_I^{s\ell}$) is equal to the equilibrium temperature T_*^s. This equilibrium temperature is given from the Gibbs-Thomson relation as follows:

$$T_*^s = T^{s\ell}(C_1^\ell, C_2^\ell, \ldots, C_n^\ell) + \varepsilon_c^s \kappa^s + \varepsilon_V^s V^s, \; s = 2,3,\ldots,N, \tag{3.6}$$

where $T^{s\ell}(C_1^\ell, C_2^\ell, \ldots, C_n^\ell)$ is the liquidus temperature obtained from the phase diagram or thermodynamic databases, κ^s is the curvature of the freezing interface, and ε_c^s and ε_V^s are the curvature and kinetic undercooling coefficients, respectively.

With $C_i^s = {k_p}_i^s C_i^\ell$ on $\Gamma^{s\ell}$, solutal rejection at the solid/liquid interfaces is governed by the following equation:

$$D_i^\ell \frac{\partial C_i^\ell}{\partial \boldsymbol{n}^s} = -(1 - {k_p}_i^s) C_i^\ell V^s \boldsymbol{n}^s, \quad \text{on } \Gamma^{s\ell}, \; s = 2,3,\ldots,N. \tag{3.7}$$

The interface velocity V^s on $\Gamma^{s\ell}$ is related to the heat flux jump at the freezing front by the classical Stefan equation (interfacial energy balance) as follows:

$$V^s = -V^\ell = (q^s - q^\ell)/(\rho L^s), \quad \text{on } \Gamma^{s\ell}, \; s = 2,3,\ldots,N, \tag{3.8}$$

where q denotes the heat flux $k \nabla T \cdot \boldsymbol{n}^s$.

The level set function ϕ^α is defined to be the signed distance from the phase boundary Γ^α as shown in Eq. (3.9), Here $d^\alpha(\boldsymbol{x},t)$ is the normal distance of a point $\boldsymbol{x}$ from Γ^α. The phase boundary position is thus implicitly stored in ϕ^α. The idea behind the level set method is to move ϕ^α with speed V^α at phase boundary Γ^α, which is extracted from Eq. (3.8).

$$\phi^\alpha(\boldsymbol{x},t) \equiv \begin{cases} +d^\alpha(\boldsymbol{x},t), & \boldsymbol{x} \notin \Omega^\alpha, \\ 0, & \boldsymbol{x} \in \Gamma^\alpha, \\ -d^\alpha(\boldsymbol{x},t), & \boldsymbol{x} \in \Omega^\alpha, \end{cases} \tag{3.9}$$

The equation of motion governing ϕ^α is given as follows:

$$\phi^\alpha_t + V^\alpha |\nabla \phi^\alpha| = 0, \ \alpha = 1, 2, \ldots, N. \tag{3.10}$$

With the above introduction of the level set function, we will re-write the Stefan condition for the classical Stefan problem as follows:

$$V^s = -V^\ell = (q^s - q^\ell)/(\rho L^s), \ \ q^s = \lim_{\phi^s \to 0^-} q, \quad q^\ell = \lim_{\phi^s \to 0^+} q, \quad \text{on } \Gamma^{s\ell}, \tag{3.11}$$

where the notation 0^+ and 0^- is used here for notational simplification to denote the values of ϕ^s ($s = 2, 3, \ldots, N$) as we approach the freezing front from the solid and liquid sides, respectively.

3.2 Diffused-interfaces and modeling of transport processes

3.2.1 Important assumptions

Many of the difficulties in the implementation of the sharp-interface multi-component alloy solidification problem defined earlier are related to the application of the thermal essential boundary condition given in Eq. (3.6) and the solute rejection flux condition given in Eq. (3.7). To avoid directly applying these conditions as needed in front tracking methodologies, we introduce a diffused-interface approximation to allow convenient modeling of the underlying transport processes while still maintaining an explicit tracking of the phase boundaries. This model was applied earlier for the solidification of pure materials [61]. Volume-averaging approximations used extensively for macroscopic solidification process modeling will be used here for modeling dendritic alloy solidification. Using a diffused-interface model for the transport equations and volume-averaging to derive the corresponding equations is a well established approach for modeling solidification using the phase field method [6].

Assumption 1: We assume that solidification occurs in a diffused zone of width $2w$ that is symmetric around $\phi^{\ell} = 0$. The parameter w used here is defined in terms of the spatial discretization selected (usually taken as the size of one finite element) and it is not related to the physics of phase transformation which usually occurs on a thickness of the order of atomic distances. The choice of w in this model is comparable to the choice for the interface width in the phase field method, and can affect the accuracy of the results, just like in the phase field method.

Following ideas similar to those in diffused-interface models, let us define the function $\Phi^{\alpha}(\boldsymbol{x}, t)$ as follows:

$$\Phi^{\alpha}(\boldsymbol{x}, t) \equiv \begin{cases} 0, & \phi^{\alpha}(\boldsymbol{x}, t) > w, \\ 1, & \phi^{\alpha}(\boldsymbol{x}, t) < -w, \\ -\phi^{\alpha}(\boldsymbol{x}, t)/_{2w} + 0.5, & \phi^{\alpha}(\boldsymbol{x}, t) \in [-w, w]. \end{cases} \tag{3.12}$$

This function was used in our earlier work [61] for the solidification of pure materials to define the liquid volume fraction within the diffused-interface. However, this definition cannot be used for multi-component alloy solidification. Indeed, for $\boldsymbol{x}$ near the phase interface $\Gamma^{\alpha\beta}$ ($\phi^{\alpha} < w, \phi^{\beta} < w$) and far away from triple points ($\phi^{\gamma} > w,\ \forall\, \gamma \neq \alpha, \beta$), we have $\Phi^{\alpha} + \Phi^{\beta} = 1$ and $\Phi^{\gamma} = 0,\ \forall\, \gamma \neq \alpha, \beta$ and Φ can be used as phase volume fraction. However, for $\boldsymbol{x}$ near triple points (interface of three phases) α, β and γ ($\phi^{\alpha} < w, \phi^{\beta} < w, \phi^{\gamma} < w$), we will have $\Phi^{\alpha} + \Phi^{\beta} + \Phi^{\gamma} > 1$ and Φ cannot be used as phase volume fraction. We herein introduce another variable for phase volume fraction as follows:

$$\epsilon^{\alpha} \equiv \Phi^{\alpha} / \sum_{\beta} \Phi^{\beta}. \tag{3.13}$$

Assumption 2: The diffused-interface between a solid phase and the liquid phase is treated as a porous medium with Kozeny-Carman approximation for the permeability:

$$K(\epsilon^{\ell}) = \frac{K_0 \epsilon^{\ell 3}}{(1 - \epsilon^{\ell})^2}, \tag{3.14}$$

where K_0 is a permeability constant taken to be a small value of 10^{-5} in our computations.

The above assumption is of no physical significance in the present calculations. It is simply for the convenience of applying the no-slip boundary condition on the growing solid phases with a fixed-grid. This idea of using a thin region near the interface to apply the no-slip condition is a well accepted technique in the literature. The present Kozeny-Carman approach for enforcing the no-slip condition depends on the selection of w and is essentially equivalent to the one in [6]. In [66], another technique using variant viscosity in the diffused-interface was applied to account for the no-slip condition on the solid-liquid interface. In our previous work [61], we had shown that the variant viscosity and the Kozeny-Carman approach lead to nearly identical results with velocity of very small magnitude within the thin artificially-constructed diffused solid-liquid interfaces.

Assumption 3: The solid-liquid interface temperature $T_I^{s\ell}$ is allowed to vary from the equilibrium temperature T_*^s in such a way that

$$\frac{dT_I^{s\ell}}{dt} = -k_N (T_I^{s\ell} - T_*^s), \tag{3.15}$$

where k_N controls the rate with which $T_I^{s\ell}$ is designed to approach the desired equilibrium temperature.

With the assumption of Eq. (3.15), and by further assuming that the mean temperature in the diffused-interface (mushy) zone can be approximated as $T_I^{s\ell}$, the energy conservation for the diffused freezing zone leads to what we here refer to as *the extended Stefan condition* [61]:

$$V^s = \frac{q^s - q^\ell}{\rho L^s} + \frac{2\tilde{c}^s w}{L^s} k_N (T_*^s - T_I^{s\ell}), \; q^s = \lim_{\phi^s \to -w^-} q, q^\ell = \lim_{\phi^s \to w^+} q, \text{ on } \Gamma^{s\ell}, \tag{3.16}$$

where $\tilde{c}^s \equiv 0.5(c^s + c^\ell)$ and the heat fluxes q^ℓ and q^s are computed at the boundaries of the diffused-interface. According to the stability analysis discussed in [61], k_N is selected based on the time step size as $k_N = 1/\Delta t$.

The extended Stefan equation is used to compute the interface velocity and evolve the interface. It weakly enforces the Gibbs-Thomson relation on the moving interface. The other two important conditions on the interface (the one governing solute rejection and the no slip condition for fluid flow) are also weakly enforced using the volume averaging techniques that are reviewed next.

3.2.2 Brief review of the volume-averaging transport equations

In order to set forward the notation and fundamental equations used for modeling the transport phenomena relevant to dendritic growth, in the section we will briefly summarize the volume-averaged transport equations applied to the diffused interface initially developed in [6] in the context of phase field solidification models. A more detailed presentation is given in [57, 59].

The equations representing momentum, energy, mass and species transport in each phase α take the following general form:

$$\frac{\partial \Theta^\alpha}{\partial t} + \nabla \cdot (\Theta^\alpha \, \boldsymbol{v}^\alpha) = \nabla \cdot \mathbf{J}^\alpha + S^\alpha, \tag{3.17}$$

for appropriate selection of the field variable Θ, diffusion flux $\mathbf{J}$ and source term S.

A phase function ν^{α} is introduced as taking value 1 in phase α and zero elsewhere. The volume-averaged quantity $< \Psi^{\alpha} >$ of any quantity $\Psi(\boldsymbol{x}, t)$ in phase α over the entire averaging volume dV can now be introduced as:

$$< \Psi^{\alpha} >= \frac{1}{dV} \int_{dV} \Psi^{\alpha} \nu^{\alpha}(\boldsymbol{x}, t) dv \tag{3.18}$$

Similarly, one can introduce the intrinsic volume-averaged quantity $< \Psi^{\alpha} >^{\alpha}$ (averaged value of $\Psi(\boldsymbol{x}, t)$ in the control volume dV^{α}) as:

$$< \Psi^{\alpha} >^{\alpha}= \frac{1}{dV^{\alpha}} \int_{dV} \Psi^{\alpha} \nu^{\alpha}(\boldsymbol{x}, t) dv = \frac{< \Psi^{\alpha} >}{\epsilon^{\alpha}}, \tag{3.19}$$

where ϵ^{α} was defined earlier in Eq. (3.13).

Finally, the fluctuating component $\hat{\Psi}^{\alpha}$ is commonly introduced to represent the deviation of Ψ^{α} from the intrinsic volume-averaged $< \Psi^{\alpha} >^{\alpha}$. It is given by:

$$\hat{\Psi}^{\alpha} = (\Psi^{\alpha} - < \Psi^{\alpha} >^{\alpha}) \nu^{\alpha}. \tag{3.20}$$

Multiplying each side of Eq. (3.17) by ν^{α}, integrating it over the averaging volume dV, one can obtain the following averaged transport equation for phase α [57]:

$$\begin{aligned} \frac{\partial < \Theta^{\alpha} >}{\partial t} + \nabla \cdot \epsilon^{\alpha} < \Theta^{\alpha} >^{\alpha} < \boldsymbol{v}^{\alpha} >^{\alpha} &= \nabla \cdot < \mathbf{J}^{\alpha} > + < S^{\alpha} > \\ &+ \nabla \cdot \frac{1}{dV} \int_{dV} (-\widehat{\Theta^{\alpha}} \widehat{\boldsymbol{v}^{\alpha}}) dv + \frac{1}{dV} \int_{dA^{\alpha}} \mathbf{J}^{\alpha} \cdot \boldsymbol{n}^{\alpha} dA \\ &+ \frac{1}{dV} \int_{dA^{\alpha}} \Theta^{\alpha} (\boldsymbol{V}^{\alpha} - \boldsymbol{v}^{\alpha}) \cdot \boldsymbol{n}^{\alpha} dA, \end{aligned} \tag{3.21}$$

where dA^{α} is the interfacial area of phase α with the other phases, $\boldsymbol{n}^{\alpha}$ is the outward unit normal of the infinitesimal element of area dA of phase α, and $\boldsymbol{V}^{a}$ is the growth velocity of the boundary of phase α.

In comparison with the original transport Eq. (3.17), three extra terms I_D^{α}, I_J^{α} and I_Q^{α}

appear from the volume-averaging procedure of the form:

$$I_D^\alpha \equiv \nabla \cdot \frac{1}{dV} \int_{dV} (-\widehat{\Theta^\alpha} \widehat{\boldsymbol{v}^\alpha}) dv, \tag{3.22}$$

$$I_Q^\alpha \equiv \frac{1}{dV} \int_{dA^\alpha} \Theta^\alpha (\boldsymbol{V}^\alpha - \boldsymbol{v}^\alpha) \cdot \boldsymbol{n}^\alpha dA, \tag{3.23}$$

$$I_J^\alpha \equiv \frac{1}{dV} \int_{dA^\alpha} \mathbf{J}^\alpha \cdot \boldsymbol{n}^\alpha dA. \tag{3.24}$$

Almost all models reported in the literature neglect the I_D^α term [25, 47, 7]. The same approximation is considered here as well. The term I_Q^α accounts for the interfacial transfer due to phase change, whereas I_J^α represents the transport phenomena between phases within dV by diffusion and is related to the gradients of microscopic velocity, temperature and species concentration on each side of the solid/liquid interface dA^α [42]. Since in the volume-averaging approach that we will follow, the averaged equations from different phases are added within the averaging volume dV, detailed modeling of the interfacial transfer terms I_J^α and I_Q^α can be avoided [57]. The heat, mass or species lost from one phase is gained by other phases, i.e.

$$\sum_\alpha I_J^\alpha = 0 \qquad \text{and} \qquad \sum_\alpha I_Q^\alpha = 0. \tag{3.25}$$

For momentum, the interfacial momentum fluxes due to solidification also balance each other, that is $\sum_\alpha I_Q^\alpha = 0$. However, $\sum_\alpha I_J^\alpha$ in the diffused-interface (mushy) zone is modeled using Darcy's law [72]:

$$\sum_\alpha I_J^\alpha = -\frac{\epsilon^{\ell 2} \mu^\ell < \boldsymbol{v}^\ell >^\ell}{K(\epsilon^\ell)}, \tag{3.26}$$

where $K(\epsilon^\ell)$ is the assumed permeability of the diffused-interface (see Assumption 2).

To arrive at a model tractable for computation, the variations of material properties in dV^α are neglected, although globally they may vary, that is $< \mu^\alpha >^\alpha = \mu^\alpha, \quad < k^\alpha >^\alpha = k^\alpha, \quad < D^\ell >^\ell = D^\ell, \quad D^s = 0$. We also assume that all phases in the averaging volume are in thermodynamic equilibrium, i.e. $< T^\alpha >^\alpha = < T^\beta >^\beta$ for any two phases α and β.

The concentration in the averaging volume is assumed to be solutally well mixed, that is, $< C_i^\alpha >^\alpha = C_i^\alpha$.

Mass conservation

For the derivation of the volume-averaged equation of mass conservation using Eq. (3.21), we substitute $\Theta = \rho$, $\mathbf{J} = 0$, and $S = 0$. By writing and adding the individual mass conservation equations for all phases, we obtain:

$$\nabla \cdot \boldsymbol{v} = 0, \tag{3.27}$$

where we have further defined:

$$\boldsymbol{v} = \epsilon^\ell < \boldsymbol{v}^\ell >^\ell . \tag{3.28}$$

Note that in this work, we have assumed that $< \boldsymbol{v}^s >^s = 0$ for all solid phases $s = 2, 3, \ldots, N$.

Momentum conservation

For deriving the averaged equation of momentum conservation from Eq. (3.21), we take $\Theta = \rho \boldsymbol{v}$ and $S = \mathbf{b}$. Furthermore, we assume a Newtonian fluid and hence the viscous-stress is given in terms of the rate of deformation as,

$$\boldsymbol{\sigma} = -p^\ell I + \mu^\ell [\nabla \boldsymbol{v}^\ell + (\nabla \boldsymbol{v}^\ell)^T]. \tag{3.29}$$

Using the previous assumptions, and the definition of $\boldsymbol{v}$ given earlier, the final form of the averaged transport equation of momentum transport then yields [72]:

$$\begin{aligned} \frac{\partial(\rho\, \boldsymbol{v})}{\partial t} + \nabla \cdot (\rho \frac{\boldsymbol{v}\boldsymbol{v}}{\epsilon^\ell}) = & - \epsilon^\ell \nabla < p^\ell >^\ell + \nabla \cdot \left[\mu^\ell \left(\nabla \boldsymbol{v} + (\nabla \boldsymbol{v})^T\right)\right] \\ & - \epsilon^\ell \mu^\ell \frac{\boldsymbol{v}}{K(\epsilon^\ell)} + \epsilon^\ell \rho^\ell \mathbf{g}, \end{aligned} \tag{3.30}$$

where $\mathbf{g}$ is the gravity vector. The change in liquid density is here expressed using the Boussinesq approximation $\rho^\ell = \rho[(1 - \beta_c(C - C_0) - \beta_T(T - T_0)]$ and it appears only in the body force term. In other places, ρ^ℓ is regarded to be the same constant as ρ.

Energy conservation

For deriving the volume-averaged equation of energy conservation we take $\Theta = \rho h = \rho \sum_\alpha \epsilon^\alpha h^\alpha$, where h represents the total enthalpy. In addition, we consider in Eq. (3.21), $S = 0$ and utilize Fourier's law $\mathbf{J} = -k\nabla T$, with $k \equiv \sum_\alpha \epsilon^\alpha k^\alpha$. Eq. (3.21) then yields the following:

$$\frac{\partial(\rho h)}{\partial t} + \nabla \cdot (\rho h^\ell \boldsymbol{v}) = \nabla \cdot (k \nabla T). \tag{3.31}$$

We assume constant but unequal specific heat for all phases. Using linear relation between enthalpy and temperature for each phase, a governing equation for energy conservation based on temperature can be derived from Eq. (3.31) in the following form:

$$(\sum_{\alpha=1}^{N} \rho c^\alpha \epsilon^\alpha)\frac{\partial T}{\partial t} + \rho c^\ell \nabla \cdot (\boldsymbol{v} T) \;=\; \nabla \cdot (k\nabla T) + \sum_{s=2}^{N} \rho L^s \dot{\epsilon}^s. \tag{3.32}$$

Species conservation

For arriving at the volume averaged equation of species conservation, we note that for this case in Eq. (3.21), $\Theta = \rho C_i$, where C_i represents solutal concentration (per unit mass) for component i, and $S = 0$. Furthermore, we utilize Fick's law for species diffusion flux, that is, $\mathbf{J} = -\rho D_i \nabla C_i$. We also assume no diffusion in the solid phase $D_i^s = 0$. The macroscopic transport equation of species conservation can then be derived from Eq. (3.21) as follows:

$$\frac{\partial C_i}{\partial t} + \nabla \cdot (C_i^\ell \boldsymbol{v}) = \nabla \cdot (\epsilon^\ell D_i^\ell \nabla C_i^\ell), \tag{3.33}$$

where

$$C_i = \sum_\alpha \epsilon^\alpha C_i^\alpha. \tag{3.34}$$

Notice that C_i^ℓ is only defined in the region with liquid fraction greater than zero (the liquid phase and diffused interface). Particularly in the diffused-interface, the following holds $C_i^\ell = C_i^s / k_{p_i}^{\,s}$, $\forall\ s = 2, 3, \ldots, N$. Any definition of C_i^ℓ in the solid phase will not affect the validity of Eq. (3.33), since the two terms in Eq. (3.33) related with C_i^ℓ are multiplied with either $\boldsymbol{v}$ (which is zero in the solid) or ϵ^ℓ (also zero in the solid). However, for numerical reasons, C_i^ℓ is usually taken to be a continuous function across the mushy zone. For example, C_i^ℓ is taken as the eutectic concentration for the solid phase in [72]. In this work, we follow the idea of [45] by taking $C_i^\ell = C_i^s / k_{p_i}^{\ s}$ in the solid phase. The *activity* scalar field is defined following [45] as:

$$a_i(\boldsymbol{x}, t) \equiv \begin{cases} C_i^\ell(\boldsymbol{x}, t) & \boldsymbol{x} \in \Omega_t^\ell, \\ C_i^s(\boldsymbol{x}, t) \Big/ k_{p_i}^{\ s} & \boldsymbol{x} \in \Omega_t^s, \end{cases} \tag{3.35}$$

where $a_i(\boldsymbol{x}, t)$ is continuous in the whole domain Ω. In particular within the diffused interface, we will have $C_i^\ell = C_i^s / k_{p_i}^{\ s} = a_i$. So Eq. (3.33) can be rewritten as

$$\frac{\partial C_i}{\partial t} + \nabla \cdot (\boldsymbol{v} a_i) = \nabla \cdot (\epsilon^\ell D_i^\ell \nabla a_i), \ \ \boldsymbol{x} \in \Omega^\ell, \ i = 2, 3, \ldots, n, \tag{3.36}$$

where using the definition of mixture concentration, $C_i = \epsilon^\ell C_i^\ell + \sum_s \epsilon^s C_i^s$ and Eq. (3.35), the relation between concentration and activity is written as follows:

$$a_i = \frac{C_i}{\epsilon^\ell + \sum_s \epsilon^s k_{p_i}^{\ s}}. \tag{3.37}$$

3.3 Numerical implementation

3.3.1 Multi-phase motion re-initialization technique

For a solidification system with only one solid phase, only one signed distance function is required since $\phi^s = -\phi^\ell$ everywhere. Correspondingly, only one level set equation (usually ϕ^s) needs to be solved. In this work, we consider the possibility of multiple solid phases with $s = 2, 3, \ldots, N$. A signed distance function ϕ^α is defined for each phase α. The velocity on the solid-liquid interface is computed using Eq. (3.16). The velocity on the solid-solid interface is zero, since solid-solid phase transformation is neglected in this work. With interface velocity defined on the boundary of each phase, a set of level set equations (Eq. 3.10) can be solved to evolve the phase boundary for each phase. However, evolving each level set equation independently leads to gaps or overlaps of zero level sets due to numerical error. This is a difficult problem addressed in many earlier publications [77, 39]. In this work, we use a reinitialization scheme based on the following property of signed distances for a multi-phase system.

Property: Each point $\boldsymbol{x}$ in the domain Ω belongs to a phase α and has a closest phase β such that $\phi^{\alpha}(\boldsymbol{x}) = -\phi^{\beta}(\boldsymbol{x})$ and $\phi^{\alpha}(\boldsymbol{x}) \leq 0 \leq \phi^{\beta}(\boldsymbol{x}) \leq \phi^{\gamma}(\boldsymbol{x}),\ \forall\, \gamma,\ \gamma \neq \alpha, \beta$.

In this work, we use a rather simple but efficient re-initialization scheme with the following three steps: 1. For each node point $\boldsymbol{x}$, find the smallest two signed distance functions $\phi^{\alpha}(\boldsymbol{x}), \phi^{\beta}(\boldsymbol{x})$ such that $\phi^{\alpha}(\boldsymbol{x}) \leq \phi^{\beta}(\boldsymbol{x}) \leq \phi^{\gamma}(\boldsymbol{x}),\ \forall\, \gamma \neq \alpha, \beta$. α will be the phase containing node point $\boldsymbol{x}$ and β will be the nearest phase to $\boldsymbol{x}$. 2. Compute $err = \frac{\phi^{\alpha}(\boldsymbol{X})+\phi^{\beta}(\boldsymbol{X})}{2}$. For all γ (including α, β), update $\phi^{\gamma}(\boldsymbol{x})$ as $\phi^{\gamma}(\boldsymbol{x}) - err$. 3. Use a fast marching technique to independently re-initialize each signed distance function.

This numerical scheme is very fast since the operation time for the three steps is only $O(m)$, $O(m)$ and $O(m \log m)$ respectively, where m represents the number of nodes in the finite element mesh.

3.3.2 Fast marching technique

Reinitialization techniques based on structured meshes are described with detail in [56, 43]. In an effort to make our developments suitable for coupling with FEM-based solvers for heat transfer, fluid flow and solute transport, we extend the fast marching method to unstructured grids and use it to implement the reinitialization procedure and extension of interface velocity. For convenience, we will only emphasize extension to the positive region $\phi > 0$. Extension of fields to the negative region $\phi < 0$ follows similar methodology.

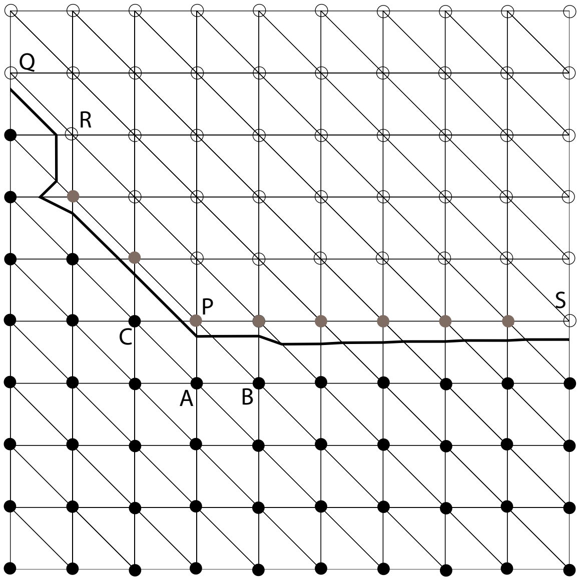


Figure 3.2: Fast marching scheme for an unstructured finite element grid in two-dimensions.

To simplify the presentation, only two-dimensional problems are addressed here. The fast marching method uses tags to indicate the type of each node: ‘alive’, ‘band’ and ‘far away’, which are shown as black, gray and empty circles, respectively, in Fig. 3.2. ‘Alive’ means that the value of a variable (e.g. of the level set variable ϕ or the interface velocity V) is already computed on the node and ready to be extended to other nodes. A node adjacent to ‘alive’ nodes is ready to have its value extended and is marked as ‘band’. Other nodes are simply marked as ‘far away’.

A very important idea in the fast marching method is that the level set function values should be extended from the nodes closest to the interface to the nodes farthest from the interface in a well-defined order. In each iteration, the ‘band’ node with smallest ϕ will have its value extended. Note that when extending ϕ itself, the ϕ value on a 'band' node is just an estimate which is updated whenever one of its neighbor nodes becomes 'alive'. After the closest 'band' node becomes 'alive', its 'far away' neighbors may be captured into 'band'. One continues with this process until all 'band' nodes become 'alive'.

How to find the closest 'band' node as fast as possible for extension of value is very crucial. A simple way is to check all of the nodal points, which leads to a method of operation time $O(m)$ for each iteration with m the number of nodes. The fast marching method uses a balanced heap data structure to store all the 'band' nodes with the nearest node always at root. The balanced heap data structure will lead to a method of operation time only $O(log(m_b))$ for each iter with m_b the number of 'band' nodes. Details of implementation and algorithms of heap data structures are given in [56, 43].

Upwind element - When we estimate value at a 'band' node P, there can be more than one finite element, which has all nodes 'alive' except P (e.g. element (A, B, P) and (B, C, P) shown in Fig. 3.2). We define as *upwind element* of a 'band' node the element with the smallest average ϕ over all its 'alive' nodes. The upwind element is used to extend value to the 'band' node. For example, if $\phi_A + \phi_B < \phi_A + \phi_C$, then element (A, B, P) will be selected as upwind element to extend value to node P.

Extend values in the fast marching method - Application of the fast marching method involves estimating the level set variable for a node from its 'alive' neighbors. Previous work [43] describes how to apply this extrapolation for structured grids. For unstructured grids, the problem boils down to estimating ϕ_P on node P from the value at 'alive' nodes of its upwind element (e.g. ϕ_A on node A and ϕ_B on node B for the case shown in Fig. 3.2). In the following, we will derive the formulation of extending values for the two-dimensional case. Suppose, the other two 'alive' nodes of 'band' node P's upwind element are A, B. The normal direction of ϕ, $\nabla\phi = (\phi_{,1}, \phi_{,2})^T$, in this upwind element will satisfy:

$$\begin{pmatrix} \Delta x_A & \Delta y_A \\ \Delta x_B & \Delta y_B \end{pmatrix} \begin{pmatrix} \phi_{,1} \\ \phi_{,2} \end{pmatrix} = \begin{pmatrix} \Delta\phi_A \\ \Delta\phi_B \end{pmatrix},$$

where $\Delta\phi_A \equiv \phi_A - \phi_P$, $\Delta x_A \equiv x_A - x_P$ (similarly for other variables). For convenience, we redefine ϕ_P, ϕ_A, ϕ_B as ϕ, ϕ_1, ϕ_2 and define matrix M as

$$M \equiv \begin{pmatrix} \Delta x_A & \Delta y_A \\ \Delta x_B & \Delta y_B \end{pmatrix}.$$

In order to calculate ϕ (ϕ_P), we take

$$\begin{pmatrix} \phi_{,1} \\ \phi_{,2} \end{pmatrix} = M^{-1} \begin{pmatrix} \phi_1 - \phi \\ \phi_2 - \phi \end{pmatrix},$$

with each component $\phi_{,i} = \sum_j (M^{-1})_{ij}(\phi_j - \phi) = -\sum_j (M^{-1})_{ij}\phi + \sum_j (M^{-1})_{ij}\phi_j$. According to the property of signed distance $||\nabla\phi|| = 1$, we can write:

$$\sum_i (\sum_j (M^{-1})_{ij}\phi - \sum_j (M^{-1})_{ij}\phi_j)^2 = 1. \tag{3.38}$$

With the geometry-related matrix M and ϕ_j known, Eq. (3.38) becomes a quadratic equation for one single variable ϕ. Generally, Eq. (3.38) has two roots. If fast marching is used in the positive region ($\phi > 0$), then the larger root is used. Otherwise, the smaller root is used. Numerically, if the ϕ value near the interface deviates significantly from signed distance, the above quadratic equation may not attain any real root. Although this did not occur in any of the simulations reported in this paper, for such cases ϕ is taken to be $\frac{\phi_1+\phi_2}{2}$ to make the algorithm robust under any circumstances.

For extension of any other variable (denote as Θ here, e.g. of the interface velocity), we can use the following equation

$$\nabla\Theta \cdot \nabla\phi = 0. \tag{3.39}$$

Similarly to extending the level set variable, we can write the following:

$$\begin{pmatrix} \Theta_{,1} \\ \Theta_{,2} \end{pmatrix} = M^{-1} \begin{pmatrix} \Theta_1 - \Theta \\ \Theta_2 - \Theta \end{pmatrix}.$$

By defining the vector $b_{\Delta\Theta} = (\Theta_1 - \Theta, \Theta_2 - \Theta)^T$ and $b_{\Delta\phi} = (\phi_1 - \phi, \phi_2 - \phi)^T$, and using Eq. (3.39), we obtain

$$(M^{-1} b_{\Delta\phi}, M^{-1} b_{\Delta\Theta}) = 0 \Rightarrow b_{\Delta\phi}^T M^{-T} M^{-1} b_{\Delta\Theta} = 0.$$

Further by defining $s^T \equiv b_{\Delta\phi}^T M^{-T} M^{-1} = (s_1, s_2)$, the variable Θ at node P can be derived as

$$\Theta = \frac{s_1 \Theta_1 + s_2 \Theta_2}{s_1 + s_2}.$$

Notice that in Fig. 3.2, the nodes Q, R and S are not 'band' nodes (thus are not stored in heap structure) even if they are neighboring one 'alive' node. This is because an upwind element with other two nodes 'alive' is required to extend values in our algorithm. Nodes Q, R and S are not captured into 'band' until there is at least one neighboring element with two other nodes (in 2D) 'alive'.

Following the same procedure, the above fast marching techniques including reinitialization and interface velocity extension can easily be extended to three-dimensional problems using a tetrahedral mesh. For example, for extending values (level set variable or interface velocity) from A, B, C to P, the matrix M becomes,

$$M \equiv \begin{pmatrix} \Delta x_A & \Delta y_A & \Delta z_A \\ \Delta x_B & \Delta y_B & \Delta z_B \\ \Delta x_C & \Delta y_C & \Delta z_C \end{pmatrix},$$

with $\Delta x_A, \Delta y_A, \ldots$ defined as before. The form of single variable quadratic equation (Eq. (3.38)) remains the same, except that the summation is done in three dimensions. The formula for extension of velocity becomes

$$\Theta = \frac{s_1\Theta_1 + s_2\Theta_2 + s_3\Theta_3}{s_1 + s_2 + s_3}.$$

3.3.3 Adaptive meshing technique

Mesh refinement procedure in two-dimensions

A uniform mesh is computationally inefficient since phase boundaries often require finer mesh density. For a given time step, a non-uniform mesh with finer mesh density near phase boundaries can greatly speed up the computation. However, the fact that phase boundaries are moving with time, requires fast adaptive meshing for each time step. In [49], a refinement procedure based on structured grids is used. Each box element, which needs refinement, is divided into a few smaller elements (4 elements in 2D and 8 elements in 3D). The adapted grid will have so-called 'hanging' nodes which appear whenever an element has a neighbor whose refinement level differs by one. Notice that 'hanging' nodes make the grid not conforming, thus not very convenient for finite element computation. To ensure continuity of the solution between elements, constraints are enforced for solution at each 'hanging' node [49]. In this work, we implemented a technique based on unstructured grids (triangles in 2D and tetrahedrons in 3D), which is more convenient for meshing with irregular geometry. A scheme with little computational cost (proportional to the number of 'hanging' nodes) is developed to make the grid conforming by linking the 'hanging' nodes in a compatible way.

At first, an initial coarse triangle mesh needs to be generated using any suitable mesh generator. If refinement is desired in a region, a triangle element may be subdivided into four small triangles with the same size and shape by adding middle points of the three edges. If the neighboring element is already refined then middle point of the neighboring edge does not have to be inserted since it is already there.

Information about the level of refinement is stored in a tree data structure, which is capable of tracking sons (refined elements) or ancestors (unrefined element) for each element. The refinement level of an element is defined according to its position in the tree data structure. Elements at the root (in the initial coarse mesh) have refinement level 0. Elements which are generated by refining elements with refinement level n are assigned refinement level $n+1$. Refinement level of the mesh is defined as the maximum refinement level of elements in the tree data structure. A typical sequence of refinement is shown in Fig. 3.3, along with the associated data structure.

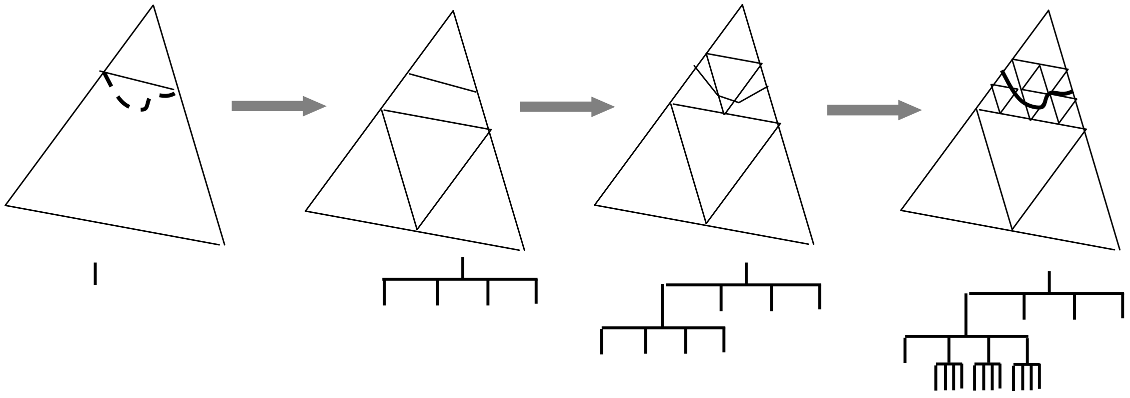


Figure 3.3: Schematic of the refinement procedure for unstructured grids in two-dimensions. The broken line refers to the interface to be represented. Continuous line demonstrates the linear representation of the interface using the adaptive mesh.

In general, the mesh will not be suitable for finite element analysis after refinement, since middle points may exist on element edges as shown in the refined mesh in Fig. 3.3. In order to make the mesh conforming, two more steps are taken as follows:

1. Elements are further refined so that the refinement level for each two neighboring elements differs by at most 1 as shown in step (1) of Fig. 3.4.
2. An element neighboring another element with higher refinement level is subdivided into a few connecting elements by connecting its nodes and middle points on edges as shown in step (2) of Fig. 3.4.

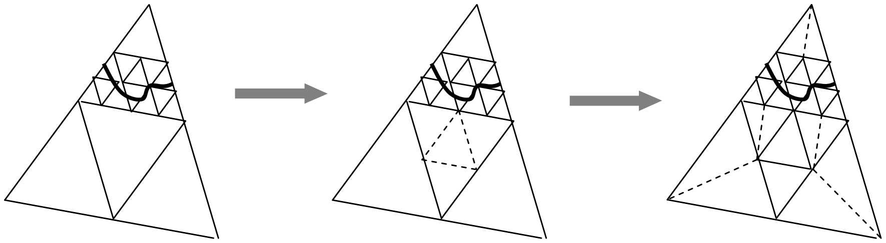


Figure 3.4: Schematic of generating a conforming grid in two-dimensions.

Using this adaptive meshing technique, the quality of the triangles (note that elements with all angles equal to 60^o are considered to be of the best quality) will be exactly the same as the input mesh since the refinement does not change the element shape. This, however, is not the case for the connecting elements (elements generated by linking the hanging nodes). Moreover, linear interpolation of data only needs to be done for newly inserted middle points by just taking the average of data on end points. These advantages make the present adaptive meshing technique very suitable for transient problems.

Mesh refinement procedure in three-dimensions

The basic mesh refinement procedure in three-dimensions is similar to that used earlier in two-dimensions. However, there are a few additional challenges.

Each tetrahedral element is divided into eight child elements. The shape of child elements is not preserved as shown in Fig. 3.5. Moreover, there are three options to divide a tetrahedron into eight smaller tetrahedrons. We compare the length of the line segments $4-9$, $5-8$, $6-7$ (Fig. 3.5), and choose the option corresponding to the shortest length. This ensures that the sum of edge lengths is minimized in refinement and consequently, the mesh quality is maintained.

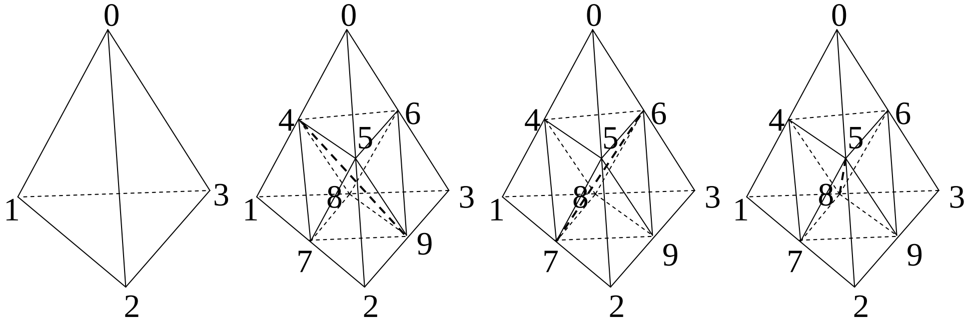


Figure 3.5: Three options of refining a tetrahedral element in 3D.

As in two-dimensions, linking elements with ‘hanging’ nodes on some of the middle of their edges are generated after the first procedure of refinement. In order to make the mesh conforming, it is necessary to link the ‘hanging’ nodes in a compatible way such that (1) each linking element is divided into a few smaller ones without ‘hanging’ nodes as shown in Fig. 3.6, (2) the common faces of neighboring elements are divided in the same way to avoid the error shown in Fig. 3.7.

Mesh refinement criterion

Different criterions can be used to determine whether an element needs to be further refined or not. For example, in [28] a refinement criterion was used based on the temperature gradient within the element e as follows:

$$Error(e) \equiv \int_{\Omega^e} \|\nabla T\| \, d\Omega < tol_T. \tag{3.40}$$

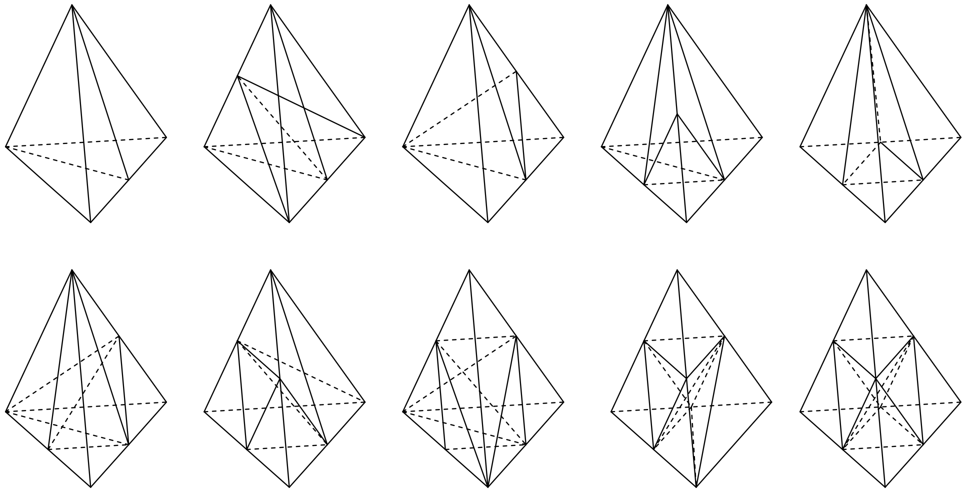


Figure 3.6: Various types of linking element in 3D.

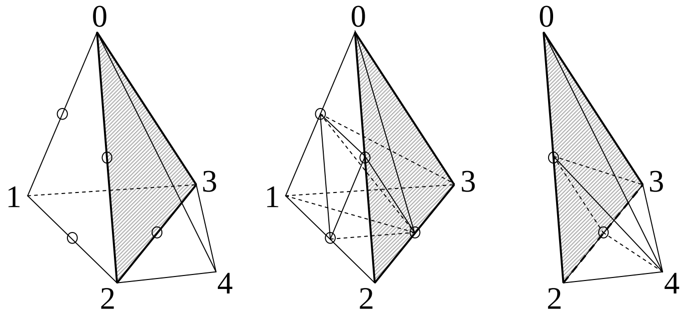


Figure 3.7: Incompatible neighboring elements in three dimensions. Although both neighboring elements ($0-1-2-3$ and $0-2-3-4$) are correctly divided into a few smaller elements by linking the 'hanging' nodes, the common face of $0-2-3$ is divided in different ways for these two neighboring elements.

Similarly, a criterion based on solute gradient can also be used as follows:

$$Error(e) \equiv \int_{\Omega^e} \|\nabla C\| \, d\Omega < tol_C. \tag{3.41}$$

In this work, since we can determine how far away a node point is from an interface based on the computed signed distances, we select a refinement criterion according the signed distances as follows:

$$h^e < \max(\frac{|\phi|}{4}, w), \tag{3.42}$$

where h^e is a characteristic length of the element computed as $\sqrt{area(e)}$, $|\phi|$ is the minimum distance to the interface of the element computed from the value of signed distances on each element node. Notice that the value of w is preselected as a spatial discretization parameter. Based on this value of w and level set field ϕ, the mesh is refined with the refinement criterion to be sure that (1) fine mesh is obtained when near the interface and coarse mesh is sustained when far away from the interface, and (2) at least one or a few elements are contained within the artificial diffused-interface (mushy) zone. w serves as a lower bound for h^e in Eq. (3.42), since generally two elements in the artificial mushy zone are enough. If more elements (e.g. 4 elements) are desired to numerically resolve the artificial mushy zone, w in Eq. (3.42) needs to be replaced with a smaller value (e.g. $\frac{w}{2}$, so that the criterion in Eq. (3.42) becomes $h^e < \max(\frac{|\phi|}{4}, \frac{w}{2})$).

Narrow band computation

Adaptive meshing can drastically reduce the number of degrees of freedom, especially for computation of heat, momentum and solute transport. However, for the level set evolution, adaptive mesh computation on the whole domain is not very efficient. Level set computation can be performed on a narrow band near the interface, instead of on the whole domain [56, 43]. An example of narrow band computing mesh (for level set only) is shown in Fig. 3.8.

3.3.4 Adaptive time step technique

For many dendritic solidification growth problems, initial conditions are given before the formation of thermal and solutal boundary layers. The initial interface velocity is large in comparison to the interface velocity after the build up of thermal and solutal boundary

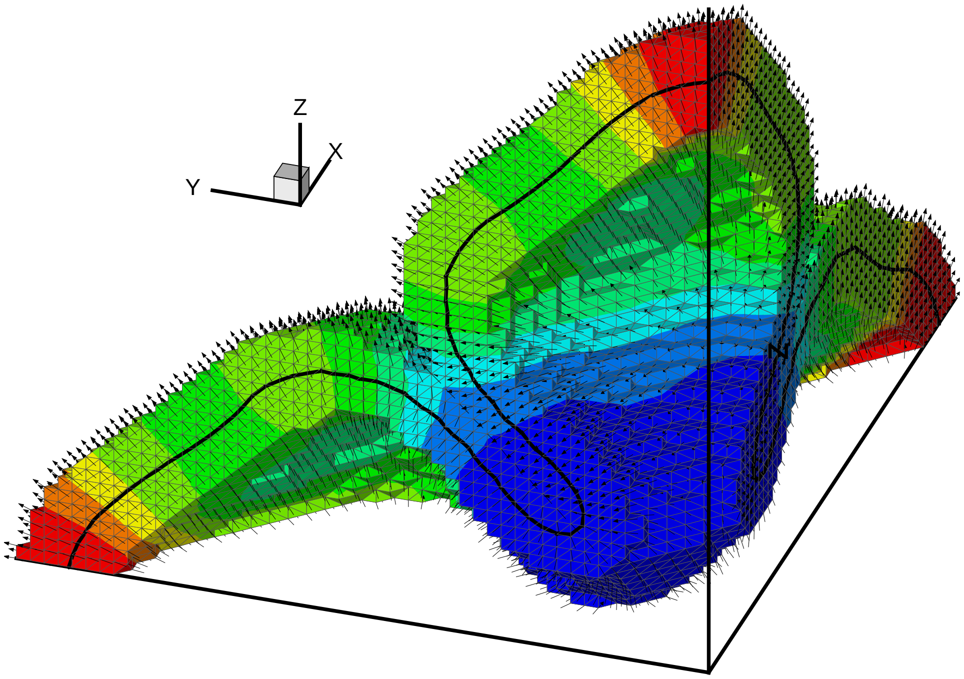


Figure 3.8: Narrow band technique for level set computation (contours show the interface velocity, the line refers to the interface position, and arrows indicate normal direction). This narrow band mesh is used for level set computation in the 3D crystal growth example (time step 51) that is discussed later in this paper in Section 3.4.5.

layers. Correspondingly, a small time step should be used initially. However, such small time step is not required in later stages. In order to speed up the computation, adaptive time step technique is adopted in this work. We select the time step size according to a CFL-type criterion of the form

$$\Delta t^{n+1} = \lambda_{CFL} w / \max(|V^n_{Interface}|), \tag{3.43}$$

where $\max(|V^n_{Interface}|)$ is the maximum interface velocity at all phase boundaries. λ_{CFL} is a coefficient controlling the accuracy, and unless is otherwise stated it is selected as 0.6 in all numerical examples. Eq. (3.43) guarantees that the interface advances about $0.6w$ in each time step at places with maximum interface velocity (normally at tips).

Note that in all of the subproblems including fluid flow, heat transfer, solute transport and level set evolution, fully-implicit backward Euler scheme is adopted for time integration in order to achieve unconditionally stability regardless of the time step size. The selection of the time step size is mainly based on solution accuracy requirements and is not related with stability issues [61].

3.3.5 Finite element implementation

In this work, the adaptive meshing technique generates triangle/tetrahedral type unstructured grids. The finite element technique is thus adopted for computations including fluid flow, heat transfer, solute transport and level set evolution. For heat transfer and solute transport, the classical SUPG stabilizing technique is used following our previous work [72]. Below, we will mainly concentrate on the stabilization techniques for fluid flow and level set computations.

Stabilized finite element method for fluid flow

Let us define the function spaces $S_{\boldsymbol{v}}$ and S_p as follows:

$$S_{\boldsymbol{v}} \overset{\text{def}}{=} \{\boldsymbol{v} | \boldsymbol{v} \in (L_2(\Omega))^{\text{nsd}}, \ \text{div}\boldsymbol{v} \ \in \ L_2(\Omega), \ \boldsymbol{v} = 0 \text{ on } \Gamma\}, \tag{3.44}$$

$$S_p \overset{\text{def}}{=} \{p | p \in L_2(\Omega), \ \int_{\Omega} p d\Omega = 0\}. \tag{3.45}$$

The classical Galerkin formulation for the flow problem (Eq. (3.30)) can be stated as follows: Find $\boldsymbol{V} \overset{\text{def}}{=} \{\boldsymbol{v}, p\} \in S_{\boldsymbol{v}} \times S_p$ such that for all $\boldsymbol{W} \overset{\text{def}}{=} \{\boldsymbol{w}, q\} \in S_{\boldsymbol{v}} \times S_p$ the following holds:

$$B(\boldsymbol{W}, \boldsymbol{V}) = L(\boldsymbol{W}), \tag{3.46}$$

where

$$B(\boldsymbol{W}, \boldsymbol{V}) = \int_{\Omega} \boldsymbol{w} \cdot \left(\frac{\rho \partial \boldsymbol{v}}{\partial t} + \frac{\rho \boldsymbol{v} \cdot \nabla \boldsymbol{v}}{\epsilon^{\ell}} + \frac{\epsilon^{\ell} \mu^{\ell}}{K(\epsilon^{\ell})} \boldsymbol{v} \right) d\Omega - \int_{\Omega} p \nabla \cdot \boldsymbol{w} d\Omega$$

$$+ \int_\Omega \mu^\ell \nabla \boldsymbol{w} \cdot (\nabla \boldsymbol{v} + \nabla^T \boldsymbol{v}) d\Omega + \int_\Omega q\rho \nabla \cdot \boldsymbol{v} d\Omega, \tag{3.47}$$

$$L(\boldsymbol{W}) = -\int_\Omega \boldsymbol{w} \cdot \epsilon^\ell \rho [\beta_c (C - C_0) + \beta_T (T - T_0)] \boldsymbol{g} d\Omega$$

$$+ \int_\Omega \frac{p}{\epsilon^\ell} \boldsymbol{w} \cdot \nabla \epsilon^\ell d\Omega \tag{3.48}$$

The above formulation theoretically works only for certain velocity and pressure interpolations. In the finite element implementation of the Navier-Stokes equations, stabilizing techniques are needed to accommodate equal-order interpolation velocity-pressure elements. Let us define a modified pressure space S'_p as follows:

$$S'_p \overset{\text{def}}{=} \{p | p \in H^1(\Omega), \int_\Omega q d\Omega = 0\}. \tag{3.49}$$

The stabilized weak form proposed here is the following: Find $\boldsymbol{V} = \{\boldsymbol{v}, p\} \in S_{\boldsymbol{v}} \times S'_p$ such that for all $\boldsymbol{W} = \{\boldsymbol{w}, q\} \in S_{\boldsymbol{v}} \times S'_p$ the following holds:

$$B_{\text{stab}}(\boldsymbol{W}, \boldsymbol{V}) = L_{\text{stab}}(\boldsymbol{W}), \tag{3.50}$$

where:

$$B_{\text{stab}}(\boldsymbol{W}, \boldsymbol{V}) = B(\boldsymbol{W}, \boldsymbol{V}) + \int_\Omega \mathcal{F}(\boldsymbol{v}, p) \cdot \mathcal{G}(\boldsymbol{w}, q) d\Omega,$$

$$L_{\text{stab}}(\boldsymbol{W}) = L(\boldsymbol{W}) + \int_\Omega \{\tfrac{p}{\epsilon^\ell} \nabla \epsilon^\ell - \epsilon^\ell \rho [\beta_c (C - C_0) + \beta_T (T - T_0)] \boldsymbol{g}\} \cdot \mathcal{G}(\boldsymbol{w}, q) d\Omega.$$

The following definitions have been introduced in the above equation:

$$\mathcal{F}(\boldsymbol{v}, p) = \rho \frac{\partial \boldsymbol{v}}{\partial t} + \frac{\rho \boldsymbol{v}_* \cdot \nabla \boldsymbol{v}}{\epsilon^\ell} + \nabla p + \frac{\epsilon^\ell \mu^\ell}{K(\epsilon^\ell)} \boldsymbol{v} - \nabla \cdot \left[\mu^\ell \left(\nabla \boldsymbol{v} + \nabla^T \boldsymbol{v} \right) \right], \tag{3.51}$$

$$\mathcal{G}(\boldsymbol{w}, q) = \tau_1 \frac{\boldsymbol{v}_* \cdot \nabla \boldsymbol{w}}{\epsilon^\ell} - \frac{\tau_2 \epsilon^\ell}{K(\epsilon^\ell) \rho} \boldsymbol{w} - \frac{\tau_3}{\rho} \nabla \cdot \left[\mu^\ell \left(\nabla \boldsymbol{w} + \nabla^T \boldsymbol{w} \right) \right] + \tau_4 \nabla q, \tag{3.52}$$

with $\boldsymbol{v}_*$ a divergence-free velocity obtained from the previous iteration.

τ_1, τ_2, τ_3 and τ_4 are parameters for the advection (SUPG), Darcy drag (DSPG), diffusion and pressure (PSPG) stabilizing terms. The selection of their values is discussed in [72, 64].

Stabilized finite element method for level set computation

The weak form of the level set equation (Eq. (3.10)) is

$$\int_{\Omega} \delta\phi^{\alpha} \left(\phi_t^{\alpha} + V^{\alpha}|\nabla\phi^{\alpha}|\right) d\Omega = 0,$$

where $\delta\phi^{\alpha}$ is a test function in C^0. To avoid oscillations, GLS and diffusion stabilizing term are introduced as shown in the following formulation [13]:

$$\sum_{e=1}^{N_{el}} \int_{\Omega^e} \left[\delta\phi^{\alpha} + \left(\tau^e_{\phi^{\alpha}} \frac{V^{\alpha}\nabla\delta\phi^{\alpha} . \nabla\phi^{\alpha}}{|\nabla\phi^{\alpha}|}\right)\right] \left(\phi_t^{\alpha} + V^{\alpha}|\nabla\phi^{\alpha}|\right) + \left(\upsilon\nabla\delta\phi^{\alpha} \cdot \nabla\phi^{\alpha}\right) d\Omega^e = 0,$$

which further leads to the following semi-discretized form:

$$\begin{aligned}
&(M_{\phi^{\alpha}} + M_{GLS})\dot{\phi^{\alpha}} + f_{\phi^{\alpha}} + f_{GLS} + f_{SC} = 0,\\
&M_{\phi^{\alpha}} = \sum_{e=1}^{N_{el}} \int_{\Omega^e} N^T N d\Omega^e, M_{GLS} = \sum_{e=1}^{N_{el}} \int_{\Omega_e} \left(\nabla N^T \cdot \frac{\nabla\phi^{\alpha}}{|\nabla\phi^{\alpha}|} V^{\alpha}\right) \tau^e_{\phi^{\alpha}} N d\Omega^e,\\
&f_{\phi^{\alpha}} = \sum_{e=1}^{N_{el}} \int_{\Omega^e} N^T V^{\alpha}|\nabla\phi^{\alpha}| d\Omega^e, f_{GLS} = \sum_{e=1}^{N_{el}} \int_{\Omega^e} \left(\nabla N^T \cdot \frac{\nabla\phi^{\alpha}}{|\nabla\phi^{\alpha}|} V^{\alpha}\right) \tau^e_{\phi^{\alpha}} V^{\alpha}|\nabla\phi^{\alpha}| d\Omega^e,\\
&f_{SC} = \sum_{e=1}^{N_{el}} \int_{\Omega^e} \upsilon\nabla N^T \cdot \nabla\phi^{\alpha} d\Omega^e.
\end{aligned}$$

In our computation, the GLS stabilize term parameter τ^e_{ϕ} is selected as $10^{-3}\frac{h^e}{|V^{\alpha}|}$. We use a larger diffusion stabilizing term at triple points than at other places:

$$\upsilon = \begin{cases} 10^{-4}h^e & \text{for triple points } (\exists\ \alpha \neq \beta \neq \gamma,\ \text{s.t. } \epsilon^{\alpha}, \epsilon^{\beta}, \epsilon^{\gamma} > 0) \\ 10^{-6}h^e & \text{otherwise.} \end{cases}$$

3.3.6 Overall solution procedure

The various subproblems considered here include the thermal, flow, solute species problems as well as the interface evolution. The tolerance level used to define convergence in all four main solution steps (heat, solute, momentum and level set) is set at 10^{-12}. The error criterion is based on the relative error in the solutions obtained between Newton-Raphson iterations within a time step. For example, in the fluid flow solver, the error norm is defined as $\|\Delta U^{i+1}\|/\|U_n^{i+1}\|$ where $U = [\boldsymbol{v}, p]$. The overall algorithm is summarized below:

1. At time t_n, all fields such as velocity $\boldsymbol{v}_n$, temperature T_n, concentrations C_{in}, level set variables ${\phi^\alpha}_n$, etc. are known.
2. Determine Δt (adaptive time step) and generate the new mesh for this time level (adaptive meshing).
3. Advance to time step $t_{n+1} = t_n + \Delta t$. Set $j = 0$, $\boldsymbol{v}_{n+1}^{j=0} = \boldsymbol{v}_n$, $T_{n+1}^{j=0} = T_n$, ${C_i}_{n+1}^{j=0} = C_{in}$, ${\phi^\alpha}_{n+1}^{j=0} = {\phi^\alpha}_n$, etc., where j is an iteration index and the subscript denotes the time level.
4. Start an inner loop to solve the coupled system till converged.
    - (a) Compute velocity at phase boundaries, use fast marching to extend velocity to the whole domain.
    - (b) Solve the level set equation for all phases to get the signed distance functions ${\phi^\alpha}_{n+1}^{j}$.
    - (c) Solve the energy equation to obtain T_{n+1}^{j}.
    - (d) Solve the solute equations to obtain ${C_i}_{n+1}^{j}$ for each component i.
    - (e) Solve the momentum equation to obtain $\boldsymbol{v}_{n+1}^{j}$.
    - (f) $j = j + 1$. If not converged, return to (a).
5. Use the re-initialization scheme to remove gaps or overlapping between ϕ^α for different phases and ensure that all ϕ^α are signed distance functions.

3.4 Numerical examples

3.4.1 Crystal growth in an undercooled binary alloy

This example was originally addressed in [75] using a front-tracking method. The material parameters defining the problem are density $\rho^s = \rho^\ell = 1$, heat capacity $c^s = c^\ell = 1$, heat conductivity $k^s = k^\ell = 1$, latent heat $L = 1$, solute diffusivity $D^\ell = 0.1$, $D^s = 0$, liquidus slope $m = -0.035$ (so that $T^{s\ell}(C^\ell) = mC^\ell$) and partition coefficient $k_p = 0.312$. On the freezing interface $\Gamma^{s\ell}$, we consider the Gibbs-Thomson relation given in Eq. (3.6) with $\varepsilon_c = \varepsilon_V = -0.002$.

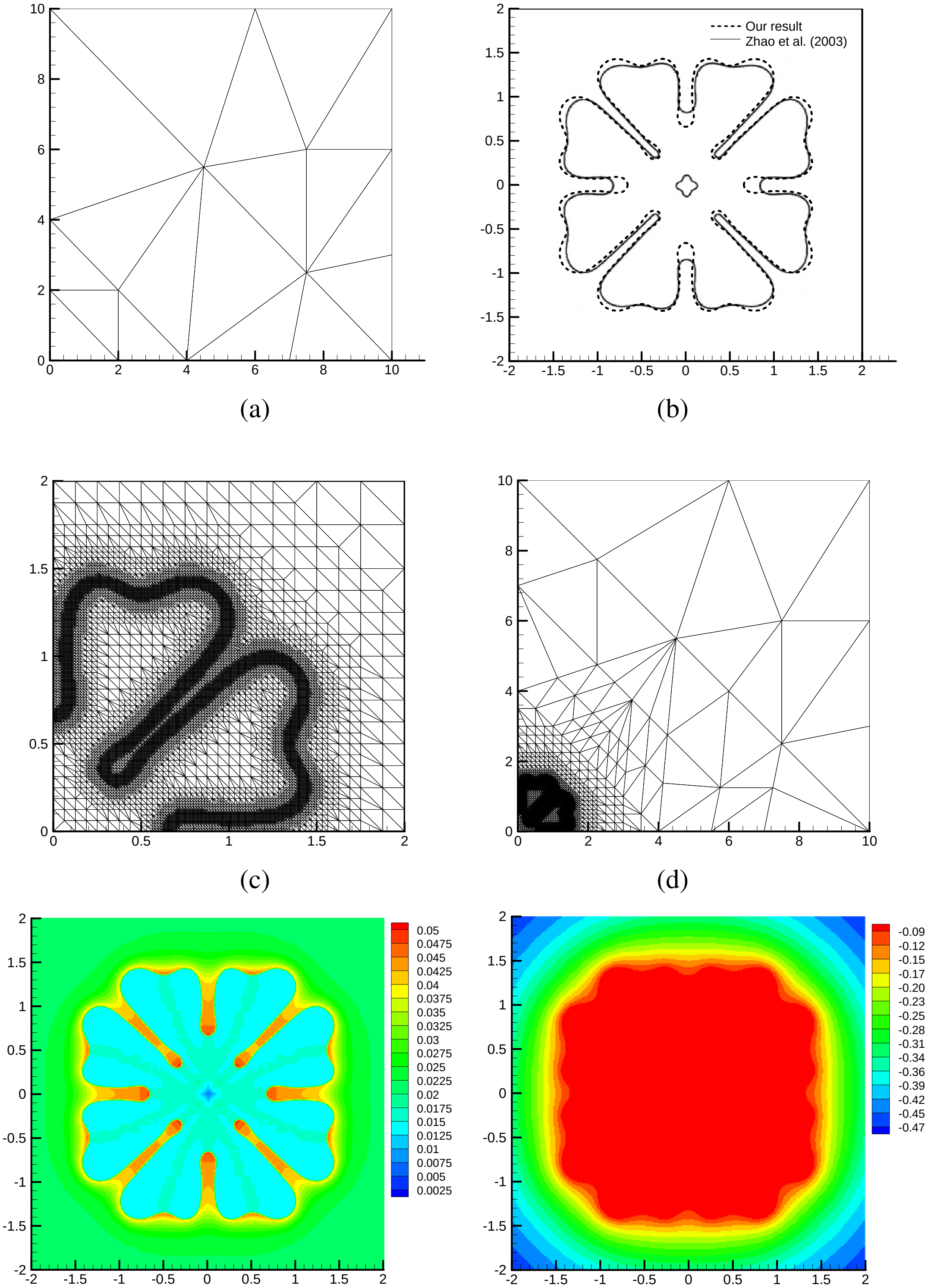
At time zero, we consider a small solid seed in the middle of the computational domain. Its geometry is described as follows:

$$\begin{aligned} x(s) &= (R + P\cos(8\pi s))\cos(2\pi s), \\ y(s) &= (R + P\cos(8\pi s))\sin(2\pi s), \end{aligned} \tag{3.53}$$

where $R = 0.1$ and $P = 0.02$. The initial concentration of the seed and the undercooled melt are $k_p C_0$ and C_0, respectively with $C_0 = 2.2\%$. The initial temperature of the seed and the undercooled melt are taken as $T^{s\ell}(C_0)$ and $T^{s\ell}(C_0) - 0.5$, respectively.

Fluid flow is not considered in this example. The computational domains for the temperature and solute fields are taken as $[-10, 10] \times [-10, 10]$ and $[-2, 2] \times [-2, 2]$, respectively. Insulated boundary conditions are considered for both temperature and solute at all sides of the respective computational domains.

Based on the very coarse input mesh shown in Fig. 3.9a, an adaptive mesh is generated in each time level with a part of it used for the calculation of the solute field and the whole mesh used for temperature calculations as shown in Figs. 3.9c and 3.9d,

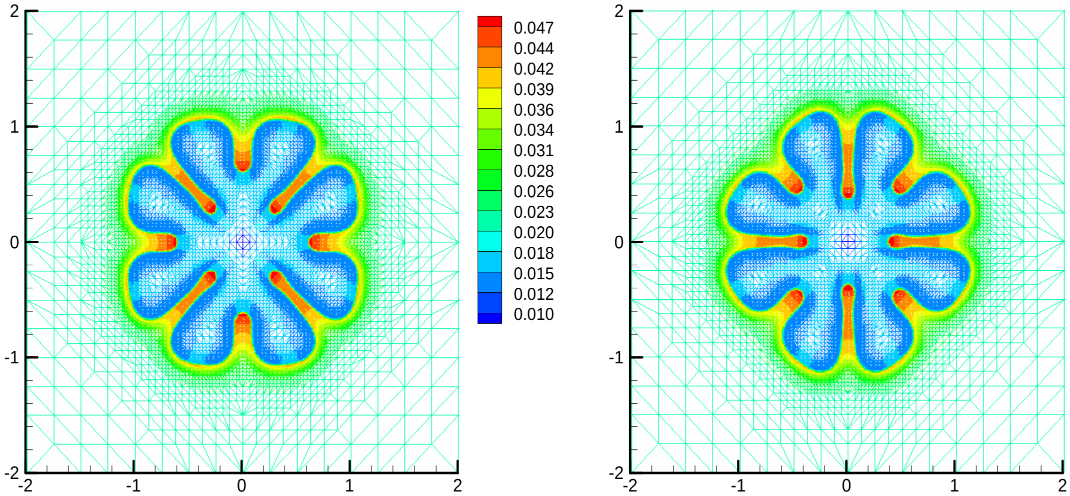


(a) (b) (c) (d) (e) (f)

Figure 3.9: (a) 1/4 of the input mesh (b) The interface at $t = 1.0$ (c) 1/4 of the mesh used for solute computation ($t = 1.0$) (d) 1/4 of the mesh used for temperature computation ($t = 1.0$) (e) Solute concentration ($t = 1.0$) (f) Temperature ($t = 1.0$).

respectively. At $t = 1.0$, the effect of the domain boundary on the temperature field is small since the temperature boundary layer is far away from the boundary as shown in Fig. 3.9f. The obtained micro-segregation pattern with maximum concentration about 5.0% is shown in Fig. 3.9e and it compares well with the results obtained in [75] using a sharp-interface (front-tracking) model. Comparison of the interface position at $t = 1$ with the results given in [75] is shown in Fig. 3.9b.

We also tested the effects of mesh orientation by rotating the initial crystal by 45 degrees. The mesh for solute transport colored with solute concentration at time about 0.6 is shown in Fig. 3.10b. The similarity between Figs. 3.10a and 3.10b indicates that the obtained numerical results do not depend on the mesh orientation.



(a) (b)

Figure 3.10: Mesh for solute transport at time about 0.6 colored with the concentration field (a) without crystal rotation and (b) with crystal rotation.

Using mesh refinements based on the level set, temperature gradient (tolerance = 2×10^{-5}) and concentration gradient (tolerance = 1×10^{-5}), the meshes shown in Fig. 3.11 were generated with a quarter of each mesh shown for easy comparison. It can be seen that all meshes provide similar results for the interface position, except that the interface position generated using the ∇T-based mesh criterion is a bit flattened at the dendrite 'valleys'. This is due to the small temperature gradient and correspondingly low mesh density at this location. With the criterion based on ∇C, many more elements were generated at the 'valleys'. This suggests that the criterion based on ∇T is not as good as the criterion based on ∇C or ϕ. The mesh generated with the criterion based on ∇C is very similar to the mesh generated with the criterion based on ϕ, except that it results in a higher mesh density at the middle of the crystal due to micro-segregation within the solid crystal.

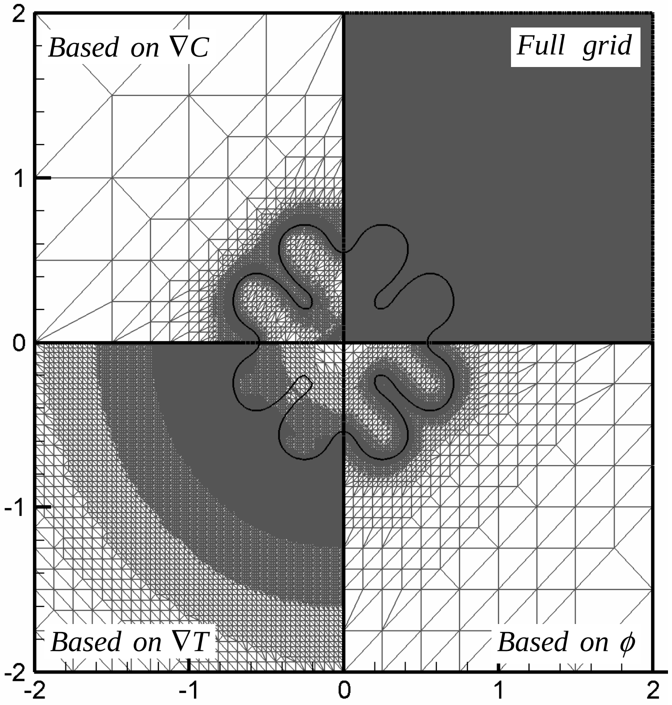


Figure 3.11: Mesh generated using different mesh-refinement criteria.

In the above benchmark problem, we compared our results with the results obtained using a front tracking method. To allow a comparison with the phase field method for further validation of our algorithm, we briefly consider a problem similar to the one examined in [53] by choosing material properties $k_s = k_l = 1$, $c_s = c_l = 1$, $L = 1$, $\rho = 1$, $k_p = 0.15$, $m_l = -1$, $C_0 = \frac{0.07}{1-0.15}$, $D_l = \frac{1}{40}$ and undercooling 0.55. We write the values of C_0 and D_l in this unfinished form, so that the reader can easily refer to both papers for the connection between the two models. The initial crystal used in our computation is a circular seed with radius 80 and without perturbation. The anisotropy in the Gibbs-Thomson relation is defined as $T^* = -d_0(1 - 15\varepsilon \cos 4\theta)\kappa,\ d_0 = 1,\ \varepsilon = 0.02$, where θ is the angle between the interface direction and the x axis. We use an initial grid of $1{\times}1$ and 9 refinement levels to discretize a domain of 1500×1500. The results of this example are shown in Fig. 3.12. To facilitate comparison between the two models, we provide the computed dimensionless temperature at the dimensionless time $280,000$. However, for concentration shown in the results, we use the concentration C instead of the definition in [53]. This is because the definition of dimensionless concentration in [53] requires a phase field variable, which is not applicable in our model. Comparing our results with those in Fig. 1 of [53], we note that the tip radius and position are practically the same. The solute boundary layer is also found to be much thinner than the thermal boundary layer due to the large Lewis number. However, there are some small differences of the interface shape at location around (300,300), where perturbations are about to develop.

3.4.2 Planar/cellular/dendritc transition

The relationship between the solidification growth velocity and the primary arm spacing described by the Mullins-Sekerka theory has been numerically studied using phase field methods [9, 60]. In these studies, the temperature gradient is assumed to be constant during directional solidification and the energy equation is not solved. Instead, the temperature field is obtained by moving the initial temperature field with a given velocity. In this example, the simulations are performed under similar conditions to those of [9, 60]. However, the simplifications on the temperature field are not taken here. The framework shown in Fig. 3.13 is adopted for a Ni-Cu alloy with the following important parameters [9, 60]: melting temperature $T_m = 1728K$, liquidus slope $m = 357$ K/at. frac.,

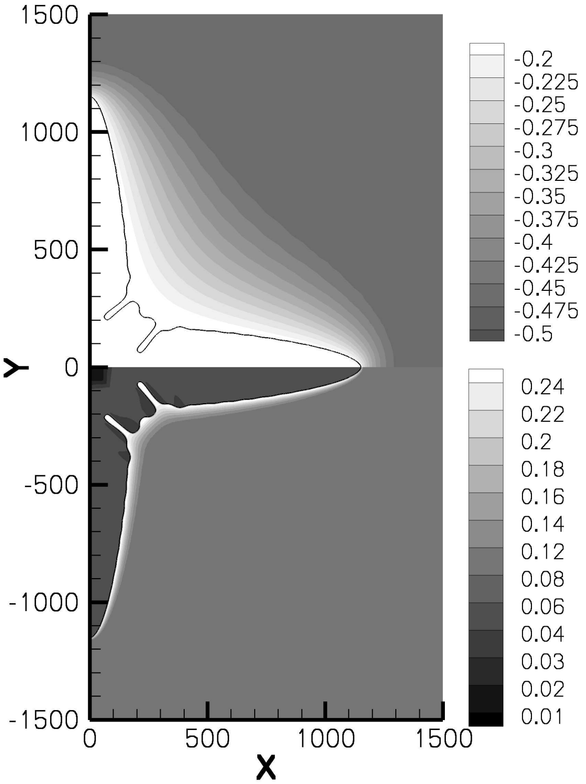


Figure 3.12: The solid/liquid interface shape at time 280, 000. The upper and lower boxes show the temperature and solute concentration fields, respectively.

initial concentration $C_0 = 0.40831$ at. frac., latent heat $L = 2350 J/cm^3$, solute diffusivity $D^\ell = 10^{-5} cm^2/s$, surface tension $\sigma = 3.7 \times 10^{-5} J/cm^2$, Gibbs-Thomson coefficient $\varepsilon_c = \sigma T_m/L = 2.7207 \times 10^{-5} cmK$, temperature gradient $G = q^\ell/k = 2.15 \times 10^4 K/cm$, partition coefficient $k_p = 0.86$, heat capacity $c = 0.46 \times 10^3 J/(kgK)$, density $\rho = 8.880 \times 10^{-3} kg/cm^3$, heat diffusivity $k = 6.07 \times 10^{-1} W/(cmK)$, and domain size is $[(1.76 \times 10^{-3} cm) \times (4.40 \times 10^{-3} cm)]$. No initial perturbation is introduced (an initially planar solid-liquid interface is assumed). The numerical error acts as a perturbation that leads to unstable growth pattern for the four considered growth velocity values (0.1 cm/s, 0.2 cm/s, 0.4 cm/s and 0.8 cm/s) as shown in Fig. 3.14.

We also plotted the obtained wavelengths for different growth velocities in Fig. 3.15

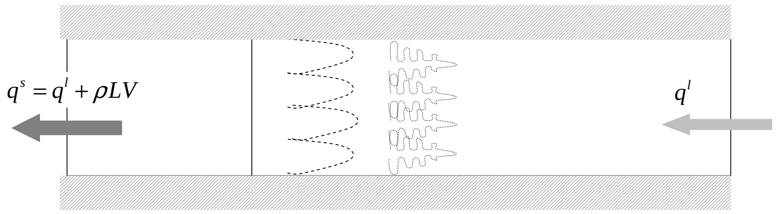
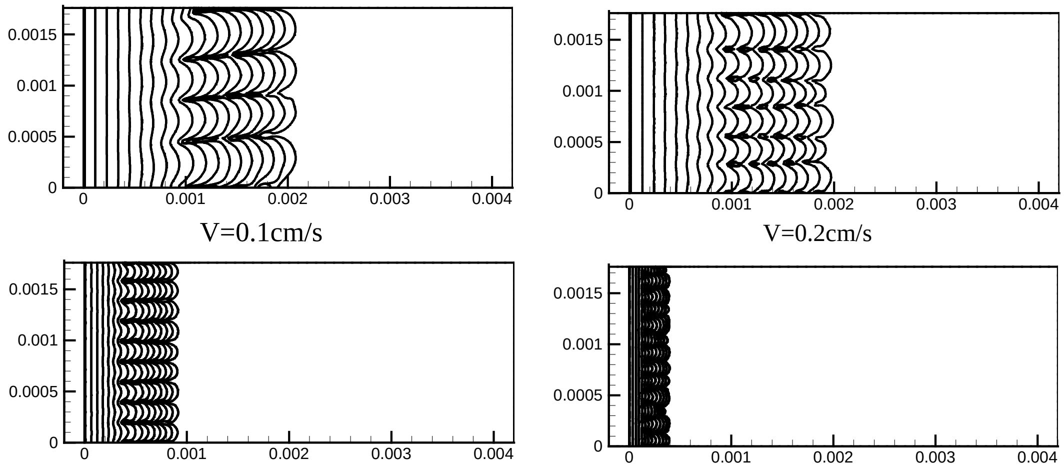


Figure 3.13: Problem definition for study of planar/cellular/dendritic transition.



V=0.1cm/s

V=0.2cm/s

V=0.4cm/s

V=0.8cm/s

Figure 3.14: Cellular growth for various growth velocities (axes unit: cm).

together with $\lambda^2 V = 1.6 \times 10^{-14} m^3/s$ and the MS (Mullins-Sekerka) loop. The MS loop is the zero contour plot of $\frac{\dot{\delta}}{\delta}$, where δ is the perturbation amplitude with wavelength λ [40]. Within the MS loop, $\frac{\dot{\delta}}{\delta} > 0.$, the perturbation with wavelength λ increases and therefore the interface is unstable. Outside the MS loop, $\frac{\dot{\delta}}{\delta} < 0$ and the perturbation with wavelength λ damps out. Planar interfaces are obtained with growth velocity $0.01 cm/s$ and $10 cm/s$ as demonstrated in Fig. 3.16. The upper limit of solidification speed for cellular growth is the absolute stability, V_a, velocity. Its approximate value is given in [34] as $V_a \approx \frac{-m_l C_0 (1-k_p) D}{k_p^2 \Gamma}$, which is about $10 cm/s$ in this problem.

In the above computation, we use an initial mesh of 50×20. Two levels of refinement for velocity $0.01 cm/s, 0.1 cm/s, 0.2 cm/s$ and three levels of refinement for velocity $0.4 cm/s, 0.8 cm/s, 10 cm/s$ are performed using the adaptive meshing technique. A typ-

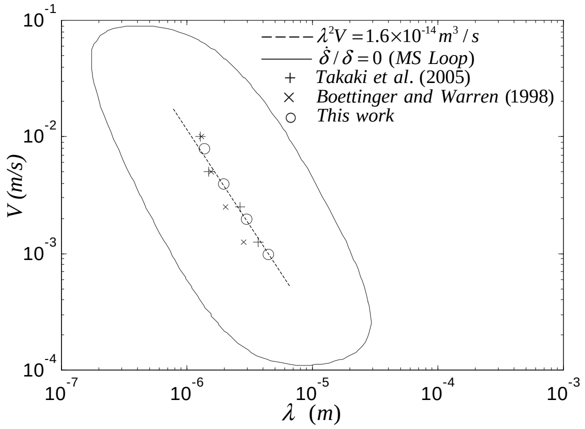
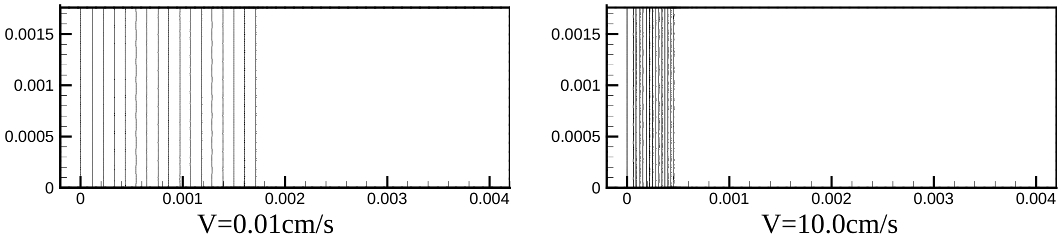


Figure 3.15: Relationship between the solidification speed and arm spacing.



V=0.01cm/s

V=10.0cm/s

Figure 3.16: Planar growth at small and large velocities (Axes unit: cm).

ical mesh colored with solute concentration is shown in Fig. 3.17.

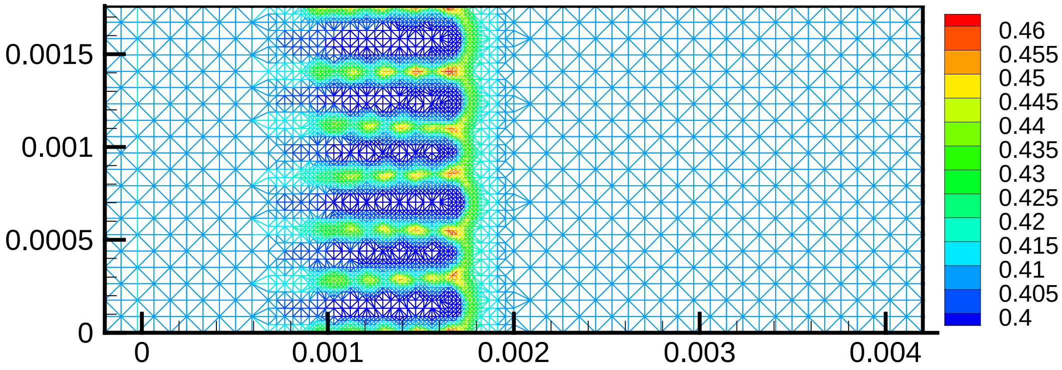


Figure 3.17: Adaptive mesh colored with solute concentration.

The computed solute boundary layer in front of the dendrite tips is shown in Fig. 3.18. Using the method discussed in [34], the equivalent solute boundary layers shown in Fig. 3.18 are about $2.2\times10^{-4}cm$, $1.2\times10^{-4}cm$, $0.6\times10^{-4}cm$ and $0.3\times10^{-4}cm$

for growth velocity values of $0.1cm/s$, $0.2cm/s$, $0.4cm/s$ and $0.8cm/s$, respectively. These values agree with the analytical approximation of the solutal boundary layer ($\approx 2D_l/V \propto 1/V$).

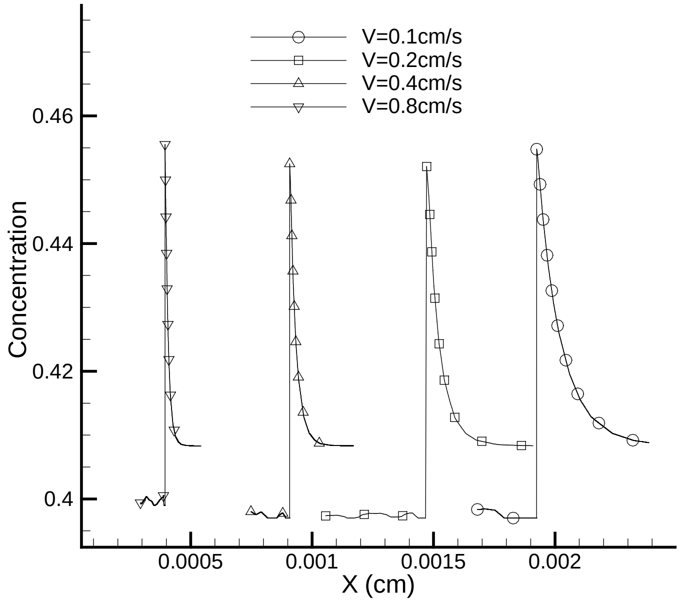


Figure 3.18: Solute boundary layer for various growth velocities.

For the initial solute concentration $C_0 = 0.40831$, we only observed planar and cellular growth patterns. The surface tension and thermal gradient prevent the transition from cellular to dendritic. Generally, the growth pattern is less stable for material with smaller surface tension, smaller partition coefficient and higher concentration.

In this case, the surface tension and partition coefficient are assumed to be constant. In order to capture the dendritic growth pattern of Ni-Cu alloy, we increase the copper concentration to 0.6. With growth speed $0.2cm/s$, we observed three solidification stages: (1) planar growth (2) cellular growth and (3) dendritic growth (Fig. 3.19).

Initially, the growth pattern is planar. Then about 6 cells are formed. Due to competition, only three major cells are left in the computational domain. Secondary branches are observed in the segregation pattern (Fig. 3.19).

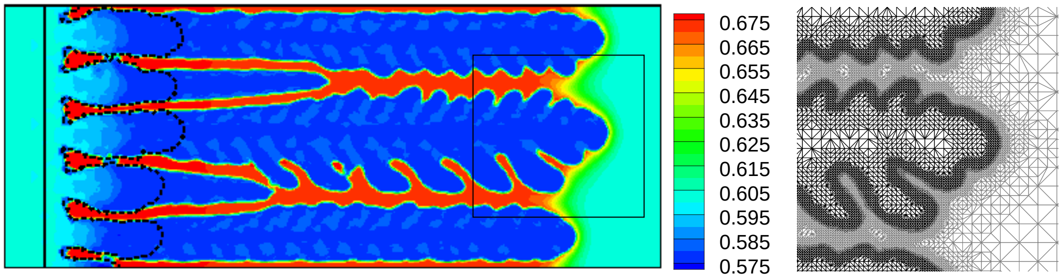


Figure 3.19: Solute concentration and adaptive mesh near tips of dendritic growth at copper concentration 0.6 $at.\ frac.$

3.4.3 Alloy solidification with multiple solid phase growth

For many alloy solidification systems, multiple solid phases are formed. Eutectic and peritectic are the most common solidification microstructures with multiple solid phases. These solidification systems were addressed earlier using multi-phase field models with a comprehensive review given in [23]. Here we present the results obtained with the present level set model for an eutectic growth problem [4] and a peritectic growth problem [65].

Eutectic growth

For directional eutectic solidification with two solid phases α and β, we use the same framework shown in Fig. 3.13. A temperature gradient $G = 50$ K/cm is maintained at the right side and a cooling rate of $R = 0.005$ K/s is applied at the left side. Initially, a few β seeds are embedded in the α phase. Other important parameters are [4]: liquidus slops $m^{\alpha} = -10$ K/at% and $m^{\beta} = +5$ K/at%, partition coefficients $k_p^{\alpha} = 0.1$ and $k_p^{\beta} = 0.05$, diffusion coefficient $D^{\ell} = 10^{-5}$ cm^2/s, surface energy $\sigma = 10^{-5}$ J/cm^2, eutectic composition $C^{eut}/C^{\beta} = 0.2$ with $C^{\beta} = 5$ at%, latent heats $L^{\alpha} = L^{\beta} = 1000$ J/cm^3, heat capacity $c = 1 \times 10^3$ J/(kg K), density $\rho = 1 \times 10^{-2}$ kg/cm^3, heat diffusivity $k = 1.0$

W/(cm K), eutectic temperature $T_e = 1000$ K and finally domain size $[0.4mm \times 0.12mm]$.

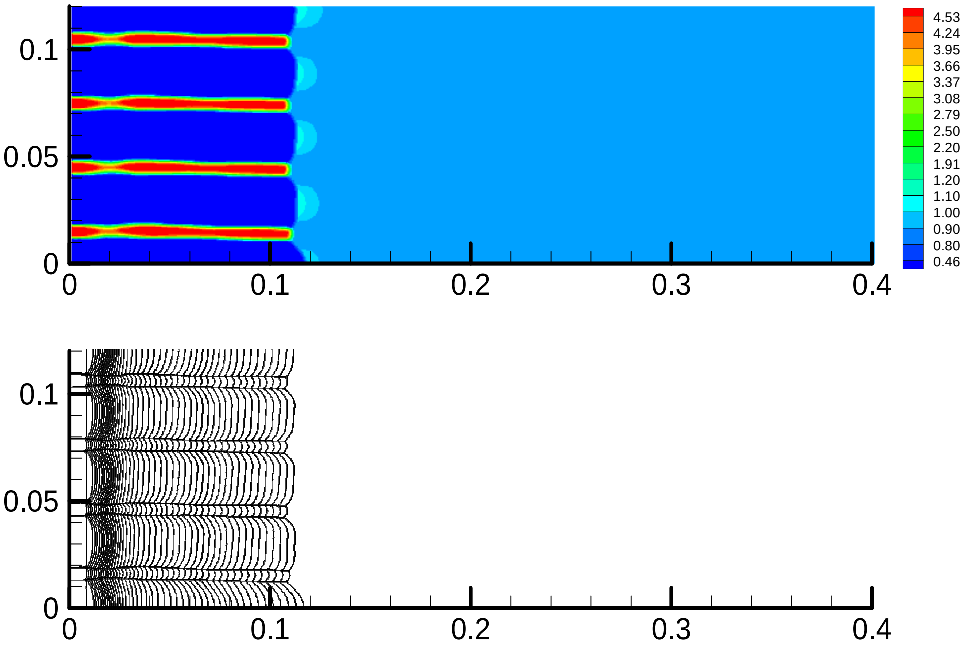


Figure 3.20: Solute concentration at time about 80 s and the evolution of the interface position (with four β seeds).

By putting initially 4 β seeds (red region shown in Fig. 3.20) into the α phase (blue region shown in Fig. 3.20), the eutectic growth follows a stable pattern as shown in Fig. 3.20 with a stable growth wavelength of about $30\mu m$. In [4], the stable growth wavelength reported was about $25\mu m$. If only two β seeds are inserted initially in the α phase, then the eutectic growth becomes unstable. Valleys with high solute concentration will form in the middle of the $\alpha - \ell$ phase interface as shown in Fig. 3.21. This is because with only two β seeds, the solute rejected during the transformation of the liquid to the α solid phase cannot be easily absorbed by the β phase. The accumulation of solute will further slow down the growth rate of the α phase and form valleys. Since the high solute concentration favors the growth of β phase, the width of the β phase increases as shown in Fig. 3.21. This unstable eutectic growth pattern for fewer β phase seeds was previously pointed out in [4]. If ten β seeds are inserted initially in the α

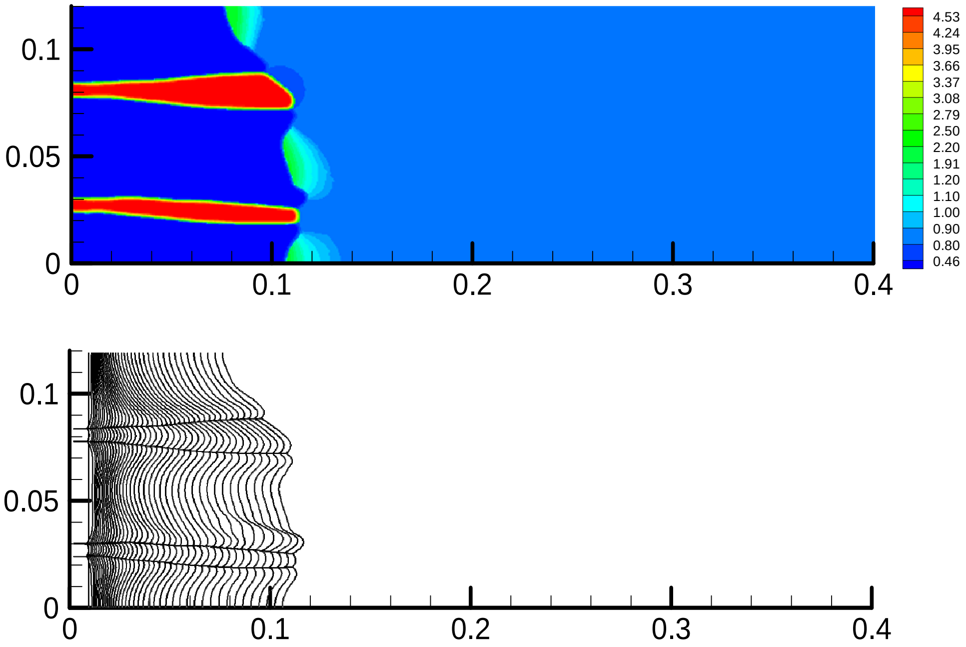


Figure 3.21: Solute concentration at time about 80 s and the evolution of the interface position (with two β seeds).

phase, half of the seeds stop growing after the tips are about $6\mu m$ away from the left boundary. The growth of the remaining five seeds shows an oscillatory pattern. Finally, one more seed stops growing. The remaining four β seeds adjust their wavelength to about $30\mu m$ as shown in Fig. 3.22.

Peritectic growth

Fe-0.3wt%C alloy is considered with the following parameters: domain size $150\mu m \times 700\mu m$, cooling rate $R = -3K/s$ applied at the bottom, thermal gradient $G = 140K/cm$ maintained at the top, diffusion coefficient $D_l = 3 \times 10^{-5} cm^2/s$ and surface tension $\sigma = 2.04 \times 10^{-4} J/cm^2$. The thermodynamic data are taken from the Fe-C phase diagram [65]. Initially, an initial small crystal is put in the bottom-left corner. The initial grid size is 3×14 with 5 levels of refinement. During the growth of the ferrite the redis-

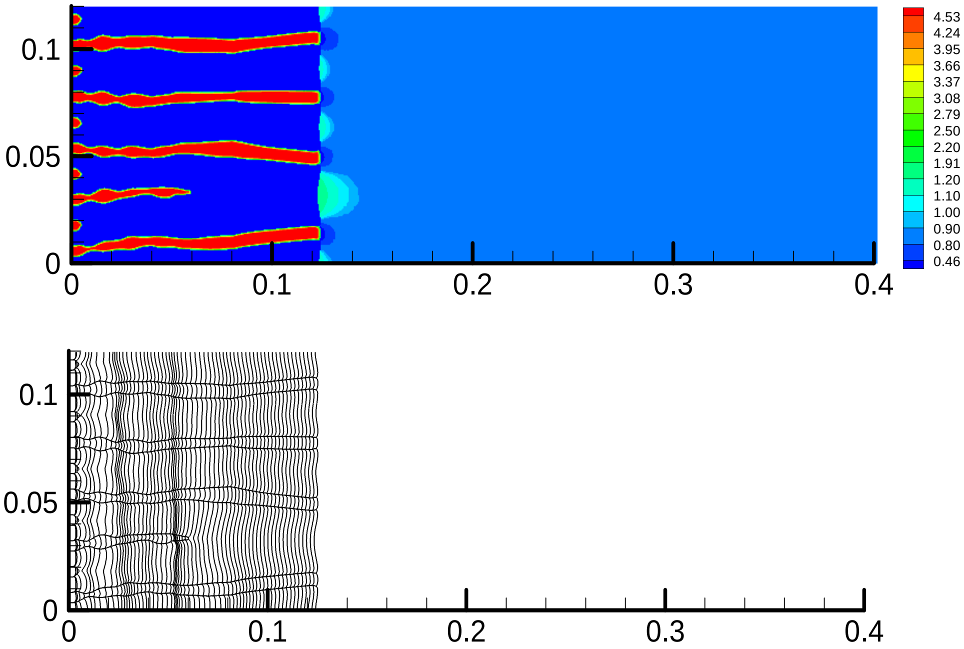


Figure 3.22: Solute concentration at time about 80 s and the evolution of the interface position (with ten β seeds).

tribution of carbon leads to morphological instability and a dendrite evolves. Below the peritectic temperature the austenitic seeds are generated by nucleation. The austenite grows around the undercooled part of the ferritic dendrite as shown in Fig. 3.23. Before time 1.6 s, the solute concentration is less than $0.51wt\%$ and thus only the ferritic phase is formed. As the solute concentration accumulates to be more than $0.51wt\%$ behind the dendrite tips, the austenitic phase is nucleated and formed around the secondary dendrite arms. Table 3.1 provides a comparison of the computed results with those reported in [65].

3.4.4 Ternary alloy with melt convection effects

Crystal growth for pure materials under convection has been well studied using phase field methods [6, 66, 28], front tracking methods [3] and level set methods [61]. In

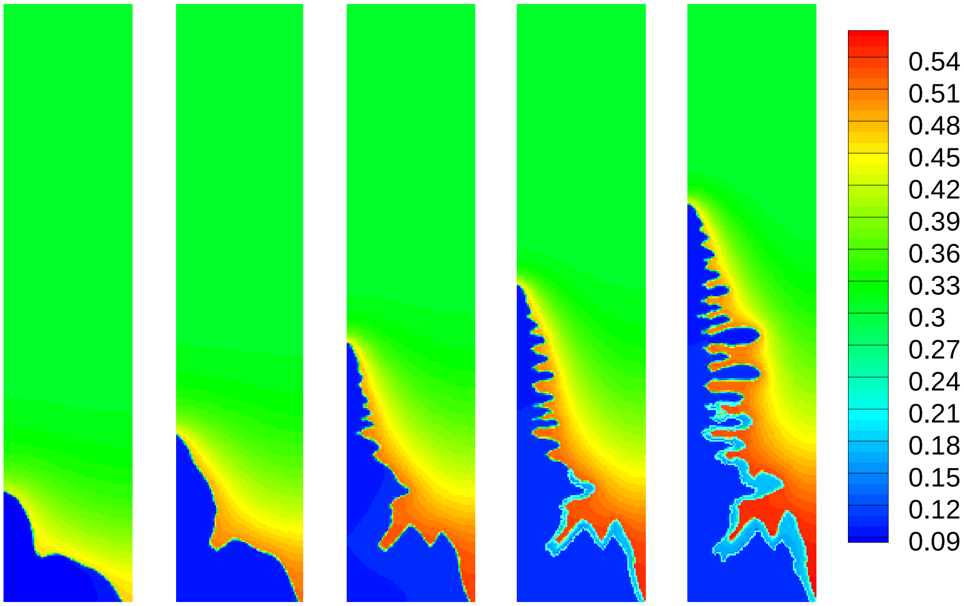


Figure 3.23: Solute concentration for peritectic growth of an Fe-C alloy at time 0.6 s, 1.0 s, 1.5 s, 1.8 s and 2.4 s.

Table 3.1: Comparison of the results obtained with the current methodology for peritectic growth with those given in [65].

Quantity	Present	From [65]
Tip velocity of ferritic dendrite	$195\mu m/s$	$200\mu m/s$
Distance of austenitic phase behind dendrite tip	$300\mu m$	$350\mu m$
Solute boundary layer (from contour plot)	$\approx 50\mu m$	$\approx 50\mu m$
Time of austenitic phase growth	$1.6s$	1.5~ 2.2s

these studies, the growth of only one crystal is investigated with a prescribed inlet flow velocity. In [14], Ostwald ripening of Al-Cu particles in the presence of melt convection was studied using the phase field method with a flow driven by a constant pressure drop. Fully coupled dendritic alloy solidification growth with heat, solute and momentum transport is computationally very difficult due to the various length scales involved. In order to resolve the dendrite tip or solute boundary layer, a small grid spacing is required. For the study of crystal growth of pure materials with a prescribed flow as in [3, 61, 6, 66, 28], a computation domain in the size of the thermal boundary layer is required. Developing substantial buoyancy driven flow requires even larger domain and correspondingly larger grid node/element number. The example in this section is the first study fully coupling dendritic growth with heat, solute and momentum transport for alloy solidification.

In this example, we consider a Ni based alloy with 5.8 wt%Al and 15.2 wt% Ta. The purpose of this example is to demonstrate the ability of the algorithms presented in this paper to couple solute diffusion of multiple components with convection. Only one solid phase is considered. Thermodynamic and transport properties for this alloy are obtained from [18] based on the one solid phase simplification.

Initially a cavity of size $0.025m \times 0.025m$ is filled with alloy of Ni-5.8wt%Al-15.2wt%Ta at its liquidus temperature (without superheat or undercooling). A cooling rate of 0.28K/s is applied at the bottom and sides. The top side is assumed to be adiabatic. Utilizing symmetry, computation is done only in the left half of the domain. For adaptive meshing, we use an initial grid 1×2 and a refinement level of 10, which corresponds to a full mesh of 1024×2048. With this mesh, the width of the diffused interface is $12.2\mu m$, which is a reasonable value for meso-scale simulations. The CFL coefficient for adaptive time stepping is 0.5.

In order to study the effects of convection, we carried two simulation runs: one without convection and one with buoyancy driven flow. With convection, the growth rate is about 22% faster than without convection. The dendrite tip front reaches roughly the same position as shown in Figs. 3.24 and 3.25 at time about 161 s in the case with convection and 196s in the case without convection. Figure 3.24 shows that the concentration of Al in the solid phase is slightly decreased due to convection. The underlying reason is that fluid flow enhances solute redistribution by taking away solute rejected from the dendrite tips into the liquid bulk. On the other hand, solute accumulated before the dendrite tips can only be taken away by diffusion. This phenomenon is more obvious for the Ta alloy component as shown in Fig. 3.25. The lower solute concentration in front of the dendrite tips explains why the growth speed is higher for the case with convection. Fluid flow also causes segregation. Solute concentration at the bottom of the cavity is higher than at the top of the cavity. This leads to a significant difference in the dendrite growth velocity resulting in a nonuniform pattern as shown in Figs. 3.24 and 3.25.

The adaptive mesh at time 161s for the convection case is shown in Fig. 3.26. As shown in the magnified picture in Fig. 3.26, fluid flow passes through the dendrite tips and creates small eddies which enhance the mixing of solute.

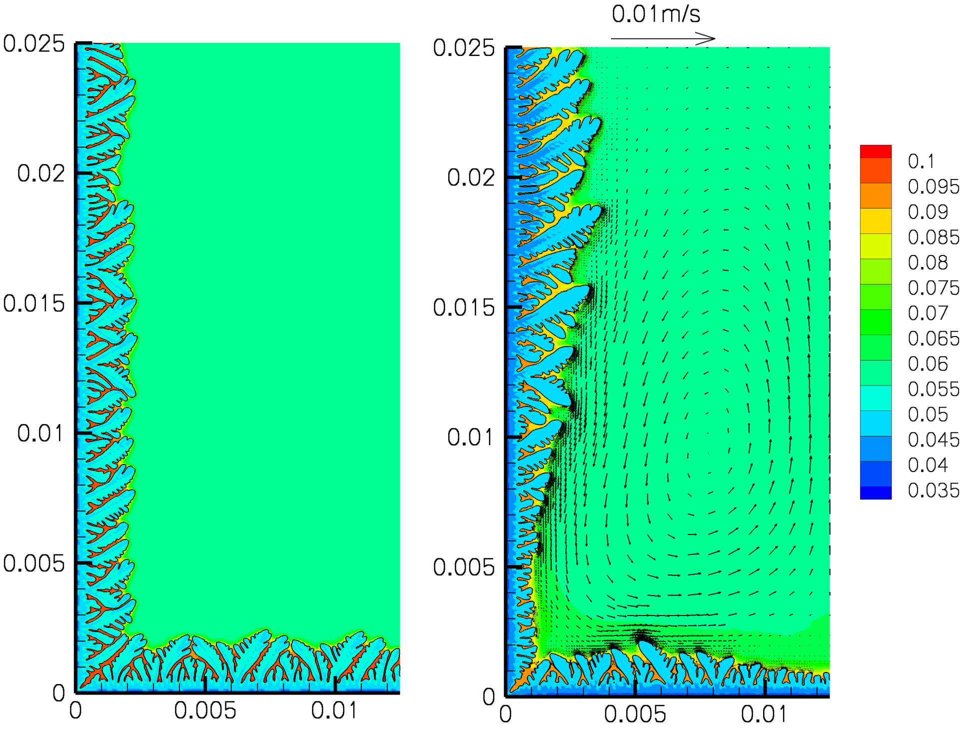


Figure 3.24: Solute concentration of Al. On the left: without convection and on the right: with convection.

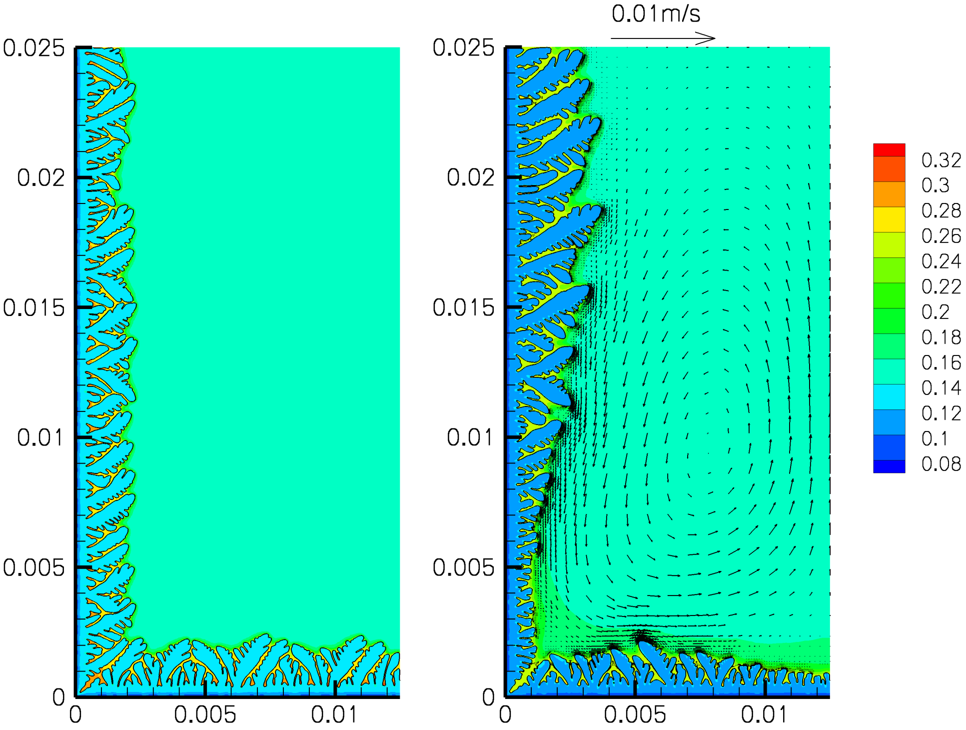


Figure 3.25: Solute concentration of Ta. On the left: without convection and on the right: with convection.

3.4.5 Three-dimensional dendritic growth

Using Ni-Cu alloy with properties the same as in the example of Section 4.3.2, a simulation was presented earlier of 3D crystal growth using a phase field model assuming constant temperature (1356 K) in a cubic domain of size $3.5 \times 10^{-3} cm$ [21]. An explicit scheme was used to perform computation on a mesh of 500^3 (125 million degrees of freedom). In this work, we use an adaptive mesh with initial grid 5^3 and 6 levels of refinement, which is equivalent to a uniform mesh of 320^3 (32 million degrees of freedom). With very high under-cooling (about 226 K), a high temperature gradient exists. So the simplification on the temperature considered in [21] (i.e. assuming constant temperature) is not taken here. As shown in Fig. 3.27, substantial secondary dendrite arms are formed behind the primary dendrite arms. Due to the significant under-cooling (226

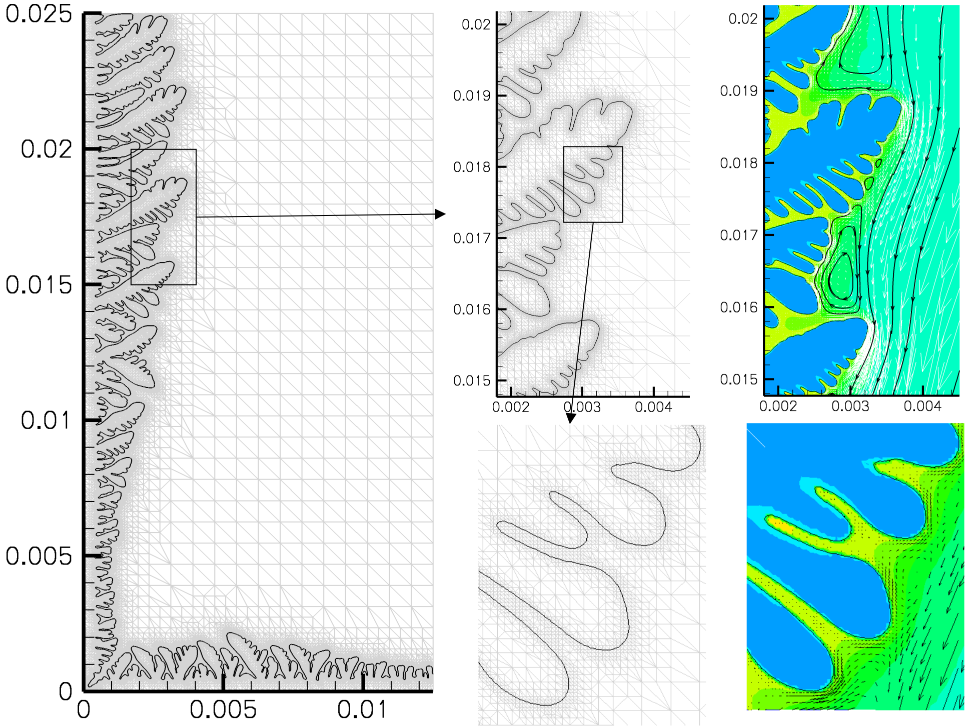


Figure 3.26: Adaptive mesh and interface position. In the right picture, the domain is colored with concentration of Ta for better visibility of the flow field near the dendrite tips.

K), the steady growth tip velocity is as high as 3000 cm/s. With such a high growth velocity, both the thermal and the solutal boundary layers are very thin as demonstrated in Fig. 3.28. The lowest concentration in the crystal is observed along the x, y, z axes, which corresponds to the growth of the primary dendrite tips.

Computation for three-dimensional problems is often very intensive. With 16 nodes (each node with two 2048 Hz CPUs) in the Cornell Theory Center, this computation takes about 12 hours. Using the adaptive meshing technique, the number of nodes and elements are only about 0.5 million and 3.0 million, respectively, at step 211. Without adaptive meshing, 32 million nodes and 192 million elements will be required to give an equivalent resolution. At previous time steps, the speed up of adaptive meshing is more obvious since the crystal is small. A typical adaptive mesh used in the computation is shown in Fig. 3.29.

In order to study the convergence of our numerical algorithm for this problem, we use an adaptive mesh with the same initial grid 1^3 and different refinement levels from 6 to 9 as shown in Fig. 3.30. Results show that at least 8 refinement levels (corresponding to a uniform mesh of 256^3) are required to numerically resolve the secondary dendrite arms.

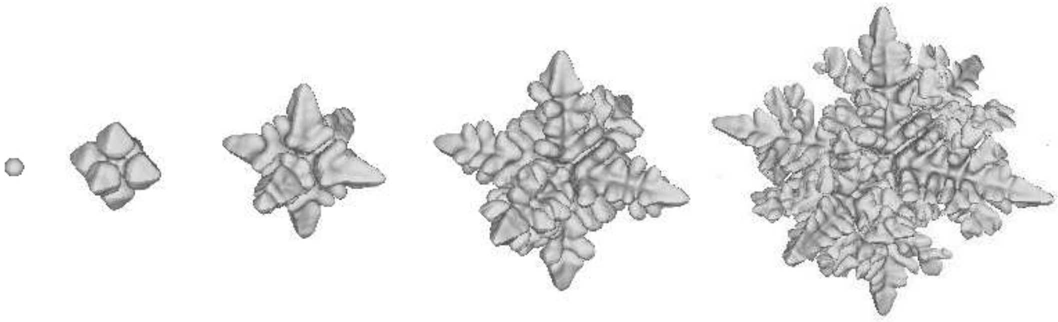


Figure 3.27: Computed Ni-Cu crystal shape at steps 1, 51, 101, 151 and 211.

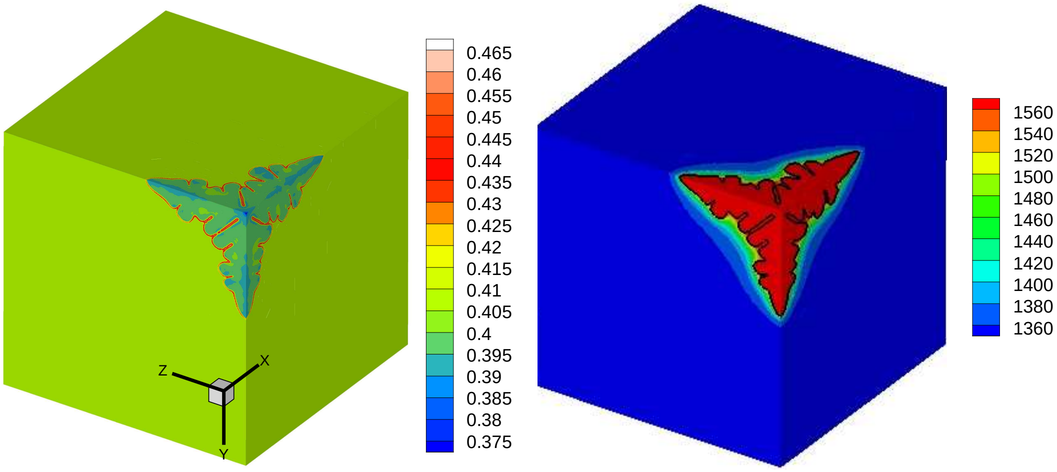


Figure 3.28: Solute and temperature field at step 151 for Ni-Cu crystal growth.

The ability to study three dimensional pure material crystal growth under convection has been demonstrated in our previous work [61]. The three-dimensional free dendritic growth computation results compared very well with other numerical studies using phase field methods [28, 38] at a high undercooling and front tracking techniques [3] at a low undercooling. Here, we present the three-dimensional results of coupling fluid flow for a binary alloy, which has not been previously reported in the literature. The same alloy Ni-Cu is used with the same conditions except that an inlet flow with velocity $0.1 m/s$ is applied at the top of the domain. In this case, we use an initial grid of $1 \times 1 \times 2$ with refinement level 8. This corresponds to a full mesh of $256 \times 256 \times 512$. Only $1/4$ of the total domain is used due to symmetry. Similar to the pure material free dendritic growth case [61, 28, 38], crystal growth at the upstream arm is enhanced. However, unlike the pure material case in which the crystal growth is relatively smooth [61, 28, 38], perturbations quickly develop into secondary dendrite arms as shown in Fig. 3.31.

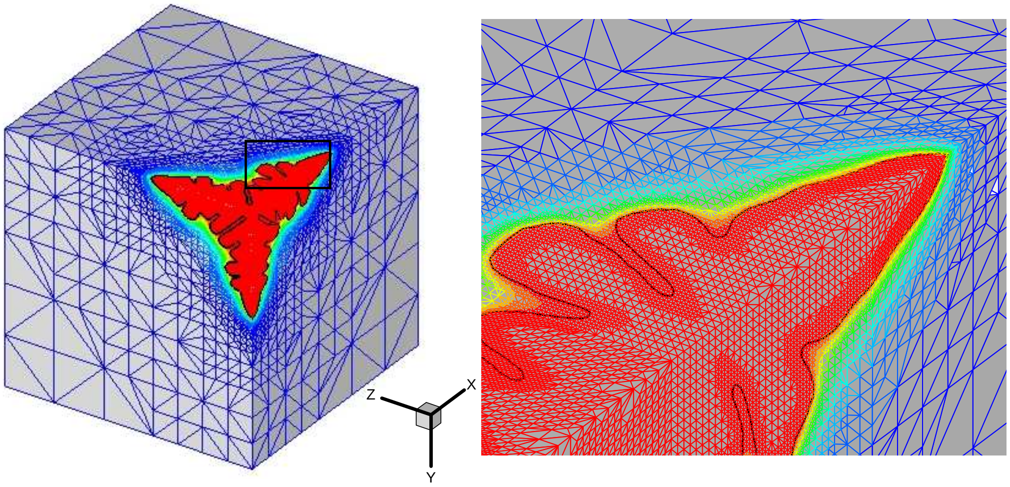


Figure 3.29: Adaptive mesh colored with temperature at step 151 for Ni-Cu crystal growth. The figure on the right shows the magnified mesh near the primary dendrite in the x direction.

3.5 Conclusions

A level set method combining features of front-tracking and fixed domain methods is presented to model dendritic solidification of alloy systems. Some of the key features of

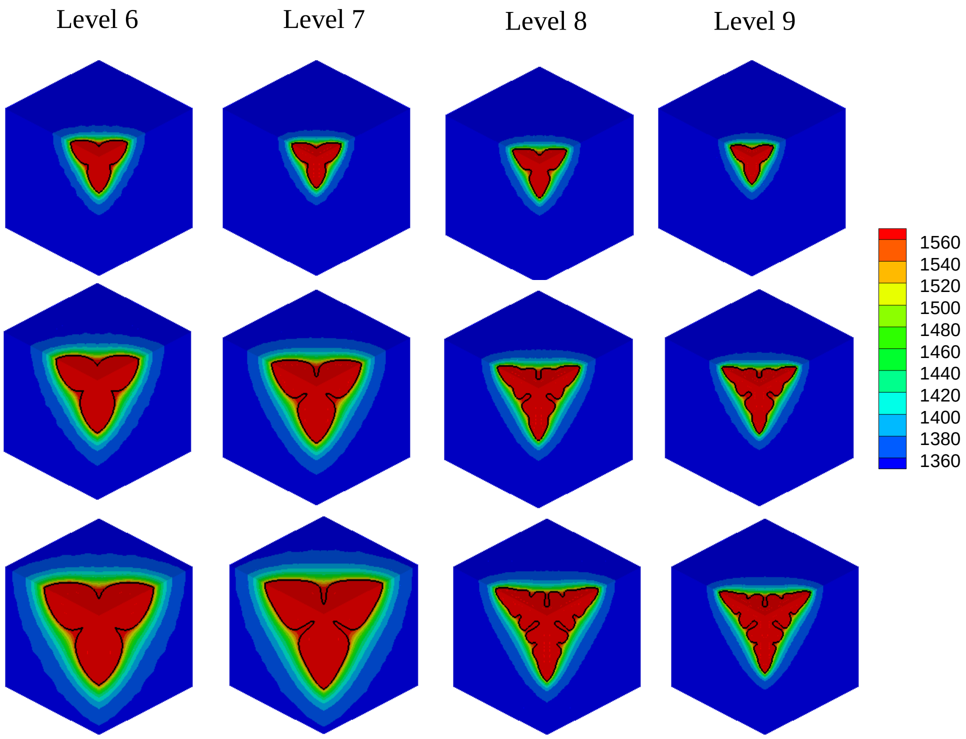


Figure 3.30: Interface position and temperature field at times about $2e-7$s, $7e-7$s and $1.5e-6$s (rows 1, 2 and 3), with the same initial grid 1^3 but different refinement levels from 6 to 9 (columns 1, 2, 3 and 4.)

the presented method include (a) accurate tracking of the interfaces using the level set method (b) heat/mass/momentum transport computation by avoiding direct application of interface conditions (solid/liquid boundaries), (c) 2D/3D finite element computation with fast adaptive meshing, and (d) multiple solid phase capability. The method is substantially simpler to implement relative to front tracking models and can serve as an efficient alternative approach to multi-phase field methods.

The numerical investigations shown here have demonstrated that the present method can serve as an accurate and computationally effective alternative tool for modeling multi-phase multi-component alloys. Current work is in progress to apply this methodology to multi-scale alloy solidification.

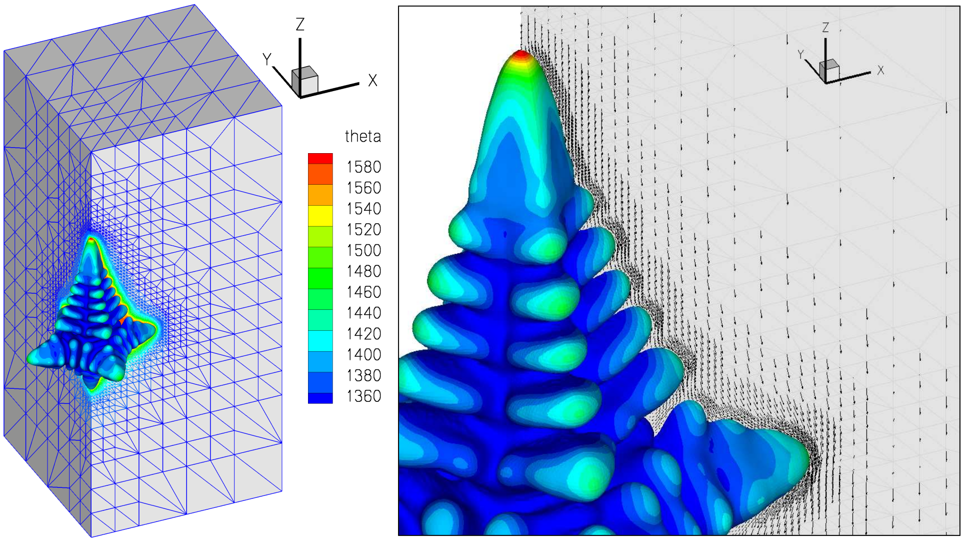


Figure 3.31: Ni-Cu crystal growth with inlet flow from the top. Left: dendrite colored with interface velocity, mesh colored with temperature; right: flow passing by the upstream and perpendicular stream.

Chapter 4

Study of interaction between multiple dendrites

4.1 Mathematical model

To focus on the modeling of nucleation, we take a few simplifications in our model.

1. All material properties are assumed to be constant if not mentioned. These include density ρ, heat capacity c, latent heat L, heat diffusion coefficient k, solute diffusion coefficient D, liquidus slope m_l, and partition coefficient k_p.
2. Fluid flow effects are not considered in this work.
3. Solute diffusion in the solid phase is neglected.

4.1.1 Previous model

In this section, we provide a brief review of our previous model with the mentioned simplifications. For more details of this model with consideration of convection, the interested reader can refer to [61].

The governing equations for modeling of heat transfer and solute transport during solidification are as follows:

$$\rho c \frac{\partial T(\mathbf{x},t)}{\partial t} = k\nabla^2 T(\mathbf{x},t), \quad \mathbf{x} \in \Omega,$$

$$\frac{\partial C^l(\mathbf{x},t)}{\partial t} = D\nabla^2 C^l(\mathbf{x},t), \quad \mathbf{x} \in \Omega^l,$$

where Ω is the total domain including both the liquid part Ω^l and the solid part Ω^s. The above two governing equations for temperature T and solute concentration in liquid C^l are simple diffusion equations and introduce a physical model that has been very well studied. However, due to the existence of the moving interface, Γ^{sl}, the boundary conditions listed below make the problem nontrivial.

Because of phase transformation, solute is rejected from the solid phase to the liquid phase leading to a solute rejection flux at the freezing interface:

$$D\frac{\partial C^l}{\partial \mathbf{n}} = -(1-k_p)C^l V\mathbf{n}, \tag{4.1}$$

where $\mathbf{n}$ is the normal direction of the solid-liquid interface pointing towards the liquid phase.

The temperature at the solid-liquid interface Γ^{sl}, T_I, equals the equilibrium temperature, T_*, given by the Gibbs-Thomson relation:

$$T_I = T_* \equiv T_m + m_l C^l - \epsilon_c \kappa - \epsilon_V V, \tag{4.2}$$

where T_m is the melting temperature of the solidifying material, ϵ_c is the curvature undercooling coefficient, κ is the curvature of the interface, ϵ_V is the kinetic undercooling coefficient, and V is the interface velocity.

The velocity of the solid-liquid interface is governed by the heat flux jump through the classical Stefan equation:

$$V = \frac{q^s - q^l}{\rho L}, \tag{4.3}$$

where q^s and q^l are the heat fluxes at the solid and liquid side of the interface, respectively.

We have introduced two assumptions for applying these boundary conditions indirectly.

1. Solidification occurs in a diffused zone of width $2w$ that is symmetric around the zero level set. A phase volume fraction ϵ can be defined according to the signed distance to the interface, ϕ, which is simply the distance to the interface with '+' sign in the liquid phase and '-' sign in the solid phase:

$$\varepsilon(x,t) = \begin{cases} 0, & \phi(x,t) > w, \\ 1, & \phi(x,t) < w, \\ 0.5 - \phi/(2w), & \phi(x,t) \in [-w, w]. \end{cases}$$

2. The solid-liquid interface temperature, T_I, is allowed to vary from the equilibrium temperature, T_*, in a way governed by

$$\frac{dT_I}{dt} = -k_N(T_I - T_*).$$

   where k_N controls the rate with which T_I is designed to approach the desired equilibrium temperature T_*. In [61], for a given time step Δt, we selected $k_N = 1/\Delta t$ to guarantee that the interface temperature stably converges to the equilibrium temperature. The selection of Δt was based on a CFL condition for the level set function calculation [61].

4.1.2 Nucleation model

Nucleation is a very complicated phenomenon related with the type and amount of impurities in the material. In this work, we assume that the number of impurities per unit volume, i.e. the density of impurities, is ρ^n. Suppose that the domain of interested problem has a volume of V, then there will be $\rho^n V$ potential nucleation sites, since each impurity serves as a potential nucleation site. For each potential nucleation site i, we model three random variables, location x_i, required under-cooling for nucleation ΔT_i^n

and orientation I_i. These random variables are independently sampled with the sampling scheme discussed below.

1. x_i is the location of the potential nucleation site. In [5], the potential nucleation sites are distributed uniformly with a certain spacing. The advantage of modeling uniformly spaced potential nucleation sites is that determination of their locations is very simple. This idea is followed in our computations in Section 4.3.2 in order to allow us to compare our results with those reported in [5]. A big disadvantage of modeling uniformly spaced potential nucleation sites is that it requires a structured mesh with proper grid spacing so that a potential nucleation site coincides with a finite element node. The nature of randomness in the location of potential nucleation sites is also lost by modeling them to be uniformly spaced. To overcome these disadvantages, we assume that x_i is equally probable at each point of the whole domain instead of deterministically at a particular mesh nodal point.

   In our computations with consideration of randomness (Sections 4.3.2 and 4.3.3), we apply the following sampling scheme element by element. For each element e of the initial coarse mesh, we assign a set of potential nucleation sites with the following steps:

   (a) Compute volume of element e, V_e. Since ρ^n is the number of potential nucleation sites in a unit volume, we will have $\rho^n V_e$ potential nucleation sites inside element e. In general, $\rho^n V_e$ will not be an integer. For example, if $\rho^n = 100$, $V_e = 0.023$, then $\rho^n V_e = 2.3$. We want to use a sampling scheme such that the expected number of potential nucleation sites assigned to the element is 2.3.

   (b) Let n_e be the integer part of $\rho^n V_e$ (e.g. when $\rho^n V_e = 2.3$, $n_e = 2$). Sample p from the uniform distribution with range $[0, 1]$. If $p < \rho^n V_e - n_e$ (i.e.

$p < 0.3$ in our example), then set $n_e = n_e + 1$. This step guarantees that the expected number of potential nucleation sites inside element e is $\rho^n V_e$. For the example of $\rho^n V_e = 2.3$, with this step, the probability of $n_e = 2$ is 0.7, while the probability of $n_e = 3$ is 0.3. So the expected number of potential nucleation sites in the element becomes $< n_e >= 0.7 * 2 + 0.3 * 3 = 2.3$, which is the value of $\rho^n V_e$.

(c) For each potential nucleation site $i = 1, 2, \ldots, n_e$, sample its location x_i (uniformly distributed in element e), required undercooling ΔT_i^n ($N(\mu, \sigma^2)$), and orientation angle I_i (uniformly distributed in $[0, 2\pi]$).

After an element is refined into a few 'child' elements, the potential nucleation sites assigned to it are further assigned to its 'child' elements according to whether the location of the potential nucleation site falls inside the 'child' element. In this way, we only need to go through all elements to determine which potential nucleation sites become seeds for crystal growth.

2. ΔT_i^n is the required under-cooling for the potential nucleation site i to nucleate and become a crystal seed. Only if the under-cooling at the potential nucleation site i is greater than ΔT_i^n, site i becomes a crystal seed. In our numerical example in Section 4.3.2, the required under-cooling for nucleation ΔT_i^n is assumed to be at a fixed value of $8K$ following [5]. In Section 4.3.3, we assume that ΔT_i^n follows a normal distribution with mean μ and variance σ^2.

3. The other random variable, I_i, is the orientation angle at potential nucleation site i. The orientation angle is the preferred crystal growth direction. In two dimensions, the preferred crystal growth direction can be mapped to a point on the circumference of a unit circle. Any value between 0 and 2π corresponds to a preferred crystal growth direction. In other words, we can use one real number, I, to represent the preferred crystal growth direction. In three-dimensional growth, the

> preferred crystal growth direction can be mapped to a point on a unit sphere. Two angles of spherical coordinate system (zenith angle and azimuth angle) are often used to represent a point on a unit sphere. So for extension in 3D, the orientation angle I is taken as a vector with two components, one for zenith angle (angle from the positive z-axis) and one for azimuth angle (angle from the positive x-axis). In the numerical examples in Section 4.3.2, we only considered the randomness of orientation angle in two dimensions. I_i is sampled from a uniform distribution from 0 to 2π. After nucleation at location of potential nucleation site i, the crystal may in general rotate as it growths (e.g. as a result of convection). This leads to change in orientation angle for the crystal. This movement of crystal is not considered in this work. In other words, the orientation of the crystal, I, is assumed to be fixed at the value when it is nucleated (i.e. I_i if it is nucleated from potential nucleation site i) during growth the crystal.

In our implementation, we maintain a link list to contain information about these potential nucleation sites including x_i, ΔT_i^n and I_i for each element. For determining whether nucleation occurs at location of a potential nucleation site or not, we use data at nodes of the element to interpolate data at location of the potential nucleation site. After temperature T_i and concentration C_i is obtained at location of a potential nucleation site, x_i, using interpolation, we check whether $T_i \leq T_m + m_l C_i - \Delta T_i^n$. If so, a small crystal seed will be put at location x_i.

4.1.3 Growth model

The velocity at the phase boundary of each crystal, V, is governed by the Stefan equation. Notice that for numerical convenience and making the scheme energy conserving, we have used an interface energy balance that is different from the Stefan equation [61],

$$V = \frac{q^s - q^l}{\rho L} + \frac{2cw}{L} k_N (T_* - T_I). \tag{4.4}$$

The converged solution using this formula has been proved to be the same as using the Stefan equation [61].

In our previous model, we have assumed a 4-fold symmetry of the crystal structure. The Gibbs-Thomson relation coefficient ϵ_c is modeled as

$$\epsilon_c = d_0\{1 - 15\epsilon \cos[4\alpha(\mathbf{n})]\}, \tag{4.5}$$

where d_0 is the capillary length, ϵ is a coefficient describing the surface anisotropy extent, and $\alpha(\mathbf{n})$ is the angle from the positive x axis to the normal direction $\mathbf{n}$. With the above formulation for the Gibbs-Thomson relation coefficient ϵ_c, the crystal grows fastest at directions with angles 0^o, 90^o, 180^o and 270^o to the x axis. In this work, we want to model the growth of multiple dendrites. Each dendrite is growing with different orientation angle I fastest at directions with angles $0^o + I$, $90^o + I$, $180^o + I$ and $270^o + I$ to the x axis. So the same 4-fold symmetry of the crystal structure is used, except with a '$-I$' term in the cos function, as shown below:

$$\epsilon_c = d_0\{1 - 15\epsilon \cos[4(\alpha(\mathbf{n}) - I)]\}. \tag{4.6}$$

Since I varies for different crystals but is the same for the same crystal, different crystals will grow with different preferred orientations.

4.2 Numerical techniques

Comparing with single crystal growth, the additional difficulty of multiple crystal growth is to track multiple interfaces. We have implemented two numerical methods:

(1) the multi-phase level set method by solving multiple signed distance functions, and (2) the level set method by solving a single signed distance function with the aid of markers. Emphasis of this work is on the second method due to its high efficiency and accuracy.

4.2.1 Multiple signed distance functions

For simulating the growth of multiple crystals, one way is to use multi-phase level set method with a signed distance function for each crystal. In the implementation of this method, a new level set solver is created to handle the evolution of a crystal when a potential nucleation site nucleates. Each level set solver is assigned with the crystal orientation I of the potential nucleation site at the time it is created. The following are the main steps of using this method at each time level:

1. Compute under-cooling at each potential nucleation site (interpolation will be required), and create new level set solvers to handle the newly created crystals if required undercooling is satisfied at the location of potential nucleation site.
2. Compute the interface velocity on the interface of each crystal with information of crystal orientation I, and use the computed interface velocity to evolve the signed distance function of the corresponding crystal.
3. Perform re-initialization of all signed distance functions.
4. Solve for temperature and concentration fields using volume averaging techniques.
5. Return to step 2 until convergence ($\|T_n^k - T_n^{k-1}\| \leq 10^{-3} \times \|T_n^k - T_{n-1}\|$) is achieved for this time level. Here n is the current time level, $n-1$ is the previous time level and k is the iteration level.

Details of this method are provided in [62], where it is applied to study the growth of multiple phases instead of multiple crystals. More theoretical analysis of this multi-phase level set method can be found at [77, 39]. In this work, this multi-phase level set method is not used as the main investigation tool but as a verification tool for the other method discussed next, because the multi-phase level set method requires the expensive solution of multiple level set equations. Solving multiple level set equations is only realistic for a small number of crystals. As the number of crystals increases, the multi-phase level set method becomes increasingly inefficient. If tens or even hundreds of crystals are present, this numerical method is impractical, since one cannot afford or does not want to store tens or hundreds of signed distance functions and solve tens or hundreds of level set equations.

4.2.2 Single signed distance function with markers

In this work, we do not model the evolution of the solid-solid phase transformation. Therefore, the solid/solid (crystal/crystal) interfaces are of no significant importance and are assumed to be static. The type of interface of great interest is the solid/liquid (crystal/liquid) interface. We can use a single signed distance function to implicitly represent this interface. However, we also want to identify different crystals, because each crystal grows with different preferred orientations due to the underlying crystal structure. So we introduce another scalar (markers) to represent different crystals. The value of the markers is just the orientation of each crystal. As demonstrated in Fig. 4.1, different markers (shown with different colors) are used to identify different crystals. Although only one signed distance function is used, the growth of a crystal at its own preferred direction can be captured with the aid of markers. The problem of evolving multiple crystal interfaces is reduced to tracking one level set variable (signed distance function) and determining the marker of a newly solidified node point. Tracking a single level set variable is implemented by solving the level set equation as before, while determining the marker of a newly solidified node point is implemented by using an algorithm updated from fast marching as discussed below.

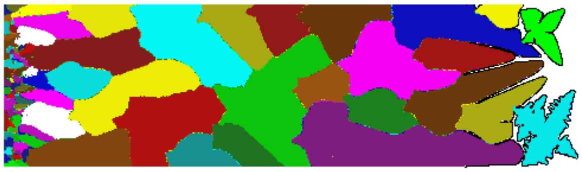


Figure 4.1: Example of using a single signed distance function with markers.

The marker (orientation of the crystal, I) has physical meaning only in the solid phase. However, doing evaluation of the interface velocity requires that I is also defined in the liquid phase. In this work, we define I in the liquid phase as the orientation of the nearest crystal. With this definition, extension of I from the solid phase to the liquid phase can be implemented using the fast marching technique. We have a balanced heap data structure for easy tracking of the next closest nodal point to the solid-liquid interface. One important step of the fast marching algorithm is to extend values in an element (say A-B-C in the 2D case). Let values on two nodes (say A and B) be known, with the value on the other node (C) being unknown. In extending the interface velocity from nodes A and B to node C, interpolation is reasonable since one is interested in obtaining a smooth velocity field over the whole domain. However, for extending the crystal orientation angle from nodes A and B to node C, interpolation will not make much sense since node C belongs to either the same crystal as node A or the same crystal as node B. The following formula is thus used for extension of the orientation angle from nodes A and B to node C:

$$I_C = \begin{cases} I_A, & if\ \|AC\| \leq \|BC\|, \\ I_B, & if\ \|BC\| < \|AC\|. \end{cases} \tag{4.7}$$

Using this method, we can efficiently extend I from the solid phase to the liquid phase as demonstrated in Fig. 4.2. Notice that in Fig. 4.2, we have only extended I from the solid phase to a part of the liquid phase with $\phi < 3w$, which is far enough for computing the interface velocity. A newly solidified nodal point definitely falls in the region with $\phi < 3w$, since the CFL coefficient is selected to be less than 1. Therefore, the orientation angle I has already been extended to the newly solidified node.

Figure 4.2: Example of extending the orientation angle in the liquid phase. The orientation angle is extended to the liquid phase with $\phi < 3w$. The artifacts of interpolation lead to a color different from all nearby colors at some places of the mesh. The plotting software we utilized, Tecplot [78], automatically interpolates the orientation and plots the color corresponding to the interpolated value.

The level set method uses a field (signed distance) to represent the interface. Suppose a new crystal is introduced into the solidification system by nucleation at location $\mathbf{x}_i$. Since the solid-liquid interface changes, the signed distance field needs to be updated with the following operation:

$$\phi(\mathbf{y}) \leftarrow \min(\phi^0(\mathbf{y}), \|\mathbf{x_i} - \mathbf{y}\| - R_0), \ \forall \ \mathbf{y} \ \in \ \Omega, \tag{4.8}$$

where ϕ^0 is the signed distance function before the potential nucleation site is nucleated at $\mathbf{x}_i$, R_0 is the size of the initial crystal seed at location $\mathbf{x}_i$ and $\mathbf{y}$ is the location of a node (see Fig. 4.3 for a related schematic). Notice that, we need to apply this operation on all nodes whenever there is a newly nucleated crystal. Fortunately, the number of operations is strictly proportional to the number of nodes. It is also very easy to parallelize, since no communication (except information of the newly created crystal) is required.

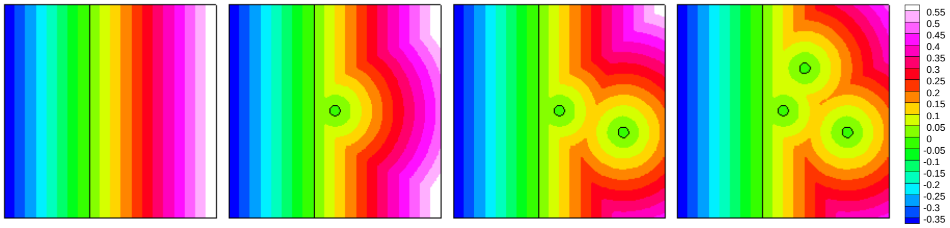


Figure 4.3: Redefinition of the level set function when nucleation occurs in the domain $[0, 1] \times [0, 1]$: The initial level set field $\phi^0 = x - 0.4$ represents an interface at $x = 0.4$ (dark line). If nucleation occurs at $(0.5, 0.5)$ with initial seed radius 0.025, we redefine the level set field to be $\phi^1 = \min(\phi^0, \sqrt{(x-0.5)^2 + (y-0.5)^2} - 0.025)$. The contour value on the 2nd figure is the value of ϕ^1 and the dark line shows the zero level set of ϕ^1 that appropriately captures the changes on the interface introduced by nucleation at $(0.5, 0.5)$. The remaining two figures demonstrate the ϕ^2, ϕ^3 fields when additional nucleation occurs at the locations $(0.8, 0.4)$ and $(0.6, 0.7)$.

The following are the main steps of using this method at each time level:

1. Compute the under-cooling at each potential nucleation site (interpolation will be required). For each newly nucleated crystal at location $\mathbf{x_i}$ apply the following operation on each node $\mathbf{y}$:

$$\phi(\mathbf{y}) \leftarrow \min(\phi^0(\mathbf{y}), \|\mathbf{x_i} - \mathbf{y}\| - R_0).$$

2. Extend the orientation angle from the solid region to part of the liquid region with $\phi < 3w$ using the fast marching algorithm.

3. Compute the interface velocity on the solid-liquid interface with information of orientation angle I, and use the computed interface velocity to evolve the signed distance function. Note that I plays a role in the computation of T_*, which plays a role in the computation of V using the extended Stefan equation.
4. Perform re-initialization of the single signed distance function.
5. Solve for the temperature and concentration fields using volume averaging techniques.
6. Return to step 3 until convergence is achieved for this time level.

Comparing the above method with the multi-phase level set method, multiple signed distance functions are reduced to just one signed distance function and an orientation angle. Moreover, we do not need to solve the multiple level set equations. So this method is much more efficient than the multi-phase level set method. However, as shown in Eq. (4.7), the extension of the orientation angle is not smooth at liquid nodes, which are close to multiple crystals. Fortunately, for most extension cases of the orientation angle from values at nodes A and B, $I_A = I_B$. In other words, $I_A \neq I_B$ occurs only at locations close to multiple crystals. Note that the interface velocity is only computed at nodes adjacent to zero level set (i.e. nodes of elements cut by the zero level set). At other nodes, the interface velocity is extended using fast marching. So assuming that I is correct for the nodes of the elements cut by the zero level set, the above approximation will not make any difference in the calculation of the interface velocity. This means that I_C will make a difference in the interface velocity only when $I_A \neq I_B$ and C is a node adjacent to zero level set, which means that two crystals are already very close to each other. We will demonstrate that the difference between the results obtained with this method and the results obtained from the multi-phase level set method is actually negligible.

4.2.3 Other techniques to speed up computation

In this work, the finite element method is used to solve for the temperature, solute concentration and level set function with implementation details discussed in [73, 62]. Comparing with the combined finite element method (for governing equations of temperature and solute) and finite difference method (for level set computation) [61], there is flexibility in the current implementation to use adaptively refined unstructured mesh, which is shown to be much more efficient than a uniform mesh in [62]. Since the level set variable gives information about how far away each node or element is from the interface, a refinement criterion based on the level set variable [62] is very convenient for generation of adaptive mesh locally refined near the solid-liquid interface. The effect of adaptive meshing with one level of refinement near the interface can be seen in Fig. 4.2. Significantly higher levels of refinement are usually used in the numerical examples discussed later. Other than adaptive meshing, domain decomposition (implemented with the aid of graph partition package ParMetis [79]) is also used for parallel computing to allow investigation of interaction between hundreds of crystals. Fast marching technique [73] is utilized to perform re-initialization of the level set variable and extension of the interface velocity. For solving the linear systems in the finite element method, we use the Krylov subspace method in the matrix solver package PETSc [80].

4.3 Numerical examples

4.3.1 Interaction between 9 crystals

In this example, we consider the interaction between 9 crystals with orientations $I = 0^o, 10^o, 20^o, 30^o, 40^o, 50^o, 60^o, 70^o, 80^o$ as shown in Fig. 4.4. The domain size of interest is $[-1200, 1200] \times [-1200, 1200]$. The 9 crystals are uniformly spaced with distance Δ. The initial shape of each crystal is circular with radius 30. A pure material is considered for this example with all parameters including ρ, c, k, L normalized to 1. The temperature on the interface satisfies the following Gibbs-Thomson relation: $T_* = -d_0 \{1 - 15\epsilon \cos[4(\theta - I)]\}\,\kappa$ with $d_0 = 0.5$ and $\epsilon = 0.05$. Initially, the domain is undercooled at temperature -0.55, while inside the 9 initial crystals, the initial temperature is taken as 0. The boundary of the domain is kept at temperature -0.55. We want to study how the 9 crystals interact with each other during their growth.

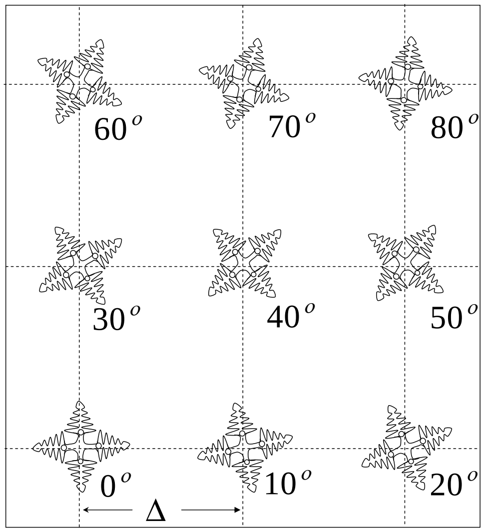


Figure 4.4: Schematic of the growth of 9 crystals.

Effects of crystal spacing

In this computation, we use an adaptive mesh equivalent to a uniform mesh of grid spacing 1. Here equivalent means that the smallest grid spacing in the adaptive mesh is equal to the grid spacing in the uniform mesh. We use the term 'equivalent', because both meshes lead to the same numerical solution as was demonstrated in our earlier work [62]. The CFL coefficient for adaptive time stepping is selected to be $\frac{1}{3}$. When the spacing between the crystals is 800, i.e. $\Delta = 800$, the interaction between the crystals is weak as shown in Fig. 4.5. The closest distance between crystals is the distance between the right dendrite tip of the crystal with $I = 0^o$ and the left dendrite tip of the crystal with $I = 10^o$. The distance between any other two crystals is greater than the thermal boundary layer size. Therefore, each crystal grows almost independently, except that the thermal boundary layers of the two crystals $I = 0^o$ and $I = 10^o$ overlap at places around their closest tips. The independence and similarity of the thermal boundary layers for the 9 crystals actually demonstrate that the computed solution is independent of mesh orientation. When the spacing between the crystals reduces to 600 or 400, the interaction between the crystals is very obvious as shown in Fig. 4.5. However, in all cases, we observe that the shapes of all 'free' dendrite tips are very similar. Here 'free' means that the dendrite tip is far away from other dendrites. More interestingly, the solution (interface position and temperature field) keeps partial symmetry in all three cases with different crystal spacing. For example, the two crystals with $I = 30^o$ and $I = 60^o$ are symmetrical to each other at places close to their center line $y = \frac{\Delta}{2}$. Similarly, the two crystals with $I = 40^o$ and $I = 50^o$ are symmetrical to each other at places close to their center line $x = \frac{\Delta}{2}$. This partial symmetry comes from the symmetry in the crystal orientations as shown in Fig. 4.4.

Comparison with the multi-phase level set method

In the above computation of studying crystal spacing effects, we have used only one signed distance function and 9 markers to identify the interface of the 9 crystals. It is more common to use multiple signed distance functions for tracking multiple interfaces with the level set method [77, 39]. Here we study the interface position using both methods with grid spacing 1 and CFL coefficient $\frac{1}{3}$. As shown in Fig. 4.6, both methods predict almost the same interface position when the dendrite tips are close to the domain boundary. This computation gives us confidence on the accuracy of using only one signed distance function with markers, which is significantly much more efficient than using multiple signed distance functions, since only one level set equation needs to be solved. In all the other computations in this work, one signed distance function with markers will be used.

Convergence study

In the previous computations of studying crystal spacing effects, we have used an adaptive mesh (initial coarse mesh 75×75, refinement level 5) equivalent to a uniform mesh of grid spacing 1. However, in [48, 32], a grid spacing of 0.4 was used for a similar problem with just one crystal. In order to demonstrate that grid spacing 1 already leads to a converged solution, we also computed the numerical solution for crystal spacing 600 using various grid spacings $1, 2, 4, 8, 16, 32$ (corresponding to refinement levels 5, 4, 3, 2, 1 and 0), as demonstrated in Fig. 4.7. Notice that in the studies here, adaptive meshing is used. Thus the grid spacing actually refers to the smallest grid spacing of the adaptive mesh. To use roughly the same time discretization, the CFL coefficient for adaptive time stepping is selected to be $\frac{1}{3}, \frac{1}{6}, \frac{1}{12}, \frac{1}{24}, \frac{1}{48}, \frac{1}{96}$ corresponding to the selection of grid spacings $1, 2, 4, 8, 16, 32$, respectively. As the grid spacing decreases, the computed dendrites become slimmer. To quantitatively study convergence, we define the error as the root mean square (rms) of the difference in the temperature and the "true" temperature (obtained with refinement level 5) at time 1.05×10^4 on nodes of a 75×75 uniform mesh, i.e. $err = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (\frac{T_i - T_i^{true}}{0.55})^2}$. Here n is the total number of nodes of the uniform mesh with value $76 \times 76 = 5776$, T_i is the temperature at node i at time 1.05×10^4, and T_i^{true} is the "true" temperature at node i at time 1.05×10^4. The relation between error and refinement level of the adaptive mesh is given in Fig. 4.9. The computed dendrites with grid spacing 32 are significantly different (much 'fatter') than the converged solution with the finest grid spacing. In fact, since grid spacing 32 is even larger than the initial crystal radius, it is expected that dendrites computed with grid spacing 32 are much 'fatter' than the converged solution. However, the solution with grid spacing 16 can still give us an insight on the interaction between the dendrites. The fact that grid spacing can be 40 times larger than the grid spacing used in the literature [48, 32] ($16 = 40 \times 0.4$) and still able to provide reasonable solution suggests that the present technique has significant potential for the study of the interaction between many dendrites. Notice that grid spacing 40 times larger means that the element size is 40^2 times larger and the number of elements just $\frac{1}{40^2}$ times the original required number of elements. This would lead to a significant reduction of computational requirement in terms of memory and time. In 3D, this advantage will be even more obvious, since the number of elements can be reduced to $\frac{1}{40^3}$ of the required number of elements using grid spacing 0.4. We will demonstrate the interaction of multiple 3D dendrites in Section 4.3.2.

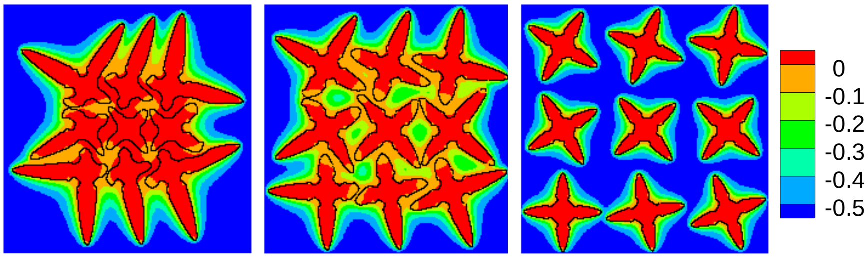


Figure 4.5: Temperature field for the interaction between nine crystals with spacing 400, 600 and 800.

The main reasons that the present method converges nicely are (1) that the interface is tracked with a level set which could give accurate normal direction and interface curvature for accurate determination of the equilibrium temperature using the Gibbs-Thomson relation, and (2) that energy is numerically conserved since no essential boundary condition is applied on the interface, which could compensate for the numerical error in the interface velocity evaluation.

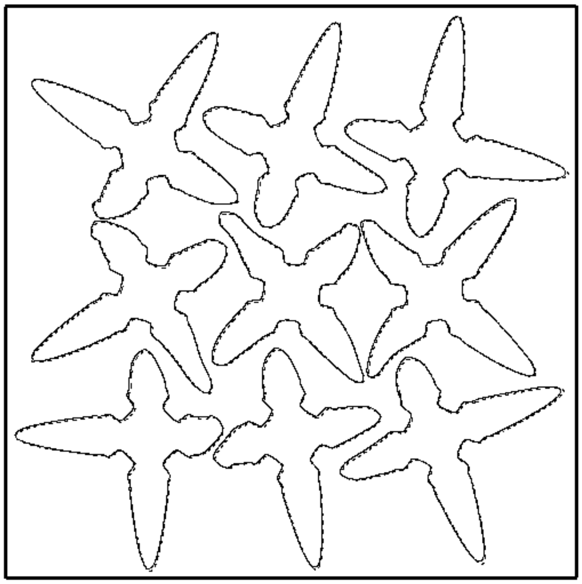
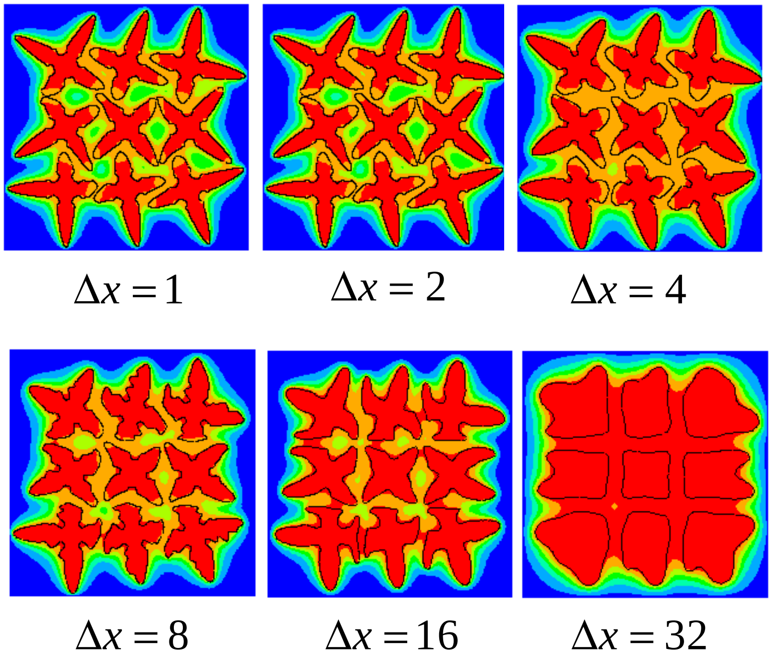


Figure 4.6: Interface position when the dendrite tips are close to the domain boundary. Solid line: Computed interface position using single signed distance function with markers; Dash dot line: Computed interface position using multiple signed distance functions.

4.3.2 Columnar-to-equiaxed transition (CET)

Columnar to equiaxed transition is a phenomenon of great importance in casting. For decades, researchers have relied on the cellular automata method [20, 33, 51, 58] to give numerical estimation of the microstructure type (columnar or equiaxed) at different solidification speeds and thermal gradients. Arguing that cellular automata method has issues of mesh dependence and lack of demonstration for its convergence, reference [5] provides the first CET study using the phase field method. In this section, we will first study the same numerical example as in [5] as a validation of our numerical method. We will compare our numerical results of columnar-to-equiaxed transition (CET) with


$\Delta x = 1$ $\Delta x = 2$ $\Delta x = 4$

$\Delta x = 8$ $\Delta x = 16$ $\Delta x = 32$

Figure 4.7: Interaction between 9 crystals with spacing 600 using various grid spacings $1, 2, 4, 8, 16$ and 32.

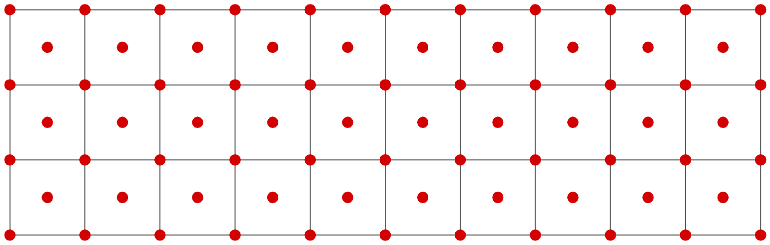


Figure 4.8: Schematic of the computational domain and potential nucleation sites for the two-dimensional crystal growth.

an analytical estimation and numerical results using a phase field method in [5]. Independence of the results on the numerical grid size is not demonstrated in the phase field study of [5].

Here we will first perform a convergence study for one typical case to determine the appropriate grid spacing for numerical simulation (Section 4.3.2). The investigation of CET for different process conditions is then reported in Section 4.3.2. Since crys-

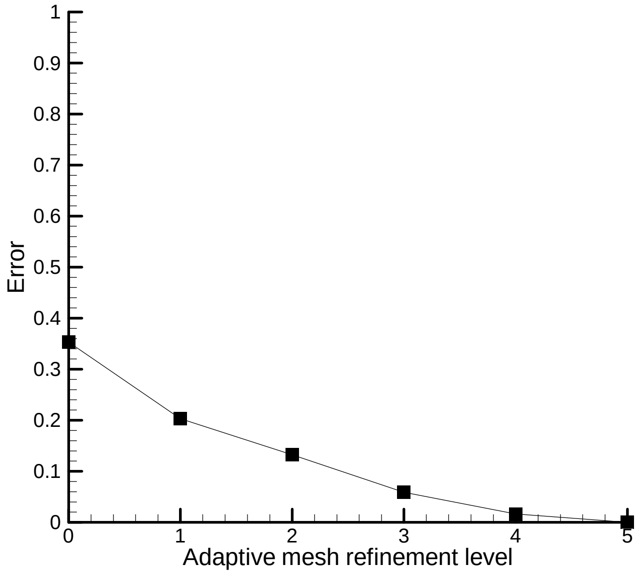


Figure 4.9: Error at refinement levels $0, 1, 2, 3, 4$ and 5 (corresponding to grid spacings $32, 16, 8, 4, 2$ and 1).

tal growth in real world is three-dimensional growth, we will consider the interaction between multiple crystals in 3D (Section 4.3.2). We will also incorporate many more crystals by using a much larger computational domain to study how the randomness of seed orientation affects the microstructure pattern (Section 4.3.2).

Convergence study

The important material properties considered are taken from [5] and are given as follows: initial concentration $C_0 = 3wt.\%$, partition coefficient $k_p = 0.14$, liquidus slope $m_l = -2.6K/wt.\%$, solute diffusion coefficient $D_l = 3000\mu m^2 s$, melting temperature $T_m = 933.47K$, required undercooling for nucleation $\Delta T^n = 8$ K, Gibbs-Thomson relation coefficient $\epsilon_c = 0.24K \cdot \mu m$, surface tension anisotropy $\epsilon = 0.01$ and domain height $3\sqrt{2}d_s$ with $d_s = 9.7\mu m$. The spacing between two potential nucleation sites is $\sqrt{2} \times d_s$, so that the density of potential nucleation sites is d_s^{-2} with a body centered structure as shown in Fig. 4.8. In [5], 23 cases with various solidification speeds and thermal gradients are considered using a mesh with 151 grid points in the y direction by fully-exploiting symmetry. The uniform mesh used in [5] is approximately equivalent to the adaptive mesh used in this work with refinement level 10 and with an initial coarse mesh 20×1. In our computation, we use a coarser adaptive mesh with refinement level 9 (smallest grid spacing is $0.08\mu m$). In order to demonstrate that refinement level 9 is actually fine enough for this problem, we performed a convergence study using refinement levels $5, 6, 7, 8, 9$ and 10 (discretization in space) for solidification speed $3000\mu m/s$ and thermal gradient $1400K/cm$. For time discretization, the CFL coefficient used for adaptive time stepping is $1/3$ for all computations in this example. To investigate the convergence quantitatively, we defined the error as the root mean square of the difference in the concentration and the "true" concentration (concentration obtained using refinement level 10) at time $3.47 \times 10^{-2}s$ on nodes of a 640×32 uniform mesh, i.e. $err = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\frac{C_i - C_i^{true}}{3.0})^2}$. As shown in Fig. 4.10, the nucleation pattern for refinement level 5 is significantly different from the results obtained with refinement level 10, which leads to a very large error (around 0.84 as shown in Fig. 4.11). When the refinement level is increased to 7, 8 and 9, the crystal shapes are visually very similar to the crystal shape obtained with refinement level 10. Since the error when using refinement level 9 is only about 0.04, we can assume that refinement level 9 is fine enough. For all two-dimensional computations in this problem, we use refinement level 9.

Investigation of CET

Here we perform the same 23 cases examined in [5] with solidification speeds and thermal gradients listed in Table 4.1. The computed results for the 23 cases are shown in Fig. 4.12. The growth types of the 23 cases (columnar, equiaxed and mixed) are plotted in Fig. 4.13 together with the analytical estimation of CET using Hunt's model [24] with $G = 0.666\sqrt{\rho^n}\Delta T^t(1 - (\frac{\Delta T^n}{\Delta T^t})^3)$ and $\Delta T^t = 216V^{1/2}$ [5], where G and V are the thermal gradient and pulling velocity for directional growth, respectively, ΔT^n is the required undercooling for nucleation and ΔT^t is the undercooling at the dendrite tip. In the model discussed in [5], only one phase field equation and one solute diffusion equation are solved. The temperature is not solved from the governing heat equation. Here, we take the same simplification. Initially the temperature is distributed linearly with a given thermal gradient G. The evolution of the temperature is then computed with a constant cooling rate, R, which is obtained from the given pulling velocity and thermal gradient through $R = V \times G$. Just as predicted by Hunt's model, the columnar growth is favored for low pulling velocity (solidification speed) and high temperature gradient, while the equiaxed growth is favored for high pulling velocity and low temperature gradient. Mixed growth (a transition microstructure from columnar to equiaxed) is observed for cases below but close to the analytical estimation. Notice that in these computational example, the temperature gradient considered are unrealistically large in order to observe and study the CET transition.

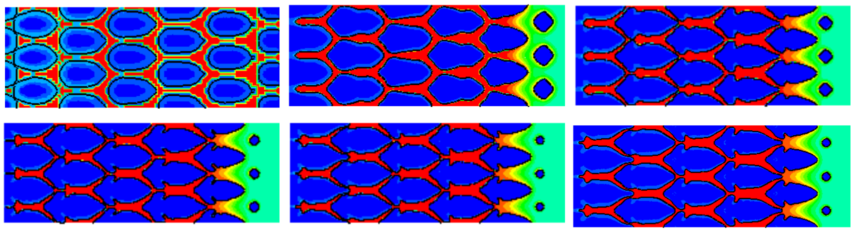


Figure 4.10: Solute concentration for solidification speed $3000\mu m/s$ and thermal gradient $1400K/cm$ using adaptive meshing with refinement from 5 to 10.

Comparing with the results in [5], we obtain similar growth pattern. However, we did not observe secondary dendrite formation. Even with the finest mesh in the convergence study as shown in Fig. 4.10, secondary dendrites do not show up in the nu-

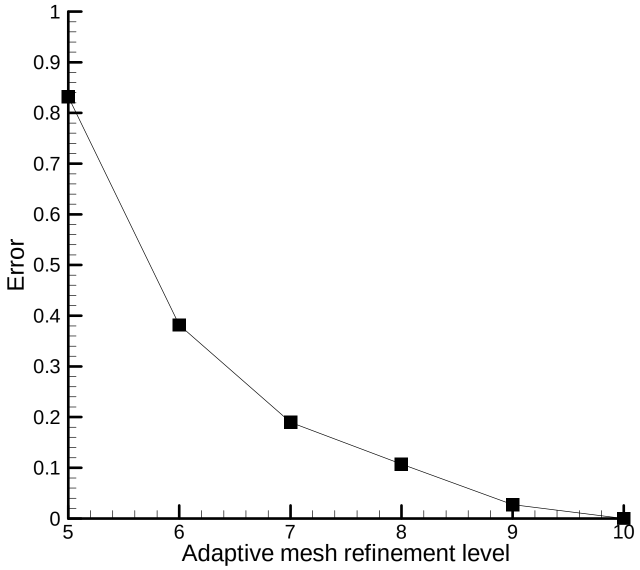


Figure 4.11: Error at refinement level from 5 to 10 for solidification speed $3000 \mu m/s$ and thermal gradient $1400 K/cm$.

merical solution. Since convergence of the numerical solution is observed in Fig. 4.10, we believe that the specified material properties will not lead to significant secondary dendrites using our mathematical model. In order to demonstrate the ability to capture secondary dendrites using the present level set method, we reduce the magnitude of Gibbs-Thomson relation coefficient ϵ_c from $0.24 K \cdot \mu m$ to $0.01 K \cdot \mu m$ with corresponding results shown in Fig. 4.14. The growth type of the 23 cases (columnar, equiaxed and mixed) are plotted in Fig. 4.15 together with the analytical estimation using Hunt's model. A comparison for case 2 is given in Fig. 4.16 to demonstrate the effects of ϵ_c on the formation of secondary dendrites. Comparing with the results shown in Figs. 4.12 and 4.13 with larger magnitude of the Gibbs-Thomson relation coefficient ϵ_c, we observe that the CET transition occurs at a smaller thermal gradient for a giving solidification speed.

Without utilizing symmetry to reduce the computation requirement as was the case in [5], we observe a continuous transition from equiaxed growth to columnar growth

Table 4.1: Considered solidification speed ($\mu m/s$) and thermal gradient (K/cm)

#	1	2	3	4	5	6	7	8
V	3000	3000	3000	3000	3000	3000	3000	3000
G	140	1400	2800	5600	7000	8400	9800	21000

#	9	10	11	12	13	14	15	16
V	6000	6000	6000	6000	10000	10000	10000	10000
G	7000	9800	21000	39200	7000	15400	21000	39200

#	17	18	19	20	21	22	23
V	10000	1000	1500	1500	2500	3500	4500
G	56000	140	140	7000	7000	7000	7000

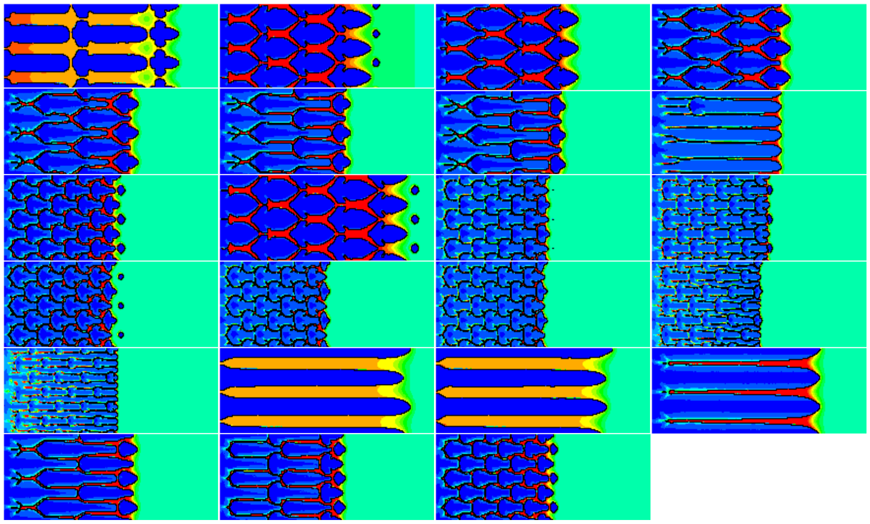


Figure 4.12: Computation results showing solute concentration for the 23 cases listed in Table 4.1 (First row: case $1, 2, 3, 4$; second row: $5, 6, 7, 8$ and so on). Gibbs-Thomson coefficient $\epsilon_c = 0.24 K \cdot \mu m$.

as shown in case 8 in Fig. 4.12, and cases 4, 12 in Fig. 4.14. If symmetry is utilized, the computation requirement is much less (only $1/6$) than that without using symmetry. However, if the solution symmetry is inherently different from the applied symmetry, we may obtain very different results as shown in Fig. 4.17. For example, the compu-

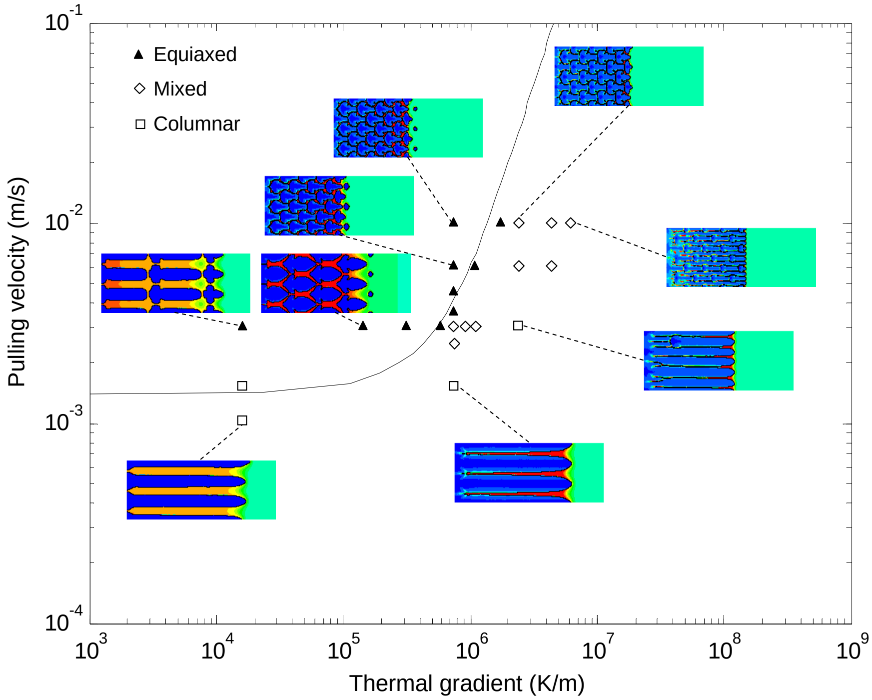


Figure 4.13: CET map for directional solidification for the 23 cases listed in Table 4.1. Gibbs-Thomson coefficient $\epsilon_c = 0.24K \cdot \mu m$.

tation domain width may not be an integer times of the dendritic growth wavelength, which is related with processing and material parameters as analytically estimated using $\lambda = 2\pi(\frac{D\Gamma}{-Vm_lC_0})^{1/2}$ [34]. As shown in the right of Fig. 4.17, the domain boundary restricts the dendrites from adjusting their wavelength to an optimal value. This leads to dendrite tips with slightly different shapes. However, if a larger domain is used, the dendrites can gradually adjust their wavelength leading to dendritic tips with the same shape. More importantly, the wavelength of dendritic growth is significantly smaller than the wavelength obtained without utilization of symmetry. Therefore, for more accurate solution, a larger computational domain is preferred as it can reduce the effects of the artificial computational domain boundary on the numerical solution. However, the computation time will also increase significantly with an enlarged domain. For example, the results obtained using symmetry shown in the right of Fig. 4.17 take around 45 minutes using one computation node with two 3.8G CPUs, while the results obtained

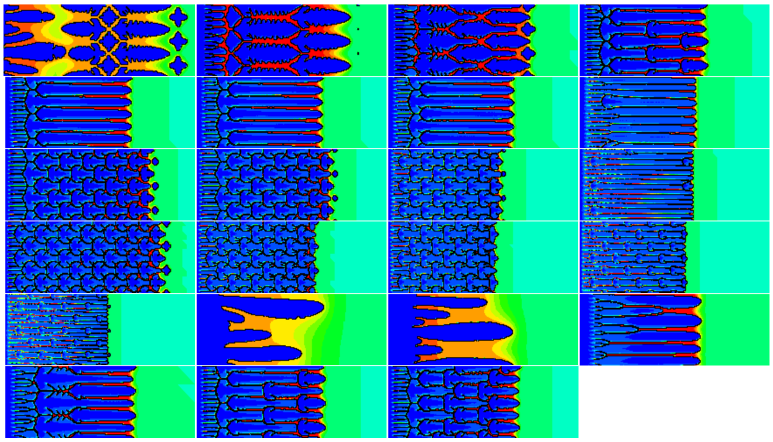


Figure 4.14: Computation results showing solute concentration for the 23 cases listed in Table 4.1 (First row: case $1, 2, 3, 4$; second row: $5, 6, 7, 8$ and so on). Gibbs-Thomson coefficient $\epsilon_c = 0.01 K \cdot \mu m$.

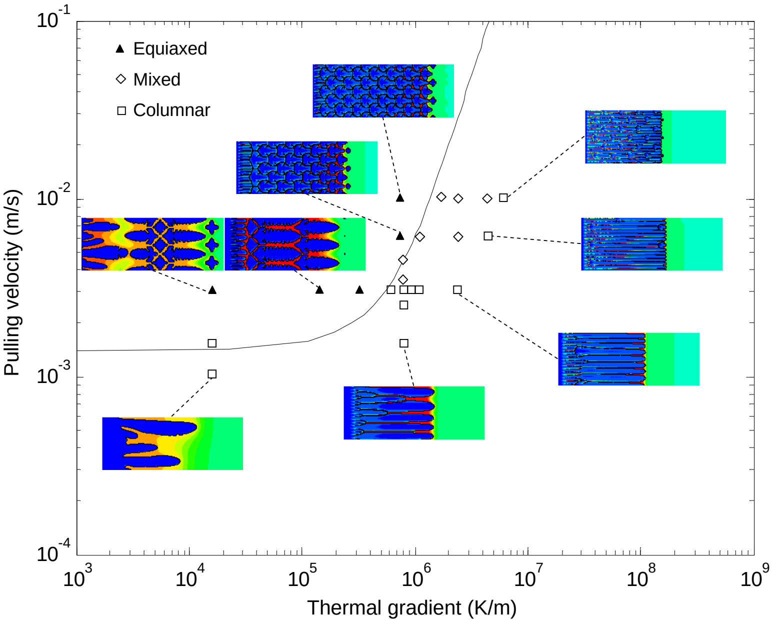


Figure 4.15: CET map for directional solidification for the 23 cases listed in Table 4.1. Gibbs-Thomson coefficient $\epsilon_c = 0.01 K \cdot \mu m$.

without using symmetry shown in the left of Fig. 4.17 take around 4.5 hours with the same computation node. Selecting appropriate computational domain size is indeed an issue of balancing between the numerical solution accuracy and the required computational effort.

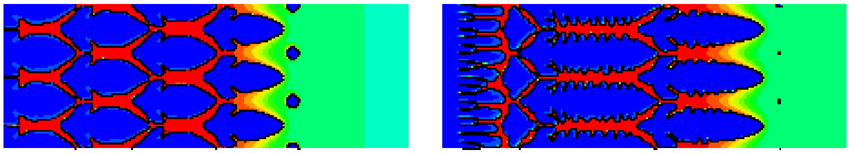


Figure 4.16: Computation results showing solute concentration for case 2 with $\epsilon_c = 0.24K \cdot \mu m$ (left) and $\epsilon_c = 0.01K \cdot \mu m$(right).

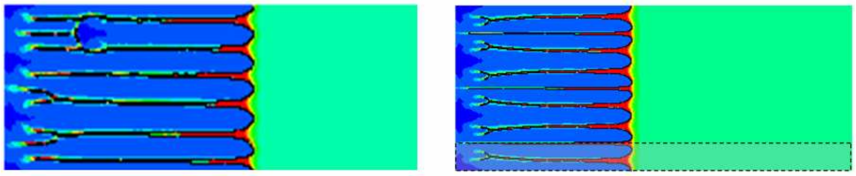


Figure 4.17: Computed growth pattern showing solute concentration without using symmetry (left) and with symmetry (right) for the case 8 listed in Table 4.1. The box with dashed line shown in the right figure is the computation domain when using symmetry.

Multiple three-dimensional crystal interaction

As we know, crystal growth is inherently three-dimensional. Here, we perform a 3D study of the interaction between multiple crystals. Material properties are selected to be the same as in the 2D study with Gibbs-Thomson coefficient $\epsilon_c = 0.24K \cdot \mu m$. However, the spacing between two potential nucleation sites is taken as $\sqrt[3]{2} \times d_s$, so that the density of potential nucleation sites is d_s^{-3} with a body centered structure as shown in Fig. 4.18.

As computation in 3D is more intensive than in 2D, we fully-utilize symmetry. Grid spacing is selected as $0.05 \mu m$ (grid spacing $0.08 \mu m$ was used for the 2D example). For solidification speed $3000 \mu m/s$ and thermal gradient $1400K/cm$, we obtain the 3D crystal shape as shown in Fig. 4.19. The solute concentration at cross sections $x = 0$ and $x = z$ is shown in Fig. 4.20. In comparison to the two-dimensional computation (Fig. 4.7), the required under-cooling for nucleation $8K$ is never obtained. Therefore, nucleation is not observed. In three-dimensions, the solute rejected from the dendrite tips can diffuse

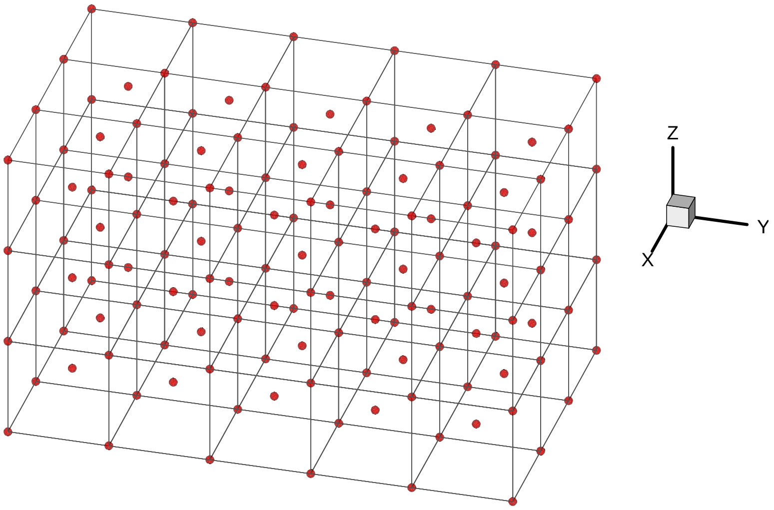


Figure 4.18: Schematic of computational domain and potential nucleation sites for the three-dimensional crystal growth.

away in various directions around the dendrite tip. So the dendrite tip velocity may be able to 'catch up' with the pulling velocity for a given thermal gradient, since the rejected solute is easier to diffuse away comparing with the two-dimensional situation and increase the equilibrium temperature on the interface. Therefore, the maximum under-cooling in front of the dendrite tips is not increasing as the dendrites grow, while in the two-dimensional growth, the diffusion of solute rejected from dendrite tips is confined to either left of right direction of the dendrite tip. Therefore, the dendrite tip velocity may not be as large as the pulling velocity if the thermal gradient is not sufficiently large. This further leads to increasing the maximum under-cooling in front of the dendrite tips until nucleation occurs as the maximum under-cooling is more than ΔT^n.

In order to observe nucleation phenomena in three-dimensions, we reduced the required under-cooling for nucleation from $8K$ to $7K$ with results shown in Figs. 4.21 and 4.22. As shown in the cross section $x = z$ (Fig. 4.22), the pattern of solute concentration has a similarity with the results obtained in two-dimensions. However, the

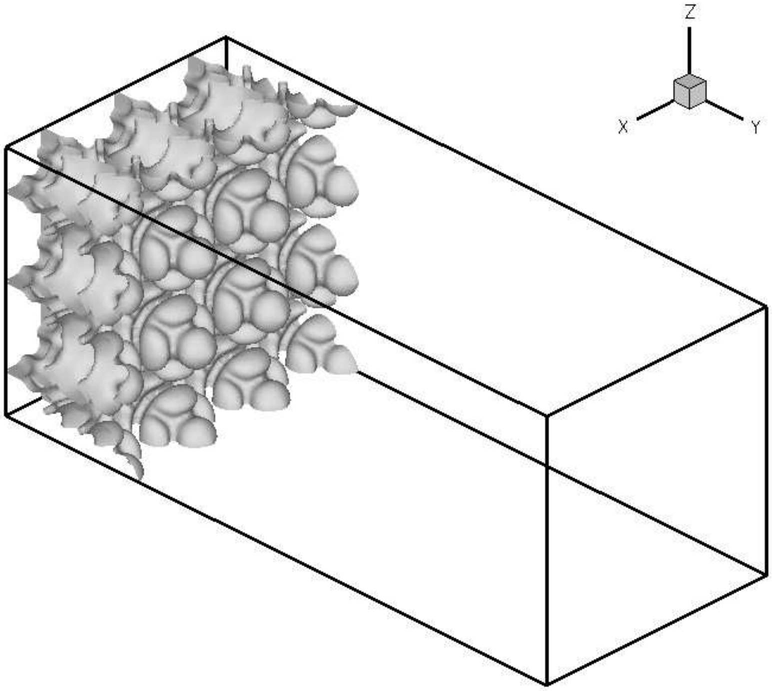


Figure 4.19: Three-dimensional crystal growth with required undercooling with solidification speed $3000\mu m/s$ and thermal gradient $1400K/cm$.

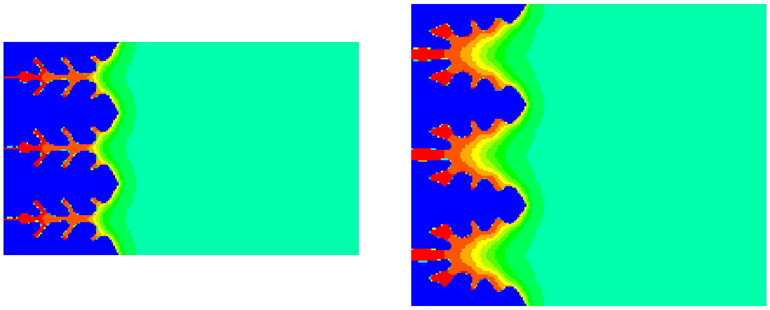


Figure 4.20: Solute concentration at cross sections $x = 0$ (left) and $x = z$ (right) for 3D crystal growth with solidification speed $3000\mu m/s$ and thermal gradient $1400K/cm$.

pattern of solute concentration in the cross section $x = 0$ is quite different from the two-dimensional results. Similarly to the 2D computation, if the thermal gradient is increased from $1400K/cm$ to $21000K/cm$, the growth pattern will convert to columnar growth as shown in Figs. 4.23 and 4.24. The three-dimensional computation in these two cases shows that the two-dimensional computational results can serve as a tool to qualitatively but not quantitatively understand realistic 3D crystal growth.

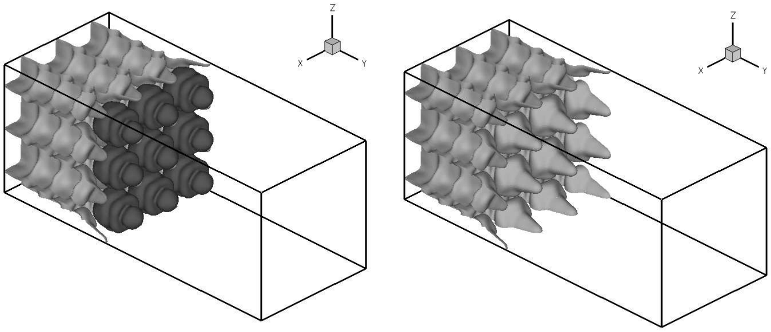


Figure 4.21: Three-dimensional crystal growth with required undercooling for nucleation $7K$ (solidification speed $3000 \mu m/s$ and thermal gradient $1400 K/cm$). The right figure is the same plot as the left figure without plotting the nucleated crystal in the dendrite front.

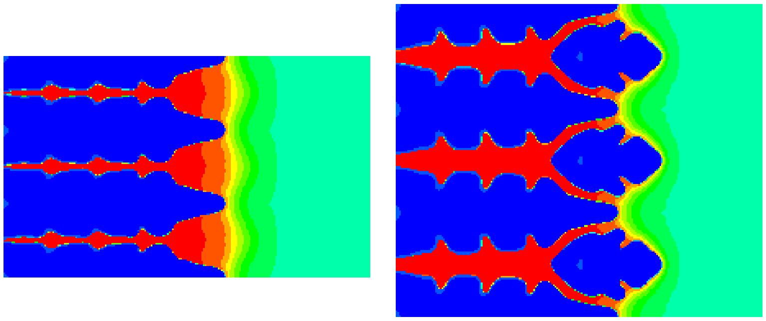


Figure 4.22: Solute concentration at cross sections $x = 0$ (left) and $x = z$ (right) for 3D crystal growth with required undercooling for nucleation $7K$ (solidification speed $3000 \mu m/s$ and thermal gradient $1400 K/cm$).

Effects of randomness on crystal orientation

In the above computation, we have assumed that all crystals have the same orientation to allow us to compare with the results in [5]. However, from the first numerical example studying interaction between 9 crystals, we have found that crystal orientation plays an important role during multiple crystal growth. Here we incorporate the effects of randomness in crystal orientation (uniformly distributed from 0 to 2π) for cases 2 and

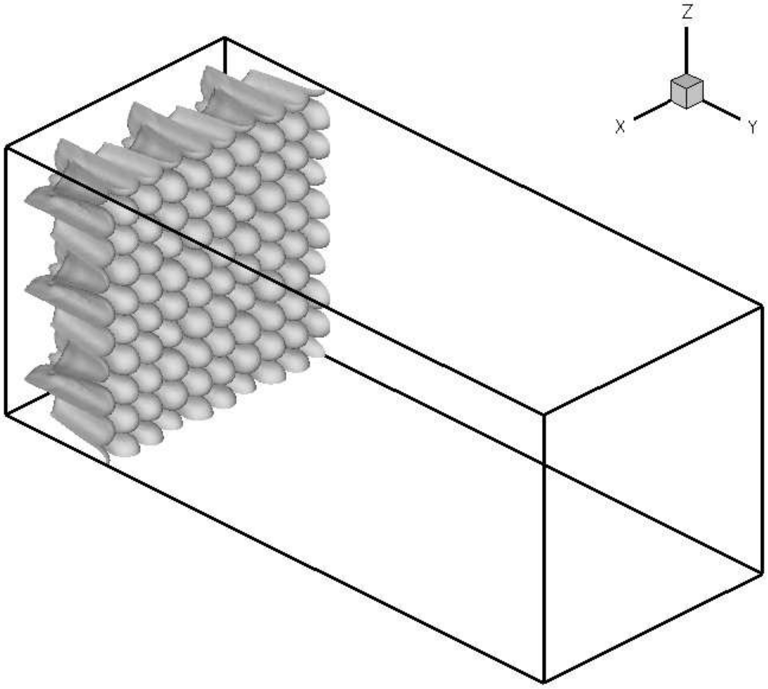


Figure 4.23: 3D Crystal growth with required undercooling with solidification speed $3000 \mu m/s$ and thermal gradient $21000 K/cm$.

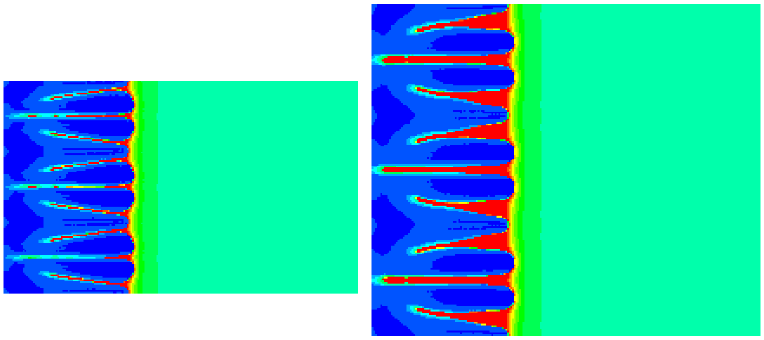


Figure 4.24: Solute concentration at cross sections $x = 0$ (left) and $x = z$ (right) for 3D crystal growth with solidification speed $3000 \mu m/s$ and thermal gradient $21000 K/cm$.

6 of Table 4.1. The Gibbs-Thomson relation coefficient is selected to be $0.01 K \cdot \mu m$ as in the computation of the 23 cases with reduced Gibbs-Thomson relation coefficient. All other parameters are selected to be the same. Due to the randomness in crystal orientation, symmetry is broken and more dendrites have to be included in the numerical

solution to obtain the growth pattern. Therefore, we increased the domain height from $3\sqrt{2}d_s$ to $9\sqrt{2}d_s$.

Computation results of cases 2 and 6 are shown in Fig. 4.25. In comparing with the results shown in Fig. 4.14 we conclude that randomness in crystal orientation has a great effect on the interaction between crystals.

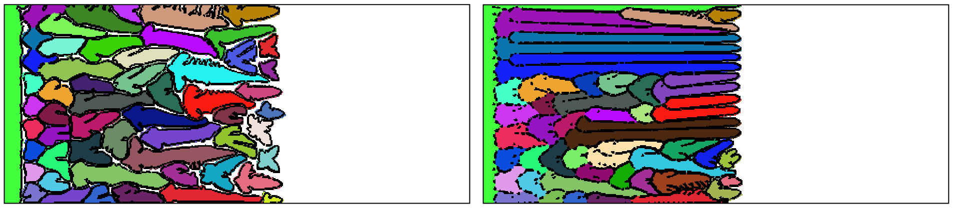


Figure 4.25: Computed growth pattern for cases 2 (left) and 6 (right) with consideration of randomness in crystal orientation. The colors used show crystal orientations.

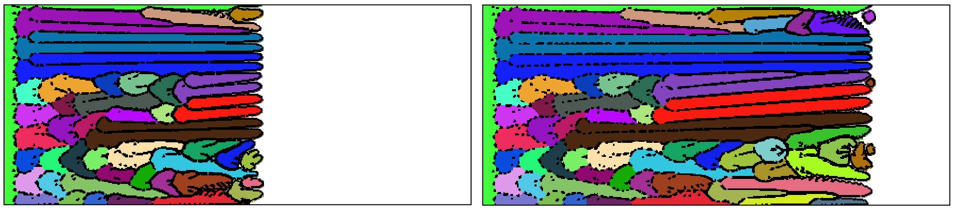


Figure 4.26: Computed growth pattern for case 6 with consideration of randomness in crystal orientation at time step 6000 and 8000. The colors used show crystal orientations.

4.3.3 Interaction between crystals with consideration of randomness in required undercooling for nucleation

In the previous example, we considered the effects of randomness in crystal orientation, while the required undercooling for nucleation was assumed to be constant (8 K). With this assumption, a potential nucleation site will never nucleate if the actual undercooling is less than 8 K. A consequence of this is that there is little variety of microstructure size even though the processing parameters, such as velocity and thermal gradient are significantly different. For example, at very low thermal gradient, we obtain slim equiaxed microstructure as shown in the results of case 1 in the previous example. However in practice, such slim equiaxed microstructure is not common. Instead, coarse equiaxed microstructure is often observed, which is known as the columnar-to-equiaxed transition and usually appears in the center of castings.

In the previous example, the temperature field was not solved. A cooling rate $R = V \times G$ was directly applied. This assumption also limits the application to directional growth where a temperature gradient can be controlled. In many other processes e.g. sand casting, the temperature (or heat flux) is only controlled at the domain boundary instead of the whole domain. To give results with more practical importance, here we perform a numerical study without these assumptions by using random undercooling for nucleation and by solving the heat equation. The following dimensionless material properties are considered: potential nucleation site density $\rho_n = 100$, required undercooling for nucleation ΔT^n is random with distribution $N(1.5, 0.2^2)$, density $\rho = 1$, heat capacity $c = 1$, heat diffusion coefficient $k = 1$, latent heat $L = 100$, Lewis number $Le = 100$, melting temperature $T_m = 0$, initial concentration $C_0 = 0.1$, liquidus slope $m_l = -10$, partition coefficient $k_p = 0.1$, Gibbs-Thomson relation with kinetic undercooling $T_* = T_m + m_l C^l - \frac{2}{3}(1 - \cos(4(\theta - I))V$. A computation domain with size 10×2.5 is used. Initially, the whole domain is liquid. The temperature in the whole domain is distributed with thermal gradient $G = G^l \exp\left[\frac{-Vx}{\alpha}\right]$ with left side temperature at $T_m + m_l C^0$. At the left boundary, a heat flux is withdrawn from the domain with magnitude $q_s = kG^s \exp\left[\frac{-V(0-Vt)}{\alpha}\right]$, while at the right boundary, a heat flux enters the domain with magnitude $q_l = kG^l \exp\left[\frac{-V(10-Vt)}{\alpha}\right]$. Here, $\alpha \equiv \frac{k}{\rho c} = 1$ and $V = \frac{k(G^s - G^l)}{\rho L}$. The reason to select such initial and boundary conditions is that it leads to uniform microstructure throughout the computational domain. Adaptive mesh with smallest grid spacing 0.0098 is used to obtain crystal pattern as shown in the left of Fig. 4.27. A computation domain of 10×2.5 is used here. Three combinations of V and G^l ($(V, G^l) = (0.1, 0.01), (0.35, 0.01), (0.137, 30.0)$) are studied with results shown in Fig. 4.27.

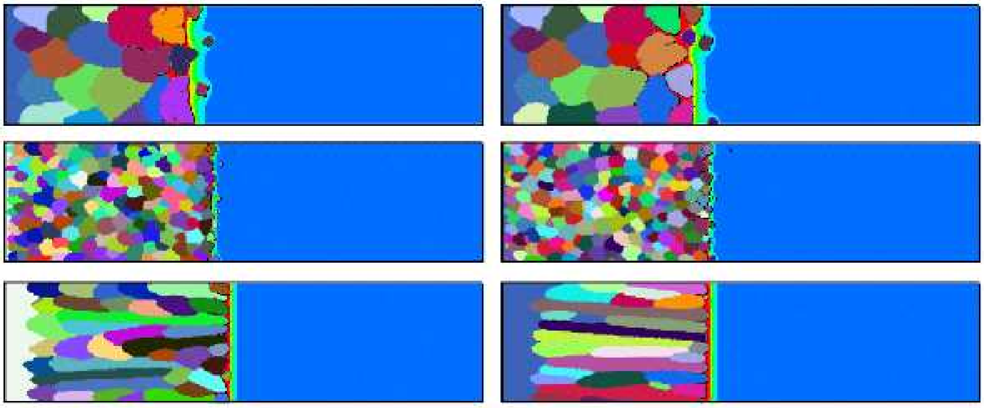


Figure 4.27: Computed growth pattern on a domain of 10×2.5 at various conditions. Left: grid spacing 0.0098; Right grid spacing 0.0049. First row: $V = 0.1$, $G^l = 0.01$; Second row: $V = 0.35$, $G^l = 0.01$; Third row: $V = 0.137$, $G^l = 30.0$. The colors used show crystal orientations.

At low solidification speed and low thermal gradient ($V = 0.1$, $G^l = 0.01$), we observe coarse equiaxed crystal growth, which is not captured in the previous example. By reducing the grid spacing by a half to 0.0049, we obtain different microstructures as shown in the left and right of Fig. 4.27. Due to randomness in crystal orientation angle and in required undercooling for nucleation, numerical noise may lead to very different microstructure details. However, the obtained microstructures with both grids are very similar in their patterns. This suggests that our model can be used to simulate microstructure evolution and predict microstructure patterns, even though there is always error in the numerical solution.

The capability of investigating the interaction between tens of crystals with demon-

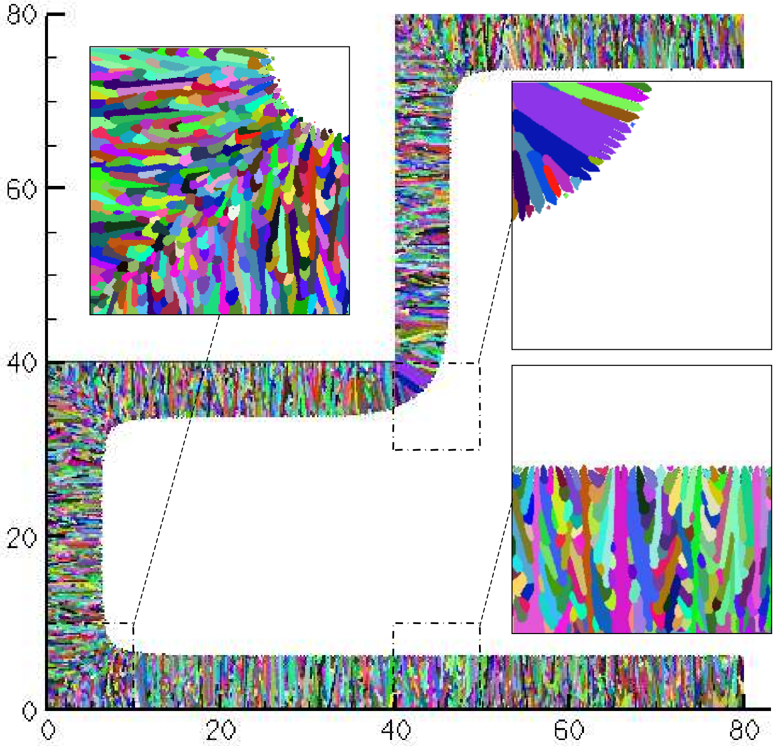


Figure 4.28: Microstructure pattern for growth of hundreds of crystals in a two-dimensional domain. The colors used show crystal orientations.

stration of convergence is an improvement in numerical study of crystal growth. However, hundreds or thousands of crystals are often involved in reality even for a small piece of casting. Based on the previous convergence study for interaction between a small number of crystals, we here demonstrate the ability of the current approach to capture interaction between hundreds of crystals by using the same material and a larger computational domain of size 80×80 (the left-top quarter is not included). The relation of the computational domain and actual domain is demonstrated in Fig. 4.29. Initially, the whole domain is at temperature 30. A cooling rate of 5 is applied at all computational domain boundaries except on the right side, where the heat flux is 0. Using the same grid spacing 0.0098 and a fixed time step of 7×10^{-3}, we obtain the microstructure at time 45 shown in Fig. 4.28. At the corner, equiaxed microstructure is obtained, while at places far away from the corner, a mixture of columnar growth and equiaxed growth is obtained. At the center of the domain, columnar growth dominates. Since hundreds of crystals are involved as shown in Fig. 4.28, the computation of this problem is very intensive. It takes around 24 hours using 16 nodes each with two 3.8G CPUs to obtain the results in Fig. 4.28.

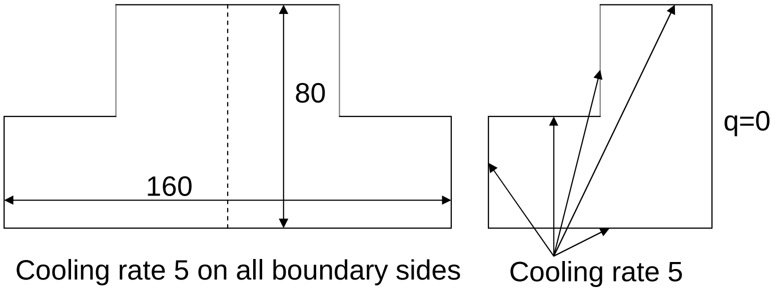


Figure 4.29: Schematic of the actual domain (left) and computational domain (right).

4.4 Conclusions

Our previous numerical work based on the level set method [61], which combines features of both front-tracking methods and fixed domain methods, was extended in this work to study interaction between multiple dendrites during solidification. By using markers to identify different crystals, we are able to study the interaction between multiple dendrites with a single signed distance function. Accuracy of our approach is demonstrated with convergence studies and comparison with the multi-level set method. Simulations of columnar to equiaxed transition are preformed and compared with recent phase field results available in the literature [5]. New abilities provided by the current numerical approach include extension to three-dimensions and arbitrary crystallographic orientations. We find that the three-dimensional growth is significantly different from the two-dimensional growth for one of the two cases considered. However, at other considered cases with high thermal gradient, the microstructure patterns obtained with two- and three-dimensional modeling are both columnar growth. Randomness in crystallographic orientations was found to have a significant effect on the results. Efficiency of the multi-level set algorithm is demonstrated in an example that includes hundreds of crystals with consideration of randomness effects in both crystallographic orientation and required undercooling for nucleation. The algorithms developed here are currently being used as localized subgrid models for the development of efficient multiscale models of solidification processes.

Chapter 5

Multiscale modeling of alloy solidification

5.1 Mathematical model

The difficulty associated with modeling of solidification processes arises from the morphological complexity of the resulting microstructure and the variety of length scales in the system. Under typical solidification conditions, the system and interfacial structures are of the orders of $10^{-1}m$ and $10^{-5}m$, respectively.

Microscopic models (e.g. [49, 41, 61]) are developed to capture physical phenomena in the length scale of the interfacial structures. As pointed out in [69], there are three disparate microscopic length scales (decreasing in order): (1) the overall size of crystals (e.g. primary arm spacing λ_1 for columnar growth), (2) the secondary dendrite arm spacing and (3) the radius of a dendrite tip. The domain size in these microscale studies is usually chosen to contain one or more crystals/dendrites. The grid spacing in the discretization of the computational domain is chosen to be less than the tip radius (therefore also less than the overall size of the crystals and the secondary dendrite arm spacing).

For macroscale models (e.g. [55]), macroscopic variables including velocity, temperature and concentration are defined as the average values of microscopic variables within an averaging volume. The averaging volume is selected [54] such that the scale it represents is small enough to capture the global effects such as fluid flow, heat transfer and species distribution, but large enough to smooth out the details of the morphological complexities, interdendritic fluid flow, latent heat release and species redistribution. Therefore, for macroscale modeling of solidification, we typically need to select the averaging volume to be of the size of a few crystals.

In this work, we will investigate the interaction of multiple crystals using a microscale model with a fine mesh (with grid spacing small enough to capture interfacial structures such as secondary dendrite arm spacing and dendrite tip radius). With the microscale model, variation of temperature and concentration is observed in the range of crystal size. After averaging the microscale model results, information will be extracted to aid computation in a coarse-mesh using the macroscale model. The microscale model will also be utilized to validate the macroscale model results.

The following simplifications are taken for both the microscale and macroscale models.

1. All material properties are assumed to be constant if not mentioned. These include density ρ, heat capacity c, latent heat L, heat diffusion coefficient k, solute diffusion coefficient D, liquidus slope m_l, and partition coefficient k_p.
2. Fluid flow effects are not considered.
3. Solute diffusion in the solid phase is neglected.

5.1.1 Microscale model

The following governing equations for heat transfer and solute transport during solidification are widely accepted for modeling of solidification in the microscale:

$$\rho c \frac{\partial \theta(\mathbf{x}, t)}{\partial t} = k \nabla^2 \theta(\mathbf{x}, t), \mathbf{x} \in \Omega, \tag{5.1}$$

$$\frac{\partial C^l(\mathbf{x}, t)}{\partial t} = D \nabla^2 C^l(\mathbf{x}, t), \mathbf{x} \in \Omega^l, \tag{5.2}$$

where Ω is the total domain including both the domain of the liquid phase Ω^l and the domain of the solid phase Ω^s. The above two governing equations for temperature θ and solute concentration in the liquid C^l are simple diffusion equations that have been very well studied. However, due to the existence of the moving interface, Γ^{sl}, the interface conditions listed below make the problem nontrivial.

1. Because of the phase transformation, solute is rejected from the solid phase to the liquid phase leading to a solute rejection flux at the interface:

   $$D\frac{\partial C^l}{\partial \mathbf{n}} = -(1 - k_p)C^l V\mathbf{n}, \tag{5.3}$$

   where **n** is the normal direction at the solid-liquid interface pointing towards the liquid phase.

2. The temperature on the solid-liquid interface Γ^{sl}, θ_I, equals the equilibrium temperature, θ_*, given by the Gibbs-Thomson relation:

   $$\theta_I = \theta_* \equiv \theta_m + m_l C^l + \varepsilon_c \kappa + \varepsilon_V V, \tag{5.4}$$

   where θ_m is the melting temperature of the solidification material, ε_c is the Gibbs-Thomson relation curvature undercooling coefficient, κ is curvature of the interface, ε_V is the Gibbs-Thomson relation kinetic undercooling coefficient, and V is the interface velocity.

3. The velocity of the solid-liquid interface is governed by the heat flux jump, which is known as the classical Stefan condition:

   $$V = \frac{q^s - q^l}{\rho L}, \tag{5.5}$$

   where q^s and q^l are the heat fluxes at the solid and liquid sides, respectively of the interface.

A level set equation is used here to implicitly describe the evolution of the solid-liquid interface:

$$\frac{\partial \phi}{\partial t} + V\|\nabla\phi\| = 0. \tag{5.6}$$

The signed distance function $\phi(x, t)$ is simply the distance of location x to the interface at time t but with '+' sign, if x is the liquid phase and '-' sign if x is in the solid phase. According to this definition, $\phi = 0$ represents the solid-liquid interface.

Nucleation is a very complicated phenomenon related with the type and amount of impurities in the material. We denote the number of impurities per unit volume, i.e. the density of impurities, as ρ^n. Suppose that the domain of interest has a volume of V, then there will be $\rho^n V$ potential nucleation sites with each impurity serving as a potential nucleation site. For each potential nucleation site i, we model three random variables, the potential location for nucleation x_i, the required undercooling for nucleation $\Delta\theta_i^n$ and the orientation I_i of the nucleated crystal. These variables are defined below:

1. x_i is the location of the potential nucleation site. We assume that x_i is equiprobable at each point of the whole domain.
2. $\Delta\theta_i^n$ is the required undercooling for the potential nucleation site i to nucleate and become a crystal seed. If and only if the undercooling at the potential nucleation site i is greater than $\Delta\theta_i^n$ (i.e. $\theta_m + m_l C_i - \theta_i \geq \Delta\theta_i^n$, where θ_m is the melting temperature, m_l is the liquidus slope, C_i and θ_i are concentration and temperature at potential nucleation site i), site i becomes a crystal seed. We assume that $\Delta\theta_i^n$ follows a normal distribution with mean μ and variance σ^2.
3. The other random variable, I_i, is the orientation angle at potential nucleation site i. The orientation angle is the preferred crystal growth direction. We only consider here the randomness of orientation angle in two dimensions, where I_i is sampled from a uniform distribution from 0 to 2π. After nucleation at location of potential

nucleation site i, the crystal may in general rotate (e.g. as a result of convection). This leads to change in orientation angle for the crystal. This movement of crystal is not considered in our microscale model. In other words, the orientation of the crystal, I, is assumed to be fixed during growth of the crystal at the value it attains during nucleation (i.e. I_i if it is nucleated from potential nucleation site i).

Implementation of the above microscale model has been discussed in [63, 61] with demonstration of convergence and accuracy for the computed solutions. In this work, we assume that the microscale model gives the true solution. Focus will be on multiscale modeling to obtain for our problem of interest the same results as obtained using the above microscale model but with significantly less computational effort.

5.1.2 Macroscale model

In volume averaging models (e.g. [54]), a volume size is specified to perform averaging and obtain macroscopic variables. This averaging volume needs to be at appropriate size. If the averaging volume is too small, there may be not enough microscopic data to smooth out the local variation. On the other hand, if the averaging volume is too large, the global variation may not be accounted. Using the size of a few crystals as the averaging volume size might be appropriate. However, crystal sizes usually vary significantly even in the same solidification system. In this work, we will employ a different approach that avoids using an averaging volume for defining macroscopic variables.

Let us first define the solution of the microscale model (microscopic variables), i.e. temperature, $\theta(x,t)$, concentration, $C(x,t)$, orientation, $I(x,t)$ and signed distance $\phi(x,t)$ to the solid/liquid interface at location x and time t. Computation of the microscopic variables is not deterministic because of the randomness in nucleation during solidification as discussed earlier in the microscale model. We define the expectation of the microscopic variables $\theta(x,t)$, $C(x,t)$, and $\phi(x,t)$ as the macroscopic temperature, macroscopic concentration and liquid volume fraction as follows:

$$\begin{aligned} T(x,t) &\equiv\ < \theta(x,t) >, && (5.7) \\ \bar{C}(x,t) &\equiv\ < C(x,t) >, && (5.8) \\ f(x,t) &\equiv\ < H(\phi(x,t)) > . && (5.9) \end{aligned}$$

Here $H(.)$ is the Heaviside function taking value 1 if the parameter is greater than 0, and value 0 if the parameter is less than 0. The expectation $< \cdot >$ in the above definitions is over all possible distributions of potential nucleation sites. As a consequence of the above definitions, microscopic variables need to be computed for a few different distributions of potential nucleation sites before the macroscopic variables above are obtained.

In this work, we will only use the definitions in Eqs. (5.7)-(5.9) above to compare a fully-resolved microscale simulation over the domain of interest (when such simulation is possible) with macroscopic variables introduced as part of the multiscale framework.

Macroscopic temperature - The macroscopic temperature T is defined in this work as the solution of the heat diffusion equation with a latent heat term as follows:

$$\rho c \frac{\partial T}{\partial t} = k \nabla^2 T - \rho L \dot{f}. \tag{5.10}$$

This governing equation for the macroscopic temperature T is very similar to simple heat diffusion except with a latent heat term $-\rho L \dot{f}$. The first law of thermodynamics, energy conservation, is implied in Eq. (5.10). This assumption is also utilized in our previous macroscale model using volume-averaging techniques [54]. The function $f(x,t)$ (and thus its time-derivative $\dot{f}(x,t)$) defining the latent heat evolution is as of now an unknown function that needs to be specified as part of the multiscale framework.

Macroscopic concentration - The macroscopic concentration $\bar{C}(x,t)$ is assumed to be constant, i.e. $\bar{C}(x,t) = C_0$, where C_0 is the initial concentration. Since convection is not considered in this work, solute rejected from the solid crystals/dendrites will only cause micro-segregation [34] with variation of concentration in the scale of the solute boundary layer. The size of the solute boundary layer (estimated as D/V, where D is the solute diffusion coefficient and V is the solidification speed) is usually less than the size of the crystals. So in the macroscale, which involves a significant number of crystals, the variation of the concentration is very small. The assumption of $\bar{C}(x,t) = C_0$ is thus reasonable. This is also true for our previous macroscale volume-averaging model in the case of negligible fluid flow [55].

Microstructure features - One is often interested in the features of the solidification microstructure at the end of the solidification process, such as the primary dendrite arm spacing, the secondary dendrite arm spacing or the Heyn's interception measure. Let us denote in an abstract format the finally obtained microstructure features at location x as $\Lambda(x)$. The precise definition of $\Lambda(x)$ used in this work will be discussed later in Section 5.2.4. In this work, we consider the macroscopic temperature $T(x,t)$, macroscopic concentration $\bar{C}(x,t)$, liquid volume fraction $f(x,t)$, and microstructure features $\Lambda(x)$, as the macroscopic variables in our multiscale framework. An averaging volume is needed to define the microstructure features Λ at a particular point x as demonstrated in Fig. 5.1. We need to orient the average volume (rectangle) in a way such that it takes account of the growth direction (the rectangle's top and bottom sides are along the growth direction). In this work, we will predict $\Lambda(x)$ utilizing the data generated from directional growth problems (on the microscale) that are defined from the features (growth velocity and temperature gradient) at point x. $\Lambda(x)$ will be used to provide a measure of the type and size of the obtained microstructure at location x. The final microstructure features are assumed to depend only on the cooling history and temperature gradient history as

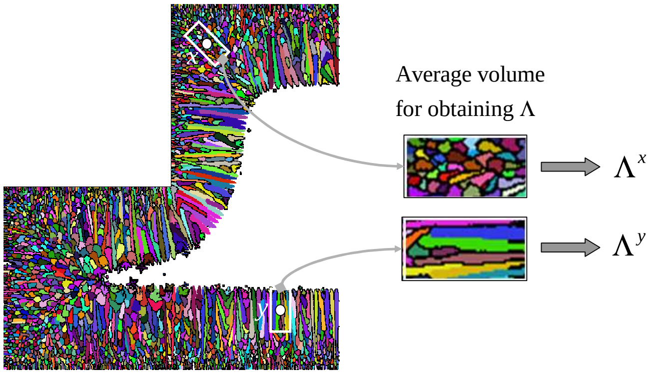


Figure 5.1: Schematic of average volume to obtain microstructure features at two points. The features Λ are defined by statistical averaging of the results of appropriately defined microscale directional solidification problems.

follows:

$$\Lambda(x) = \Lambda(\mathbf{R}^x, \mathbf{G}^x), \tag{5.11}$$

where the cooling history $\mathbf{R}^x$ and temperature gradient history $\mathbf{G}^x$ are two functions of time t. They are defined as $\mathbf{R}^x \equiv \frac{\partial T(x,t)}{\partial t}$ and $\mathbf{G}^x \equiv \|\nabla T(x,t)\|$, respectively. In this assumption (Eq. 5.11), we use superscript x to emphasize that the cooling rate and temperature gradient depend on the location x. The bold form (**R** and **G** instead of R and G) is used to denote that the cooling rate and temperature gradient vary with time. The final microstructure features after solidification, $\Lambda(x)$, are often of great interest as they are related with the mechanical properties of cast products.

There are lots of analytical and numerical studies in the literature [63, 34, 24, 5] investigating the effects of cooling rate and thermal gradient on the microstructure pattern for directional solidification. All these studies [63, 34, 24, 5] demonstrate that the cooling rate and temperature gradient determine the microstructural features for a given material even though there is no universal agreement on the functional form of this dependence. However, they all point to Eq. (5.11) as a reasonable assumption.

One significant simplification made in these available studies is that of constant cooling rate and constant temperature gradient, so the commonly used approximations are of the form $\Lambda(x) = \Lambda(R^x, G^x)$. In this work, we are relaxing this assumption to $\Lambda(x) = \Lambda(\mathbf{R}^x, \mathbf{G}^x)$. Although our assumption states that the whole history of cooling rate and temperature gradient determines the microstructure, intuitively we know that only a part of this history is important. At time much earlier or much later than a particular period, cooling rate and temperature gradient have only minor effects on the microstructure. Therefore, it should be possible to replace $\mathbf{R}^x, \mathbf{G}^x$ in the above assumption with much simpler parameters. The proposed multiscale approach will provide a reasonable replacement.

Liquid volume fraction f - We assume that the macroscopic liquid volume fraction only depends on the temperature and microstructure features as follows:

$$f(x,t) = f\left(T(x,t), \Lambda(x)\right). \tag{5.12}$$

In our previous macroscale volume-averaging model [54], the volume fraction was taken only as a function of temperature for a given alloy, i.e. $f(x,t) = f(T(x,t))$. However, for a given alloy, we may obtain different microstructure patterns, e.g. planar, cellular, dendritic, or mixed. To more accurately model the liquid volume fraction variation, we consider its dependence on microstructure features, which leads to the above assumption.

As a summary, our macroscale model defining T, $\bar{C}$, $f(x,t)$ and $\Lambda(x,t)$ is given as follows:

$$\rho c \frac{\partial T}{\partial t} = k\nabla^2 T - \rho L \dot{f}, \tag{5.13}$$

$$f = f\left(T, \mathbf{R}, \mathbf{G}\right), \tag{5.14}$$

$$\Lambda = \Lambda(\mathbf{R}, \mathbf{G}), \tag{5.15}$$

$$\bar{C}(x,t) = C_0. \tag{5.16}$$

The purpose of the above simplified model is not only to capture the effects of microstructure evolution on the macroscopic variables, but also to provide means to extract the microstructural features for a given values of macroscopic variables.

5.1.3 Unknown functions

Notice that even though the macroscale model contains only macroscopic variables, one cannot solve it directly due to the existence of the two unknown functions, $f(T, \mathbf{R}, \mathbf{G})$ and $\Lambda(\mathbf{R}, \mathbf{G})$. Information is needed from the microscale in order to determine these two functions:

One approach in computing these functions is to perform analytical studies. For example, assuming directional growth with a constant cooling rate R and thermal gradient G, the primary spacing λ_1 has been estimated for cellular type or dendritic type growth as follows [34]:

$$\text{Cellular type growth} : \lambda_1(R,G) = (\frac{3\Delta T_t R}{G})^{1/2}, \tag{5.17}$$

$$\text{Dendritic type growth} : \lambda_1(R,G) = \frac{4.3(-\Delta T_t D \varepsilon_c)^{0.25}}{(k_p R)^{0.25}}, \tag{5.18}$$

where ΔT_t is the undercooling at the dendrite or cell tips. As pointed out in [34], these estimations can only be used as a qualitative description of primary arm spacing, since a number of simplifications are taken to make this analytical study possible.

Simplifications and assumptions are also taken for analytical estimation of the other unknown function, $f(T, \mathbf{R}, \mathbf{G})$. It is often simplified as $f(T)$. Assuming infinite back diffusion or zero back diffusion, $f(T)$ can be estimated as [11]:

$$\text{Lever rule} : f = 1 - \frac{T - T_L}{(1-k_p)(T-\theta_m)}, \tag{5.19}$$

$$\text{Scheil rule} : f = (\frac{T - \theta_m}{T_L - \theta_m})^{\frac{1}{k_p - 1}}, \tag{5.20}$$

where T_L is defined as $\theta_m + m_l \bar{C}$.

Note that in the above two estimations of $f(T)$, very little microscale information is incorporated. For example, solidification alloys with the same concentration may have different densities of potential nucleation sites or different required undercooling for nucleation. These parameters do not appear at all in the above equations. Therefore, the above phenomenological estimations may significantly deviate from the actual relation between the liquid volume fraction and temperature.

Another approach is to use numerical analysis that is capable of removing several of the simplifications required in analytical calculations. In fact, numerical methods and of course experimentation are the only ways to quantitatively evaluate the unknown functions f and Λ without introducing too many simplifications. A number of efforts in the direction of numerical study have been presented. For example, in our previous work [63], we proposed a method based on the level set method with markers, which can be used to simulate interaction between multiple crystals. In this earlier work, demonstration of convergence of the results with adaptive mesh refinement was shown. In this work, we will use this previously developed microscale solver to generate information for evaluating the two unknown functions.

5.2 The database approach

It is straightforward to think of running the microscale model for some particular problems and post-process the obtained results to derive some estimation of the functions f and Λ. The particular problems, which usually have small domains (finer scale), must be of some relevance to our macroscopic problem of interest, which usually has a much bigger domain (coarser scale). We call the selected particular problems with smaller domains as the ‘sample problems’. This idea is referred to here as the database approach. One can define different sample problems with different domains, different initial conditions or different boundary conditions. Some of them may be relevant to our problem of interest, some may not. Selecting good sample problems would be the key to success of the database approach for multiscale solidification modeling.

5.2.1 Domain of the sample problem

5.2.2 Model M and features F_M

In addition to the domain of the problem of interest, we herein define the domain of the sample problem where dendritic solidification results will be generated for use in the database multiscale approach. The domain of the sample problem in general depends on the physics that we are interested to resolve on the microscale. In particular, it should be large enough to contain a sufficient number of crystals for averaging and small enough to minimize the computational cost of simulating in this sample domain dendritic solidification (microscale model). In our numerical examples, the domain of the sample problems is selected as a rectangle that at the end of solidification contains around 10 to 100 crystals. For the examples considered later in this paper, this is approximately $\frac{2}{100}$ to $\frac{8.64}{100000}$ of the area of the problem of interest (i.e. of the problem where we are interested to perform multiscale analysis).

In Section 5.1, we discussed two models (microscale and macroscale models). The microscale model (also referred to here as the fully-resolved model) can be applied both to the sample problem domain as well as to the domain of the problem of interest. However, the simulation corresponding to the last problem may be impractical to perform for most realistic systems due to the intensity of its computational requirements. Here, we introduce a third model (that for simplicity of presentation we will refer to as model M) to solve the solidification problem. This model 'treats the material as a pure material without modeling of nucleation'. Melting temperature of the 'pure' material is modeled to be $\theta_m + m_l C_0$. The governing equations for model M are as follows:

$$\begin{aligned} \rho c \frac{\partial \theta_M}{\partial t} &= k \nabla^2 \theta_M, && (5.21) \\ V_M &= \frac{q_s - q_l}{\rho L}, && (5.22) \\ \frac{\partial \phi_M}{\partial t} &+ V_M \|\nabla \phi_M\| = 0. && (5.23) \end{aligned}$$

Here, θ_M, V_M and ϕ_M are the temperature, interface velocity and signed distance from the interface. The subscript M is used to emphasize that the solution is obtained from model M (not from the microscale model). q_s and q_l are the heat fluxes at the solid and liquid sides of the interface, respectively. Model M can be applied both to the problem of interest as well as to the sample problem domain. Fig. 5.2 shows for example its application to the domain of the problem of interest. Because the solidification material (an alloy) is treated as a pure material without modeling of nucleation in model M, we expect that the solution of model M (θ_M and ϕ_M) will not capture the important physics during the solidification process as well as it will be captured by the previously discussed microscale model or macroscale model.

The solid-liquid interface solved with model M is stable since pure material is assumed without nucleation. We define the solution features of model M at x, $F_M(x)$, to be the solidification speed, $V_M(x)$, and thermal gradient in the liquid phase, $G^l_M(x)$, when the solid-liquid interface passes through location x, i.e. $F_M(x) \equiv (V_M(x), G^l_M(x))$ as demonstrated in Fig. 5.2. For any given set of solution features (V_M, G^l_M), we can

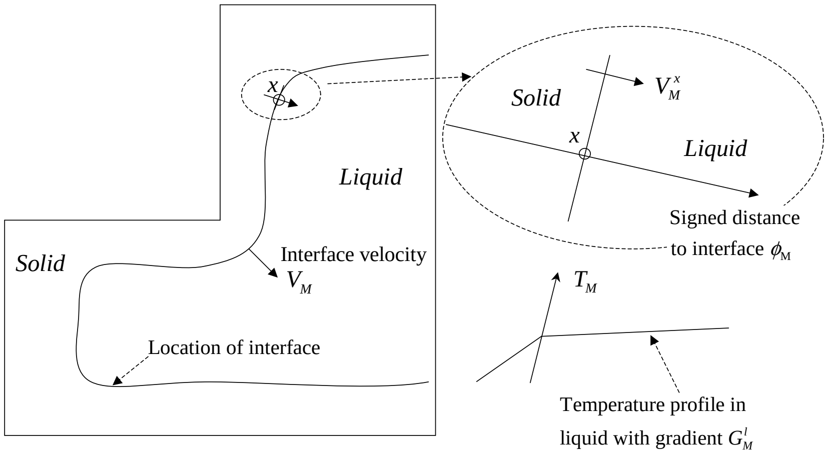


Figure 5.2: Schematic of solution features of model M ($V_M(x)$ and $G_M^l(x)$) as applied to the domain of the problem of interest.

find a unidirectional sample problem such that when it is solved with model M, the obtained solution features at any location of the problem domain are the same as the given solution features. This is discussed next.

5.2.3 Model M applied to the sample problem domain for modeling directional solidification with constant features F_M

Model M is introduced here as a simplified model capable of identifying sample problems relevant to our problem of interest. Let F_M represent the features at point x of the problem of interest obtained with the application of model M. We can now identify a sample problem that when solved with model M corresponds to directional solidification with constant features F_M. Indeed, it is not hard to verify that

$$\theta_M = \theta_m + m_l C_0 + \begin{cases} \frac{\alpha G_M^s}{V_M}\left\{1 - \exp\left[\frac{-V_M(x - V_M t)}{\alpha}\right]\right\}, \text{ when } x < V_M t \\ \frac{\alpha G_M^l}{V_M}\left\{1 - \exp\left[\frac{-V_M(x - V_M t)}{\alpha}\right]\right\}, \text{ when } x \geq V_M t \end{cases}, \tag{5.24}$$

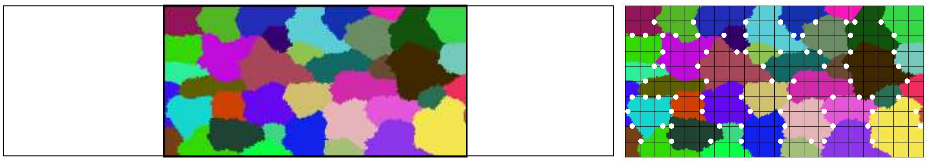


Figure 5.3: Left: Domain for computation (outer rectangle) and domain for performing averaging (inner rectangle); Right: Schematic of the process for obtaining microstructure features.

is an analytical solution that satisfies model M and gives the exact features (V_M, G^l_M) on the whole domain. Here $\alpha \equiv \frac{k}{\rho c}$ and $G^s_M \equiv G^l_M + \rho L V_M/k$. The above solution satisfies $\rho c \frac{\partial \theta_M}{\partial t} = k\nabla^2 \theta_M$. The interface position at time t is at $x = V_M t$, since the temperature θ_M at location $x = V_M t$ equals to $\theta_m + m_l C_0$, which is the melting temperature of the 'pure' material. The temperature gradient at the left side of the interface is G^s_M, whereas the temperature gradient at the right side of the interface is G^l_M. To define a sample problem giving solution features (V_M, G^l_M) everywhere on its domain, we only need to select a domain and apply boundary/initial conditions according to the above analytical solution. In our numerical examples, the following is used to define a sample problem:

(1) Its domain is as discussed in Section 5.2.1.

(2) The initial temperature is given by Eq. (5.24) with $t = 0$,

(3) Adiabatic conditions are considered at the top and bottom sides, whereas the heat fluxes at the left and right side are taken by differentiation of the analytical solution in Eq. (5.24):

$$q_{\text{left}} = kG^s_M \left\{ \exp\left[\frac{-V_M(0 - V_M t)}{\alpha} \right] \right\}, \tag{5.25}$$

$$q_{\text{right}} = -kG^l_M \left\{ \exp\left[\frac{-V_M(L - V_M t)}{\alpha} \right] \right\}, \tag{5.26}$$

where L is the length of the sample domain in the growth direction.

5.2.4 Microscopic (fully-resolved) model applied to the sample problem domain for modeling directional solidification with constant features F_M

The solution of the fully-resolved microscale model in the sample domain is not unidirectional since symmetry in the vertical direction is not preserved due to the unstable solid-liquid interface (as a result of solute rejection) and the random nature of nucleation. However, application of the simple model M with the same initial and boundary conditions results in a unidirectional solution which further gives constant features F_M.

Our sample problems with the microscale model represent directional solidification from the left to the right. To avoid the effects of initial and final transient stages on the evaluation of f and Λ, we perform averaging of features only on a part of the computational domain as demonstrated in Fig. 5.3.

At the end of solidification, we count the number of intersections of the horizontal lines with crystal boundaries, N_x, within the domain for averaging as demonstrated in the right plot of Fig. 5.3. Similarly, we count the number of intersections of the vertical lines with crystal boundaries, N_y. In general, N_x is much less than N_y if the solidification microstructure pattern is columnar, while for equiaxed type of microstructure, N_x is only slightly less than N_y. If the microstructure is fine (coarse), the number of intersections N_y is large (small). Since the microstructure type (columnar or equiaxed) and microstructure size are of great interest, we define the microstructure features as $\Lambda \equiv (\frac{N_x}{N_y}, \frac{1}{N_y})$. If $\frac{N_x}{N_y} \leq 0.7$, we assume that the microstructure type is columnar. If $\frac{N_x}{N_y} > 0.7$, then we assume the microstructure type to be equiaxed. Here, $\frac{1}{N_y}$ is a measure of the microstructure size, thus a larger $\frac{1}{N_y}$ corresponds to a larger microstructure size.

Let us now concentrate in the computation of the liquid volume fraction $f(T, F_M)$ at a given temperature T for solidification with features F_M. Recall that each sample problem is designed such that it leads to constant features F_M when solved with model M. $f(T, F_M)$ can be interpreted here as the probability of being liquid 'in the average' at a given temperature T within the sample problem defined by features F_M. Using the sample problem results, we estimate $f(T, F_M)$ as the ratio of the number of grid points in our sample problem that remain liquid at T to the total number of grid points. For each grid point x in the sample problem grid, 'x remains liquid at temperature T' is equivalent to 'x solidifies at a temperature lower than T', that is '$\theta_s(x) < T$', with θ_s the temperature at x at the time t_s that it solidifies. Therefore, $f(T, F_M)$ can be evaluated using the field of θ_s as follows:

$$f(T, F_M) \approx \frac{\#\, of\ nodes\ with\ \theta_s < T}{total\ \#\ of\ nodes}, \quad sample\ problem\ defined\ with\ F_M. \qquad (5.27)$$

Notice that the evaluation of $f(T, F_M)$ in Eq. (5.27) is performed for each microscale sample problem only after the solidification is complete in the whole domain. The above formulation (Eq. 5.27) can be used only when θ_s (temperature at solidification) is obtained for all nodes. Before all nodes are solidified, $\theta_s(x)$ (temperature at solidification) remains unknown for the liquid nodes ($\phi(x) > 0$). If the sample problem domain is increased, then more number of points are used for averaging, which means better accuracy of estimation $f(T, F_M)$. However, the computation cost of solving the microscale model for the sample problem will also increase.

5.2.5 Sample problems relevant to the problem of interest

Let us denote the above discussed solution features (f, Λ) obtained in the sample domain with the fully-resolved model as F_{full}. Before we discuss the strategy of selecting good sample problems to generate data for solving the problem of interest, let us first focus on an easier question: 'Given a sample problem, how can we infer its relevance to our problem of interest?'

One idea is to use the microscale model for making this decision. Suppose that at location x for the problem of interest, we obtain liquid volume fraction and microstructure features, $F^x_{full} = (f^x, \Lambda^x)$ using the microscale model (this is of course a difficult or computationally impossible problem). Let us also assume that for the sample problem, we obtain (f, Λ) using the microscale model as discussed in Section 5.2.4. If $F^x_{full} \approx F_{full}$, i.e. $(f^x, \Lambda^x) \approx (f, \Lambda)$, then we could say that the selected sample problem is appropriate for evaluating f and Λ at x.

However, we do not want to run the fully-resolved (microscale) model to determine whether a sample problem is relevant or not, because of the huge computational cost it requires for solving the problem of interest. Instead of the fully-resolved model, we can use the much simpler model M to figure out the relevance of the sample problem to the problem of interest. We could run this simple model M, for the problem of interest and the sample problem to obtain solution features F^x_M and F_M, which are defined on the solution of model M. Comparing these solution features, we could have a sense whether the selected sample problem is relevant or not to our problem of interest at location x.

Before exploring further the above idea, let us first introduce a number of definitions.

1. Given predefined solution features F, if a sample problem $\mathbf{P}$ (with appropriate domain $\Omega^{\mathbf{P}}$, initial and boundary conditions) gives constant solution features over $\Omega^{\mathbf{P}}$ (denote the constants as $F^{\mathbf{P}}_M$) using model M (i.e. $\forall y \in \Omega^{\mathbf{P}}$, $F^y_M = F^{\mathbf{P}}_M$), and if the constant solution features equal the predefined solution features (i.e. $F^{\mathbf{P}}_M = F$), then we say that the sample problem $\mathbf{P}$ is *relevant* to solution features

F with model M.

2. Suppose that for the problem of interest at location x, the solution features of model M are F_M^x. If a sample problem $\mathbf{P}$ is *relevant* to solution features F_M^x with model M, then we say that the sample problem $\mathbf{P}$ is *relevant* to location x for the problem of interest with model M.
3. If the sample problem $\mathbf{P}$ is *relevant* to location x for the problem of interest with both model M (i.e. $F_M^{\mathbf{P}} = F_M^x$) and the fully-resolved model (i.e. $F_{full}^{\mathbf{P}} = F_{full}^x$), then we say that the sample problem $\mathbf{P}$ is *consistent* with model M at location x for the problem of interest.

Remark 1: Note that in the definition of 'relevance' of the sample problem $\mathbf{P}$ to location x of the problem of interest, we require constant solution features of model M for the sample problem. Constant solution features over the sample problem domain gives us convenience in averaging the fully-resolved model solution, since all grid points within the sample problem domain undergo the same condition (having the same solution features) in the sense of model M. It is reasonable to use all grid points for performing the average to smooth out variations in the microscopic solution (obtained using the fully-resolved model) as was detailed in Section 5.2.4.

5.2.6 Multiscale framework

Even though it is difficult to find a sample problem that is *relevant* to location x with the fully-resolved model, we can find a sample problem, $\mathbf{P}$, *relevant* to location x with the simpler model M (i.e. $F_M^{\mathbf{P}} \approx F_M^x$) as discussed earlier. By introducing the assumption that $\mathbf{P}$ is *consistent* with model M (i.e. $\mathbf{P}$ also *relevant* to location x with the fully-resolved model), we can run the fully-resolved model to obtain $(f^{\mathbf{P}}, \Lambda^{\mathbf{P}})$, which is also

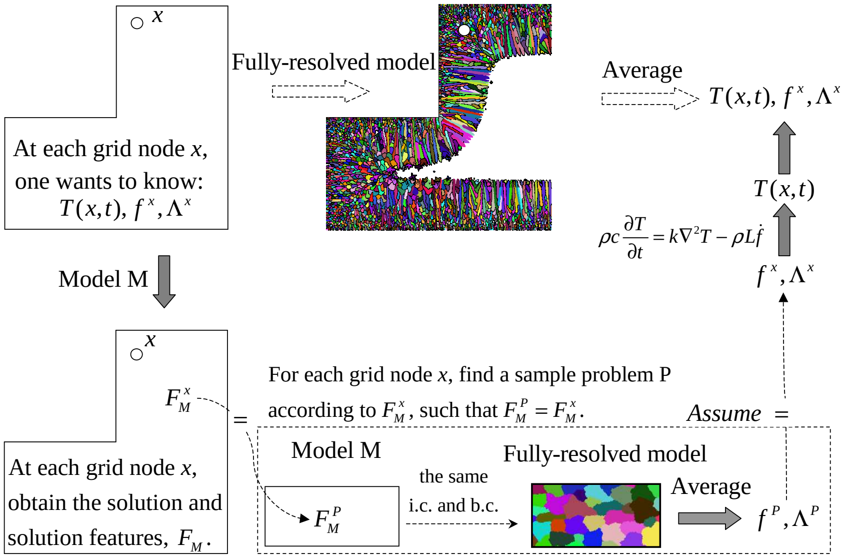


Figure 5.4: Schematic of the multiscale framework (steps indicated with the dark arrows).

(f^x, Λ^x) according to the *consistency* assumption discussed earlier. In other words, once solution features F_M are given, we can find a *relevant* sample problem and use the *consistent* assumption to obtain f and Λ. Therefore, f and Λ can be understood as functions of F_M, i.e. $f = f(T, F_M)$ and $\Lambda = \Lambda(F_M)$. With this idea, the macroscale model is changed to a multiscale model as follows (see Fig. 5.4):

$$\rho c \frac{\partial T}{\partial t} = k\nabla^2 T - \rho L\dot{f}, \tag{5.28}$$

$$f = f(T, F_M), \tag{5.29}$$

$$\Lambda = \Lambda(F_M). \tag{5.30}$$

Comparing the above model with the original macroscale model, we have now used F_M to replace $(\mathbf{R}^x, \mathbf{G}^x)$. The procedure of using the database approach would then involve three important steps:

1. Solve model M for the problem of interest to obtain the solution (θ_M and ϕ_M) and

extract the solution features $F_M = (V_M, G^l_M)$.

2. Define and solve *relevant* sample problems using the microscale model to evaluate $f(T, F_M)$ and $\Lambda(F_M)$.
3. Solve $\rho c \frac{\partial T}{\partial t} = k\nabla^2 T - \rho L \dot{f}$ with the obtained information of $f = f(T, F_M)$.

These three steps are demonstrated in Fig. 5.4. As shown in the box with dashed line (lower right of the figure), model M and the fully-resolved model are used for the same sample problem. The sample problem strategy discussed in Section 5.2.3 guarantees that the solution features of model M are equal to the given solution features F^x_M. We have an analytical solution with model M for the sample problem, so model M is not numerically solved for the sample problem (as is the case for the problem of interest in step 1 of the algorithm above). Only the fully-resolved model is numerically solved for the sample problem to generate data for estimation of $f(T, F_M)$ and $\Lambda(F_M)$ so that the macroscale model incorporates information from the microscale.

Once we have obtained the values of f at the various temperature levels for each node (according to F_M at the corresponding node) after step 2, we need to keep track of these values at each node in the memory. Notice that the liquid volume fraction f appears in the temperature governing equation of the macroscale model, whereas the microstructure features Λ do not appear in the temperature governing equation. Obtaining microstructure features Λ can be understood as a post-processing process. Since their values do not affect other computations, the microstructure features Λ do not have to be tracked in the memory during the computation. All the effect of the microscale on the macroscale is in the liquid volume fraction f.

The two horizontal arrows on the upper part of Fig. 5.4 demonstrate the procedure of using the fully-resolved model to obtain macroscopic variables. This procedure is usually very time consuming and may be impossible to perform due to limitations in computational resources. For this reason, we only solve the fully-resolved model for the problem of interest in the first numerical example in Section 5.4 for validating the database approach. The term “average” above the second arrow on the upper part of Fig. 5.4 is used to represent the process of obtaining macroscopic variables including temperature, liquid volume fraction and microstructure features as discussed in Section 5.1.2. However, due to the high computational cost, in the numerical example with validation, only the macroscopic temperature T is computed following its definition (i.e. by averaging microscopic temperature obtained from different sampling of potential nucleation sites). For validation of the volume fraction f at a given time t, we will compare the solid-liquid interface position at time t (obtained by the fully-resolved model) with the volume fraction contour lines (obtained by the database approach). Also for validation of the microstructure features Λ at a given location x, we will compare the microstructure at location x obtained by the fully-resolved model with the microstructure obtained by the database approach (i.e. by searching in the database to obtain the microstructure with solution features nearest to $F_M(x)$).

The first step in the above algorithm can be understood as a ‘prediction step’, since the temperature is predicted as θ_M using model M. The second step gathers information on volume fraction and microstructure features from the microscale computations in the sample problem. The third step can be understood as a ‘correction step’, since the temperature is corrected with updated information on volume fraction.

The boundary and initial thermal conditions for the algorithm discussed up to now are the same for the problem of interest when it is solved with model M, the fully-resolved model (e.g. in our first example in Section 5.4.1) or the multiscale model. However, to increase the accuracy of the methodology while maintaining the simplicity of model M, in our multiscale implementation, model M is applied only for a part of the

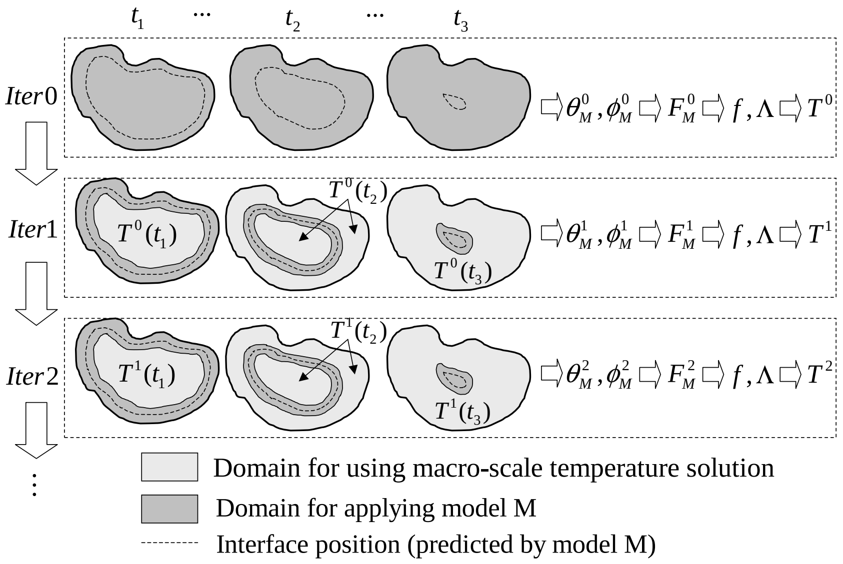


Figure 5.5: Schematic of the overall algorithm.

domain around the interface. The details of the overall algorithm are discussed next.

5.2.7 Overall multiscale algorithm

The procedure discussed in the previous section assumes that the selected sample problem is *consistent* at any location x of the domain of the problem of interest with model M. This is a strong assumption especially for a problem with large domain. The model M is not capable to very accurately model the evolution of temperature. At the beginning of solidification, the error in temperature (i.e. the difference to temperature computed by the fully-resolved model) is small. So the above procedure can predict the solidification microstructure quite well for the early stages of solidification. However, as the error in temperature obtained using model M (i.e. θ_M) becomes larger with time, the above procedure becomes less accurate for predicting the obtained solidification microstructure.

In order to deal with this difficulty, we need to find a way to constrain the error accumulation in model M by utilizing the temperature data obtained (from an earlier iteration) from solving the macroscale model (by solving the heat diffusion equation with a latent heat term $-\rho L\dot{f}$). The iterative method we used is demonstrated in Fig. 5.5. For the first iteration (iteration 0 in Fig. 5.5), the procedure is as before. But at later iterations, we apply model M only on the region near the solid-liquid interface (darker region in Fig. 5.5). At places far away from the solid-liquid interface (lighter region in Fig. 5.5), we use the temperature from the macroscale model obtained in the previous iteration. In this way, the temperature from the macroscale model, is always applied as boundary condition for the region near the solid-liquid interface (i.e. darker region in Fig. 5.5) on which model M is solved. Using temperature from the macroscale model as boundary condition constrains the error of model M from accumulating.

In this approach, we need to define the region for applying model M (i.e. darker region in Fig. 5.5). Since model M gives a signed distance field ϕ_M, we have the information of distance to the solid-liquid interface at any location. We can introduce a parameter L_M to specify the size of the region for applying model M. For the region within distance L_M away from the solid-liquid interface (i.e. $|\phi_M| \leq L_M$), we apply model M. Otherwise (i.e. $|\phi_M| > L_M$), we use the temperature data from the macroscale model obtained in the previous iteration.

The following is the procedure for implementation of this algorithm:

1. Set iteration number $i = 0$.
2. Solve model M to obtain θ_M^i, ϕ_M^i. If $i > 0$, essential boundary condition $\theta_M = T^{i-1}$ is applied at nodes with $|\phi_M| > L_M$ when solving for the temperature field with model M. Obtain solution features F_M^i for every node point.

3. For solution features F_M^i at each node of the grid for the whole domain, find $f(T, F_M)$ and $\Lambda(F_M)$ by solving the appropriate sample problems.
4. Obtain solution of temperature field T^i by solving $\rho c \frac{\partial T}{\partial t} = k\nabla^2 T - \rho L \dot{f}$ with the updated liquid volume fraction f.
5. Set $i = i + 1$ and return to step 2 (three iterations were typically sufficient in the numerical examples considered).

This algorithm uses solution of the macroscale model to improve our prediction of f, Λ and correspondingly of T. Comparing with the algorithm without iterations, the additional cost we pay is the computation time for the required iterations and the storage for T^{i-1}. To perform iteration i, we need to track the solution of T^{i-1} on the whole domain (for each node) and at all times (for each time step). Since we are using coarse grid spacing and large time steps for T (macroscale computation), this additional storage is quite affordable.

5.3 Numerical implementation

5.3.1 Reducing the number of the needed sample problems using interpolation in the feature space

Solving the sample problem using the fully-resolved model is computational intensive due to the required fine mesh. So in order to speed up the computation, we want to minimize the number of times for solving sample problems. The idea of interpolation is shown in Fig. 5.6. We only need to run a sample problem on each node of the mesh (triangular mesh is shown in the schematic, but quadrilateral mesh is also used in the examples) generated in the feature space (G^l_M v.s. V_M instead of x v.s. y). For evaluation of f and Λ at an arbitrary feature, we will do interpolation.

As shown in Fig. 5.6, we first obtain F^x_M on each node (macroscale grid) using model M. The relation between temperature and liquid volume fraction at features F^x_M (i.e. $f(T, F^x_M)$) in general is not available in the database. However, we can find a few such relations at features close to F^x_M (as demonstrated in the top left triangle) in the database. They will be used to interpolate $f(T, F^x_M)$. We need to do this interpolation and store the interpolated liquid volume fraction f for all nodes. After this, solving the macroscale model requires the solution of the diffusion equation with a latent heat term determined by the interpolated f.

5.3.2 Storing sample problem results

In the database approach, there is a huge amount of data to be processed. For example, in our first numerical example in Section 5.4, if a uniform mesh is used, the required mesh size is 1024×256 with around 0.25 million nodes. If a CFL coefficient 0.1 is used, then we will need about $1024/0.1 \approx 10,000$ time steps. On each node and at each time step, we have variables such as C, θ, I, and ϕ. So we will have around $0.25M \times 10,000 \times 4 = 10G$ float numbers as results of just one sample problem. For efficiency in storing data and extracting useful information from data, we want to store the minimum amount of data. Another difficulty in storing sample problem results is related with the adaptive meshing technique we used for solving sample problems on the microscale. Computation is performed on an adaptive mesh, while statistics must be extracted using a uniform mesh.

To deal with these difficulties, during computation of the sample problem, at each

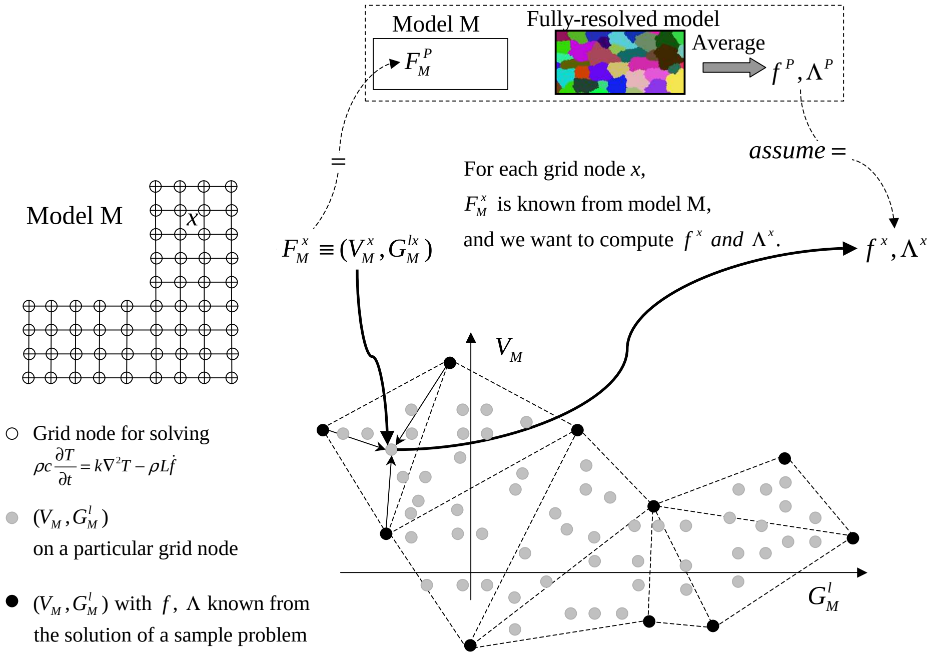


Figure 5.6: Schematic of using interpolation to reduce the number of the required sample problem solutions. The top shows how we can use the time consuming fully-resolved model to evaluate f and Λ corresponding to an arbitrary feature. The bottom shows all obtained solution features (gray dots), a mesh generated in the feature space and how interpolation is used to obtain f and Λ based on results of sample problems corresponding to nodes of the mesh (black dots).

time step, we store the location x, current time (solidification time) t_s, solidification temperature (temperature at x at the time t_s) θ_s and orientation angle I for each node only at the time it is solidified (i.e. $\phi^{n-1} > 0$ and $\phi^n \leq 0$). Since the smallest grid spacing Δx is used for the region within some distance ($3\Delta x$ in our computations) away from the solid/liquid interface in both solid and liquid sides of the interface, each node i in the equivalent uniform mesh (i.e. structured mesh with grid spacing Δx) will have data (including I, θ, C) directly obtained from finite element computation when $-3\Delta x \leq \phi_i \leq 3\Delta x$. If $\phi_i > 3\Delta x$ or $\phi_i < -3\Delta x$, then data (including I, θ, C) may have to be obtained from interpolation, since the node may not exist in the adaptive mesh. During our computation, the selected CFL coefficient is always less than 1. Therefore, each node in the equivalent uniform mesh exists in the adaptive mesh for the time interval within which it is solidified. In this way, each node in the equivalent uniform mesh will have data (i.e. x, t_s, θ_s and I) recorded exactly once, even if adaptive meshing is used for computation. We keep appending data (x, t_s, θ_s and I) to a file, until all nodes in the adaptive mesh are solidified (i.e. with $\phi < 0$). These recorded data basically contain information of t_s, θ_s and I for all nodes in the equivalent uniform mesh. However, the data is sorted by solidification time (not by location), since each row of data is recorded at the time step during which the corresponding node is solidified (i.e. $\phi^{n-1} > 0$,and $\phi^n \leq 0$). So after computation, we read the recoded file and obtain the t_s, θ_s and I fields on the equivalent uniform mesh according to the location information x.

Although thousands of time steps are usually involved for a sample problem, the three fields ($t_s(x)$, $\theta_s(x)$ and $I(x)$) provide all necessary information for the multiscale model. The utilization of the fields $\theta_s(x)$ and $I(x)$ to provide an estimation of the liquid volume fraction f and microstructure features F_M was already discussed in Section 5.2.4.

Finally, note that the field t_s contains information of the interface position at any time level. The contour line of the field t_s gives the position of the solid-liquid interface at the corresponding time level (contour line value) as shown in Fig. 5.7.

5.3.3 Other implementation details

Note that the potential nucleation sites generally are not at the finite element nodes. In the implementation of the nucleation algorithm and for storing the location of potential nucleation sites, we assign a list to each element which contains the locations of all po-

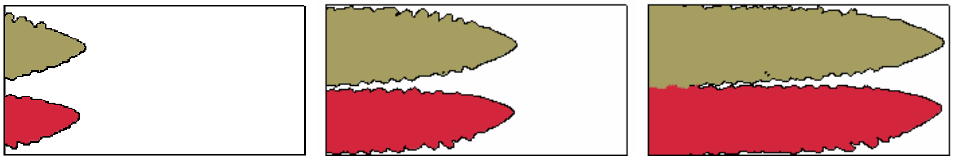


Figure 5.7: Contour line of field t_s at value $t_s = 250$, $t_s = 325$, $t_s = 400$ for one of the sample problems with $V_M = 0.02281344$ and $G^l_M = 0.6708713$ discussed in the numerical examples section. Regions of $t_s \leq 250$, $t_s \leq 325$ and $t_s \leq 400$ are colored with orientation angle to identify two different crystals.

tential nucleation sites sampled inside this element. This list may be empty, if there is no potential nucleation site inside this element or very long if there are lots of potential nucleation sites inside it. For determination of whether a potential nucleation site nucleates, we do interpolation (using data on the nodes of the element which contains the potential nucleation site) to find the actual undercooling at the potential nucleation site, and compare it with the required undercooling. One also needs to pay attention to the potential nucleation sites while remeshing. If an element is divided into a few sub-elements, each potential nucleation site inside the parent element is assigned to the sub-element which contains it.

5.4 Numerical examples

5.4.1 Verification of the database approach

The following dimensionless material properties are considered in this section: potential nucleation site density $\rho^n = 100$, the required undercooling for nucleation $\Delta\theta_n$ satisfies a normal distribution $N(1.5, 0.2^2)$, density $\rho = 1$, heat capacity $c = 1$, heat diffusion coefficient $k = 1$, latent heat $L = 100$, Lewis number $Le = 100$, melting temperature

$\theta_m = 0$, initial concentration $C_0 = 0.1$, liquidus slope $m_l = -10$, partition coefficient $k_p = 0.1$, Gibbs-Thomson relation with anisotropy in kinetic undercooling $\theta_* = \theta_m + m_l C^l - \frac{2}{3}\{1 - \cos[4(\alpha - I)]\}V$, where α is the angle between the positive x axis and the interface normal direction. Notice that the large latent heat $L = 100$ makes the problem nontrivial on the macroscale, whereas the large Lewis number $Le = 100$ makes the problem nontrivial on the microscale. The computational domain is of size 40×40. However, the left-top quarter is not included making the computational domain irregular. Initially, the whole domain is at temperature 10. The right side is assumed to be adiabatic. The temperature applied on all other sides is taken as $T_b = 50\exp(-t/10) - 40$ or $T_b = 100\exp(-t/20) - 90$ in two different runs in order to study the effects of the boundary conditions on the obtained solidification microstructure.

Computational results using model *M*

Using an adaptive mesh with smallest grid spacing 0.1563, we obtain solution features of V_M and G^l_M for temperature boundary condition $T_b = 50\exp(-t/10) - 40$ and $T_b = 100\exp(-t/20) - 90$ as shown in Fig. 5.8. Using 4 computation node (each with $2 \times 3.8G\ CPUs$), the required computational time was around 1 hour. By comparing the results for the two cases, we find that the growth velocity V_M for the case with $T_b = 50\exp(-t/10) - 40$ is much lower than for the case with $T_b = 100\exp(-t/20) - 90$, whereas the temperature gradient G^l_M is only slightly smaller.

Results of sample problems

Fig. 5.9 plots the computed V_M and G^l_M for all locations in both runs (with temperature boundary condition $T_b = 50\exp(-t/10) - 40$ and $T_b = 100\exp(-t/20) - 90$) in the

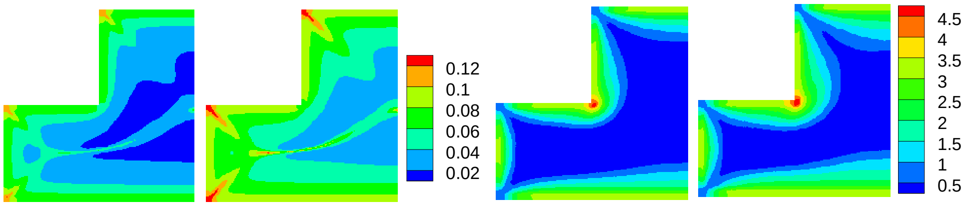


Figure 5.8: Contour of V_M (left two plots) and G^l_M (right two plots) for temperature boundary condition $T_b = 50\exp(-t/10) - 40$ (the 1st and the 3rd plots) and $T_b = 100\exp(-t/20) - 90$ (the 2nd and 4th plots).

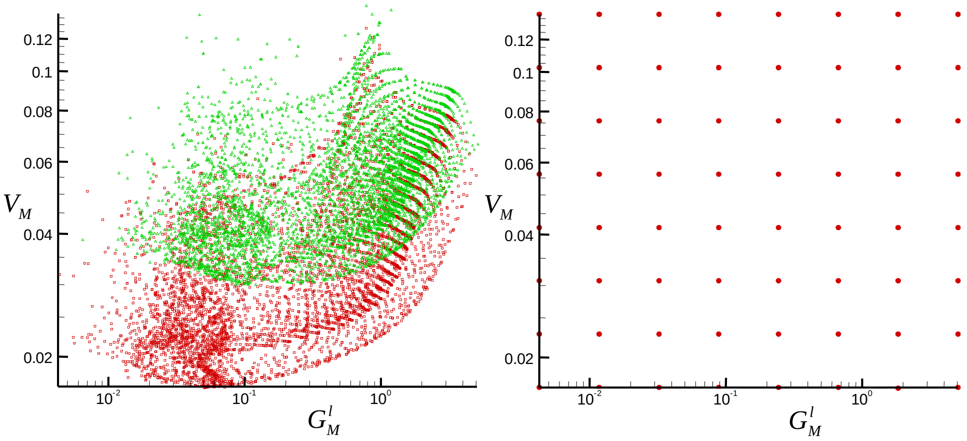


Figure 5.9: Left: obtained features of model M, (V_M and G^l_M) with temperature boundary condition $T_b = 50\exp(-t/10) - 40$ (red square symbols) and $T_b = 100\exp(-t/20) - 90$ (green triangle symbols); Right: (V_M, G^l_M) of 64 sample problems selected for applying the fully-resolved model.

feature space. As discussed in Section 5.3.1, we can use interpolation to obtain the liquid volume fraction and the microstructure features for each of the features in the left plot of Fig. 5.9. Sixty four sample problems are selected to give data for interpolation. The features (V_M and G^l_M) of these 64 sample problems are selected to be uniformly distributed in the log scale as shown as shown on right plot of Fig. 5.9. Recall that both analytical and experimental results show that microstructure features are in the power form of growth velocity and thermal gradient (i.e. $\sim V^\alpha G^\beta$ with α, β constants).

For each sample problem, we use a domain size of 10×2.5 with smallest grid spacing

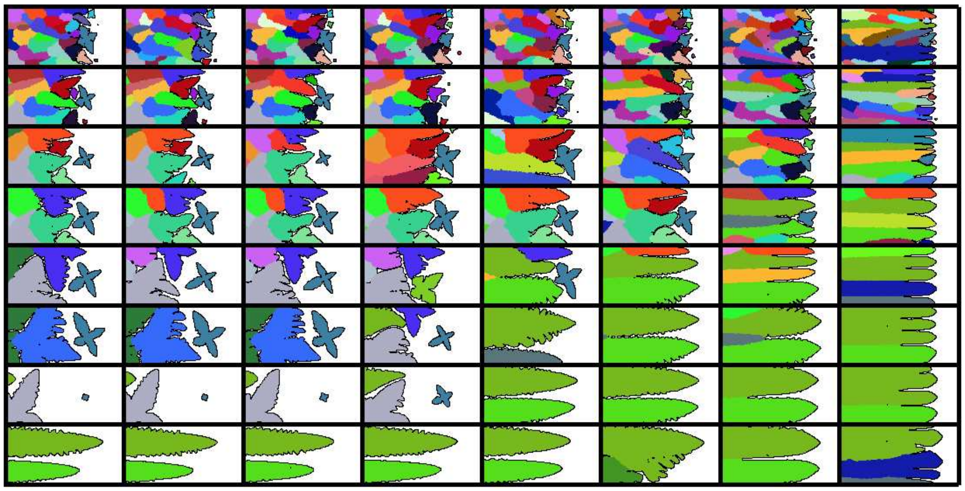


Figure 5.10: Obtained microstructure of 64 sample runs. Each rectangle corresponds to a feature (V_M, G^l_M) shown in the right plot of Fig. 5.9.

0.0098 for adaptive meshing. The selected grid spacing 0.0098 is much smaller than the grid spacing 0.1563 used for running model M. Using a CFL coefficient 0.125, it takes around 8000 time steps to perform the simulation for each sample problem. We use a cluster with 64 nodes (each node with $2 \times 3.8G\ CPUs$) to compute these sample problems. Each node is used to perform computation of one sample problem. The computational time is around 5 hours. Results of these sample problems in the domain for statistical analysis (as demonstrated in Fig. 5.3) are shown in Fig. 5.10. From results of these sample problems, we observe a continuous transition between equiaxed growth and columnar growth with various grain sizes. This study is similar to the previous numerical studies [5, 63], in which computations are performed at various thermal gradients and growth velocities. In [5, 63], the thermal gradient G and growth velocity V are assumed to be constant, which are only applicable to directional growth. There is no relation established between the performed numerical study at a particular growth speed V and thermal gradient G with other problems. However, in this work, the thermal gradient G^l_M and growth velocity V_M are obtained from the solution of model M for the problem of interest. The sample problems are *relevant* to our problem of interest

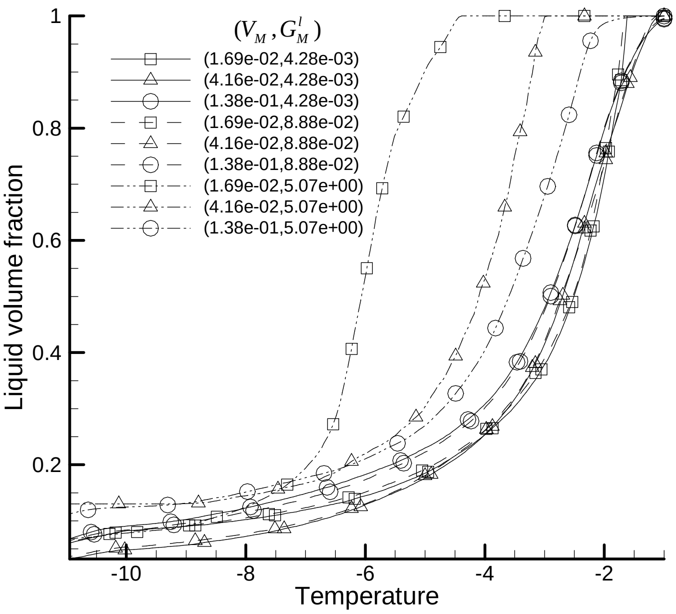


Figure 5.11: Obtained relation between the liquid volume fraction and temperature from sample problems with various features F_M (64 sample problems are computed but only 9 are plotted here).

with model M. We will utilize the information obtained from the microscale solution of sample problems to solve the problem of interest by introducing the assumption that the sample problem is also *consistent* to our problem of interest with model M.

Notice that the microstructure pattern (obtained from the fully-resolved model) over the whole sample problem domain is rather uniform for each sample problem. This is a consequence of constant solution features (from model M) over the sample problem domain.

Results of the database approach

From the results of model M, we obtain $F_M(x)$ on the whole domain. The computed relation between the liquid volume fraction f and temperature T for the 64 sample problems considered is shown in Fig. 5.11. Using interpolation based on the obtained relation of

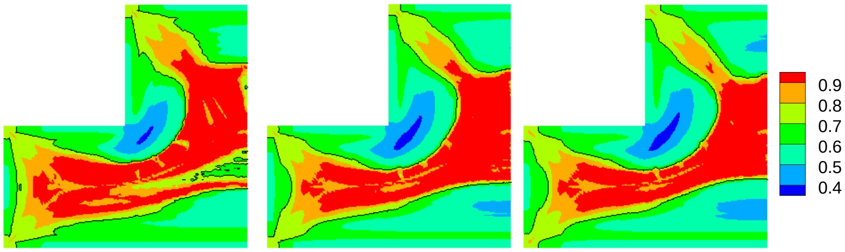


Figure 5.12: Predicted field of $\frac{N_x}{N_y}$ at the 1st, 2nd, and 3rd iteration for the case with $T_b = 100\exp(-t/20) - 90$. Contour line with value 0.7 demonstrates the predicted location of columnar to equiaxed transition (CET).

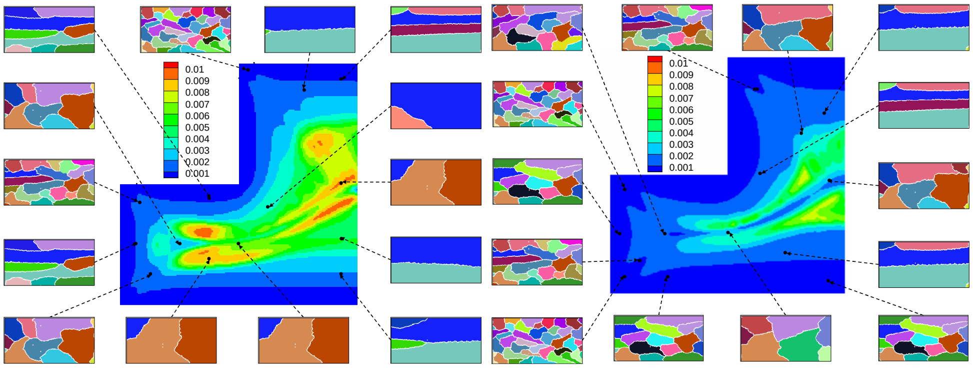


Figure 5.13: Field of $\frac{1}{N_y}$ (The 14 microstructures shown correspond to the closest microstructure in the database). Left: $T_b = 50\exp(-t/10) - 40$; Right: $T_b = 100\exp(-t/20) - 90$.

the liquid volume fraction and temperature for the sample problems with different F_M (i.e. $f(T, F_M)$), we are able to find the liquid volume fraction $f(x,t)$ for solving the macroscale model. The temperature obtained from the macroscale model can then be used to improve the computation of $F_M(x)$. Fig. 5.12 demonstrates how the predicted microstructure feature $\frac{N_x}{N_y}$ changes with iterations. The field of $\frac{N_x}{N_y}$ at the 2nd iteration is very similar to the field of $\frac{N_x}{N_y}$ at the 3rd iteration demonstrating that the solution has converged after three iterations. In our computation, L_M is selected to be 5.

With the results of the 64 sample problems (database) in place and the obtained V_M, G_M^l for the problem of interest, we can now perform interpolation to obtain the mi-

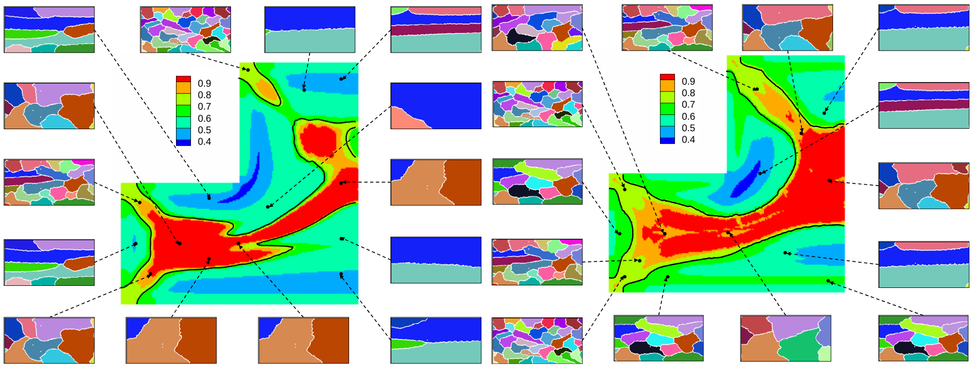


Figure 5.14: Field of $\frac{N_x}{N_y}$ (The 14 microstructures correspond to the closest microstructure in the database). Left: $T_b = 50\exp(-t/10) - 40$; Right: $T_b = 100\exp(-t/20) - 90$.

crostructure size $\frac{1}{N_y}$ and microstructure type $\frac{N_x}{N_y}$ as demonstrated in Figs. 5.13 and 5.14. Using a uniform mesh with grid spacing 0.3125 and a fixed time step of 0.1, the computation time with 1 computation node ($2 \times 3.8G\ CPUs$) is around 20 minutes for the case with boundary condition $T_b = 50\exp(-t/10) - 40$ and around 10 minutes for the case with boundary condition $T_b = 100\exp(-t/20) - 90$.

In order to give a picture of how the microstructure may appear at a particular location with features $(V_M(x), G^l_M(x))$, we can search in the database to find the microstructure with the features closest to $(V_M(x), G^l_M(x))$ as demonstrated in the 14 microstructures in Figs. 5.13 and 5.14.

From the contour of field $\frac{1}{N_y}$ or the closest microstructures in Fig. 5.13, we can predict that large grains will be formed in the center of the casting. The case of the boundary condition $T_b = 50\exp(-t/10) - 40$ leads to larger grains than the case with boundary condition $T_b = 100\exp(-t/20) - 90$.

The contour line in Fig. 5.14 corresponds to $\frac{N_x}{N_y} = 0.7$. It is the predicted location of columnar to equiaxed transition. In both cases, it is predicted that we will obtain equiaxed microstructures at the corners and also at the center of the domain.

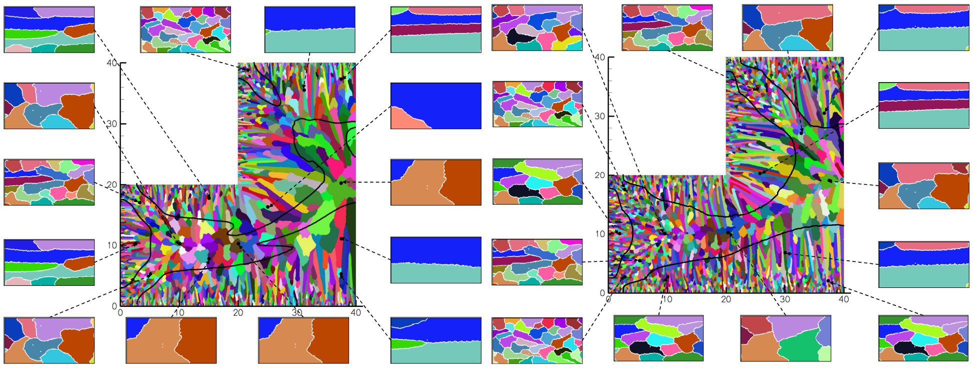


Figure 5.15: Comparison of the predicted microstructures using the database approach with the microstructures obtained from solving the problem in the whole domain using the microscale model. Left: $T_b = 50\exp(-t/10) - 40$; Right: $T_b = 100\exp(-t/20) - 90$. For each plot (left or right): the picture in the middle is the fully-resolved result; the dark line in the middle picture is the predicted location of CET transition using the database approach ($\frac{N_x}{N_y} = 0.7$); the 14 pictures (around the middle picture) are the closest microstructure in the database based on features F_M at selected locations.

Comparison of microstructure features and liquid volume fraction obtained from the microscale model and the database approach

As a validation of the database approach, we also performed the microscale model on the whole domain with smallest grid spacing 0.0098 for adaptive meshing. The CFL coefficient is selected to be 0.125. Using 8 computation nodes (each node with $2 \times 3.8G\ CPUs$), the total computation time for each case is about 2 days. For validation of the database approach, the microstructure features $\Lambda(x)$ are not directly compared with the microscale results. We perform the comparison in the following way: (1) solve the microscale model to obtain microstructure details all over the domain, (2) obtain Λ using the database approach, pick up a few locations in the domain, search the closest microstructure in the database for these locations, and (3) for each selected location, compare the searched microstructure (from the database) and the microstructure obtained using the microscale model at the corresponding location in the first step.

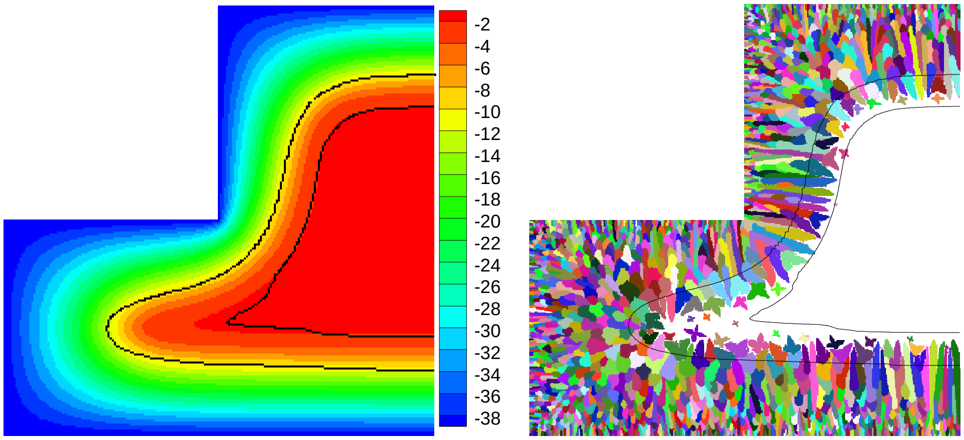


Figure 5.16: Left: Predicted temperature field and liquid volume fraction contours with values 0.95 and 0.05 at time 130 for the case with $T_b = 50\exp(-t/10) - 40$. Right: Obtained microstructure (using microscale model) and liquid volume fraction contours with value 0.95 and 0.05 (using the database approach) at time 130 for the case with $T_b = 50\exp(-t/10) - 40$.

Such a comparison for both cases (different boundary conditions) is demonstrated in Fig. 5.15. Although this type of comparison is only a qualitative one, it demonstrates that the database approach is capable of predicting microstructure patterns quite well with significantly less computational cost.

At time 130 for the case with $T_b = 50\exp(-t/10) - 40$, the solid-liquid front (using the microscale model) falls almost exactly inside the region with volume fraction $0.05 \leq f \leq 0.95$ (using the database approach) as shown in Fig. 5.16. This suggests that the database approach is also accurate for predicting volume fractions.

Comparison of the temperature field obtained from the microscale model and the database approach

As discussed earlier, the macroscopic temperature obtained from the database approach is the expectation of microscopic temperature (i.e. $T(x,t) =< \theta(x,t) >$). In or-

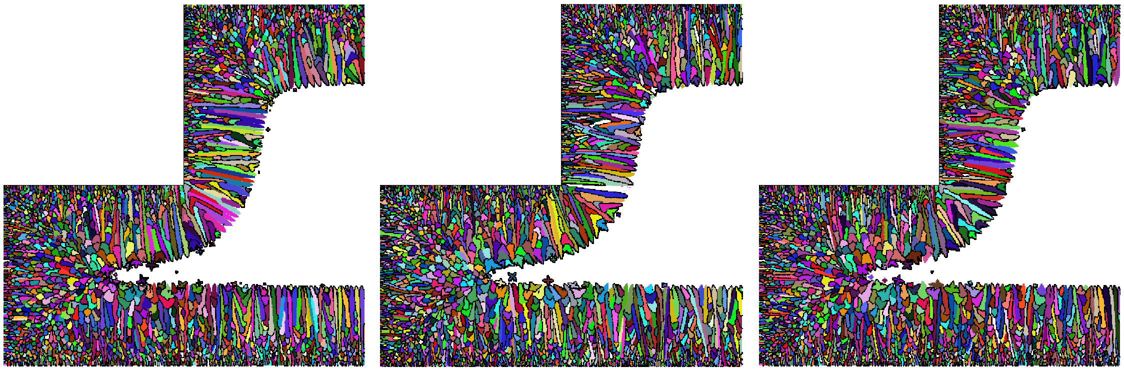


Figure 5.17: Microstructure at time 81.6 for the case with $T_b = 100\exp(-t/20) - 90$ using different sampling of potential nucleation sites.

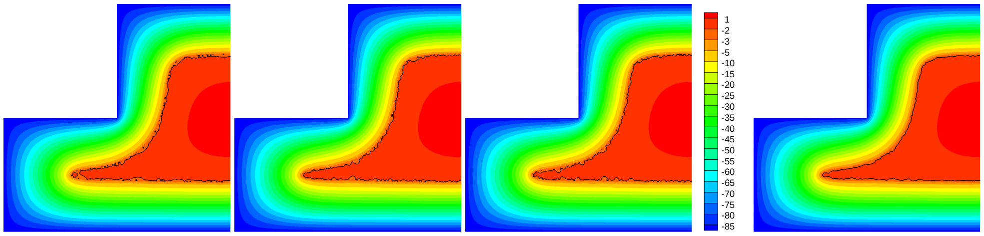


Figure 5.18: Left 3 plots: Microscopic temperature at time 81.6 for the case with $T_b = 100\exp(-t/20) - 90$ using different sampling of potential nucleation sites; Right plot: Averaged microscopic temperature.

der to obtain the expectation of microscopic temperature for comparison with macroscopic temperature, we performed three microscale computations for the case with $T_b = 100\exp(-t/20) - 90$ using different sampling of potential nucleation sites. The microstructure at time 81.6 for these three computations is shown in Fig. 5.17. Because the potential nucleation sites in these three microscale computations are different, the obtained microstructural details are also different. Correspondingly, the microscopic temperature is also different as shown in Fig. 5.18.

In the database approach, we have approximated that $f(x,t) = f(T(x,t), F_M(x))$ instead of using $f(x,t) = f(T(x,t))$. Two questions may be raised regarding this approximation. First, will $f = f(T, F_M)$ lead to significantly different results (volume fraction and temperature) from $f = f(T)$?

Secondly, is the approximation $f = f(T, F_M)$ good enough for obtaining similar results to those from the fully-resolved model. It is very obvious that F_M has a great effect on volume fraction as shown in Fig. 5.11. For the case with $T_b = 100\exp(-t/20) - 90$, we computed the temperature field at time 81.6 using the fully-resolved model (after averaging among the three microscale computations), using the database approach ($f = f(T, F_M)$) and using the level rule ($f = f(T)$) with results shown in Fig. 5.19.

The temperature field obtained from the database approach is very similar to the temperature field obtained from the microscale model except that small scale variations of temperature exist in the results of the microscale model, which is due to nucleation and growth of equiaxed crystals and the variation in dendrite front position.

The temperature field obtained from the Lever rule with approximation $f = f(T)$, however, is quite different from the microscale model result. The results in Fig. 5.16 and Fig. 5.19 demonstrate that the approximation of $f = f(T, F_M)$ can successfully incorporate the effects of microstructure morphology on volume fraction and temperature.

5.4.2 Application to the solidification of an Al-Cu alloy

In the previous numerical example, the material parameters were selected to demonstrate and validate the multiscale modeling approach. The domain size of the interested problem and domain size of the sample problem were selected to be around 40×40 and 10×2.5. The material parameters and domain size were selected in the above example so that it will be possible to perform fully-resolved computations in the microscale to

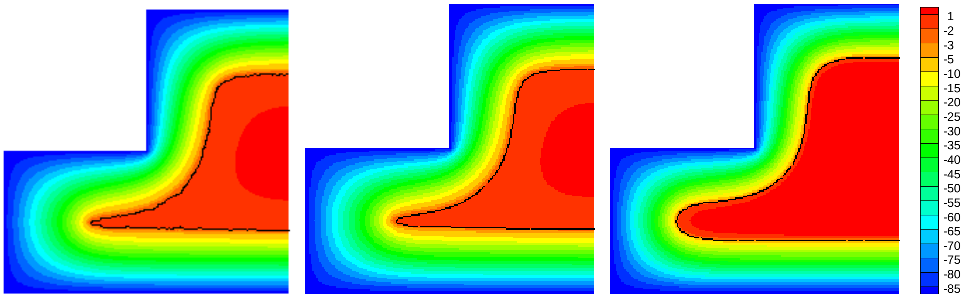


Figure 5.19: Temperature field at time 81.6 for the case with $T_b = 100\exp(-t/20) - 90$. Contour line shows the position where the temperature is -2. Left: Temperature field obtained from the microscale model by averaging among three computation results; Middle: Predicted temperature field from the database approach; Right: Predicted temperature using the Lever rule.

validate the multiscale model results.

In this section, we will use an Al-Cu alloy with material parameters provided in [5]: partition coefficient $k_p = 0.14$, liquidus slope $m_l = -2.6K/wt.\%$, solute diffusion coefficient $D_l = 3000\mu m^2 s$, melting temperature $T_m = 933.47K$, Gibbs-Thomson relation coefficient $\epsilon_c = -0.24K \cdot \mu m$, surface tension anisotropy $\varepsilon = 0.01$, density of potential nucleation sites d_s^{-2} with $d_s = 9.7\mu m$. Here, d_s is the potential nucleation site distance. We model the required undercooling for nucleation as a Gaussian random variable with mean $7.5K$ and standard deviation $1.25K$.

The domain size of the sample problem should be in the order of $10d_s$ to $100d_s$ to ensure that enough crystals are included to capture the overall behavior. The grid spacing for the fully-resolved model should be in the order of $0.1d_s$ to ensure that microstructure details can be computed. To satisfy these two constraints, we select our sample problem size as $120\sqrt{2}d_s \times 36\sqrt{2}d_s$ and grid spacing for the fully-resolved model as $0.8574\mu m$.

Other parameters related with heat transfer (which is not considered in [5]) are: density $\rho = 2400kg/m^3$, heat capacity $c = 1.06KJ/(kg \cdot K)$, heat diffusivity $k =$

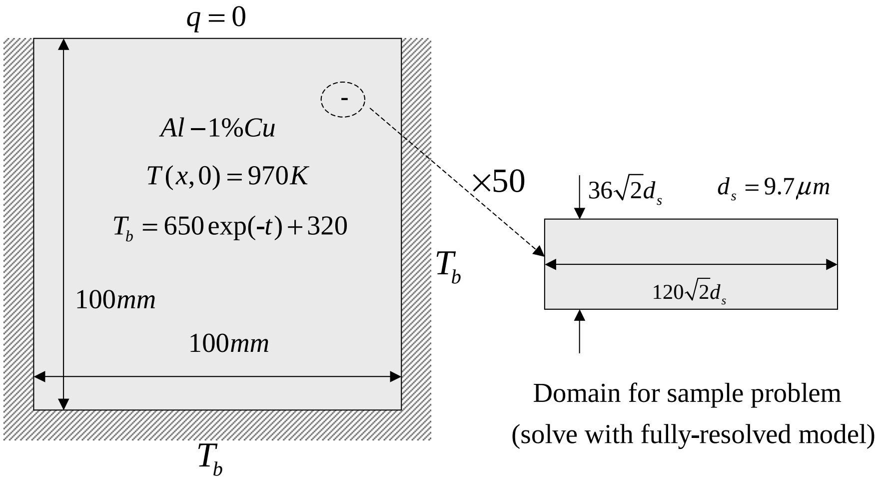


Figure 5.20: Schematic of the solidification of an Al-Cu alloy. The very small rectangle inside the elliptic shape is used to demonstrate the relevant size of the domain of the sample problem versus the size of the domain of the problem of interest. It is magnified by 50 times in the right plot.

$82.61W/(m \cdot K)$, and latent heat $L = 397.5KJ/kg$. We consider a square region with side length $100mm$ (which is the typical size of a casting) filled with the Al-1%Cu alloy at an initial temperature of $970K$. The top side is adiabatic (i.e. $q = 0$), while other three sides are kept at temperature $T_b = 650\exp(-t) + 320$. A schematic of this example is provided in Fig. 5.20.

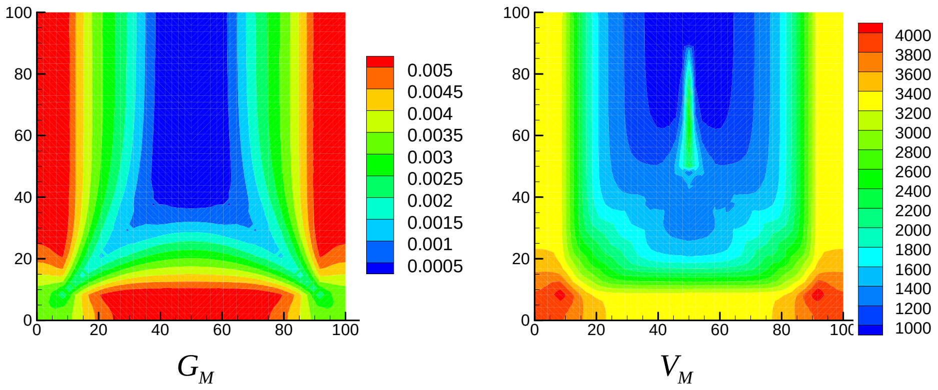
Our aim is to obtain the microstructure pattern after solidification and also the temperature field during solidification. If a fully-resolved model is used, the estimated number of grid nodes is $(\frac{100mm}{0.8574\mu m})^2$ with value about $14G$. If a CFL coefficient 0.1 is used, the estimated number of time steps is about $10\times(\frac{100mm}{0.8574\mu m})$ with value $1.2M$. $1.2M$ time steps and $14G$ degrees of freedom is almost impossible to solve with current computational resources. Therefore, we will only provide results of the database approach for this example.

As in the first example, we first use model M to obtain solution features i.e.

(V_M, G_M). Using symmetry of the problem, an uniform mesh with size 192×384 is used. Around 30 minutes are required to perform the computation using model M using one computation node (3.8G×2CPU). The obtained fields of V_M and G_M are shown in Fig. 5.21. By plotting the obtained data in the V_M and G_M coordinates, we obtain the left plot in Fig. 5.22. A mesh with 11 nodes is generated to occupy roughly the same region as the obtained (V_M, G_M) data points. For each node of the mesh, we are going to solve a sample problem using the fully-resolved model. An adaptive mesh (equivalent to a uniform mesh with size 1920×576) is used for solving the sample problem. Note that in this example, periodic boundary conditions are applied for heat transfer, solute transport, level set evolution when solving the sample problem with the fully-resolved model as demonstrated in Fig. 5.23. Each sample problem takes about 36 hours using one computation node (3.8G×2CPU). 11 computation nodes are utilized to solve simultaneously all 11 sample problems. Results of a few typical sample problems are shown in Fig. 5.24.

As shown in Fig. 5.25, the liquid volume fractions for these sample runs are different from the one predicted by the Lever rule, which is widely used in macroscale solidification models. With liquid volume fraction f obtained for each of the sample problems, we are able to use interpolation to obtain f for any given (V_M, G_M) and solve the macroscale model (i.e. $\rho c \frac{\partial T}{\partial t} = k\nabla^2 T - \rho L \dot{f}$) and perform iterations to improve accuracy of the solution.

It takes about 1 hour to finish two more iterations using one computation node (3.8G×2CPU). The predicted temperature at time 12.7 seconds is shown in the right half of Fig. 5.26. Comparing with the predicted temperature using Lever rule (shown in the left half of Fig. 5.26), the temperature in the center of the domain is about 5 K lower. This difference is expected since the volume fraction using the database approach is



G_M V_M

Figure 5.21: Obtained G_M (left) and V_M (right) fields using model M. Units of axes, G_M and V_M are mm, $K/\mu m$ and $\mu m/s$, respectively.

significantly different from the predicted volume fraction using the Lever rule. Finally, as a post-processing procedure, we can search in the database according to the fields of V_M and G_M to obtain the microstructure. Microstructure in the database with nearest solution features are given in Fig. 5.27 for 8 locations as the location approaches from the side to the center (A to B to C to D in Fig. 5.27). The microstructure pattern changes from fine columnar to coarse columnar and then to coarse equiaxed. As the location approaches from the corner to the center (E to F to G to H in Fig. 5.27), the microstructure pattern changes from fine equiaxed to coarse equiaxed.

5.5 Conclusions

In this work, we utilize three models for an efficient study of solidification. A computationally efficient model is used to select relevant sample problems. A microscale model is used to generate the database by solving the selected sample problems. A macroscale model is used to efficiently compute solidification with inputs from the database. The results of the macroscale model are further utilized to improve the evaluation of vol-

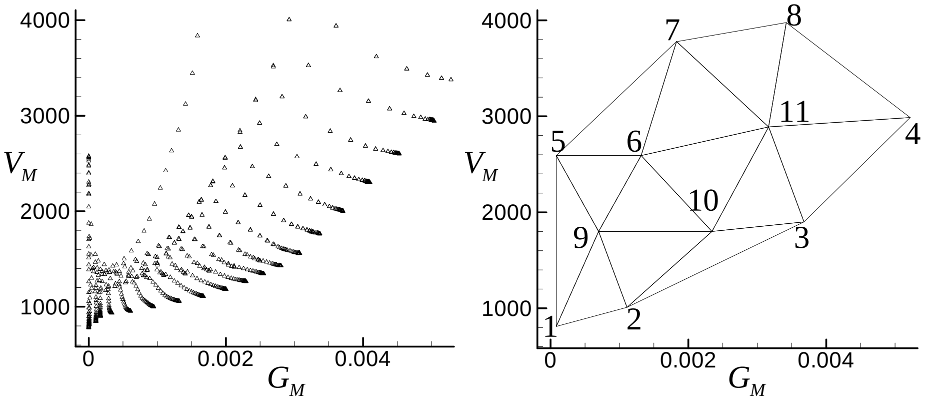


Figure 5.22: Left: Obtained solute features in the V_M and G_M coordinates; Right: V_M and G_M for the sample runs. Units of G_M and V_M are $K/\mu m$ and $\mu m/s$, respectively.

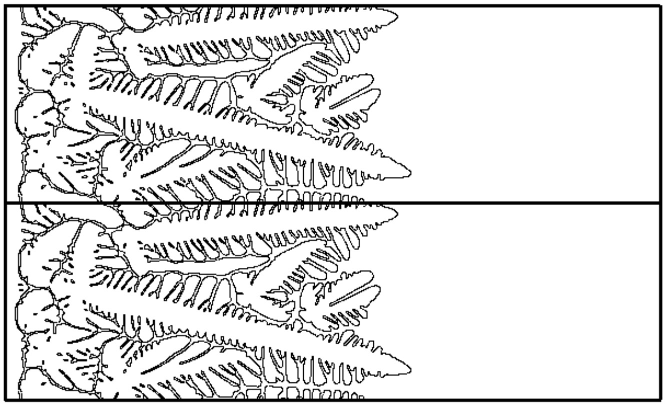


Figure 5.23: Demonstration of sample problem domain with periodic boundary conditions applied at the top and bottom sides. The bottom half is the computational domain, the top half is just a copy of the solution from the bottom.

ume fraction and microstructure features. In other multiscale solidification studies in literature, only two models are used: a microscale model to capture microstructure details and a macroscale model to capture global effects. The computationally efficient model plays a crucial role in our multiscale framework. An analytical solution of the computationally efficient model is utilized to define *relevant* sample problems based on solution features obtained from the computationally efficient model. Interpolation

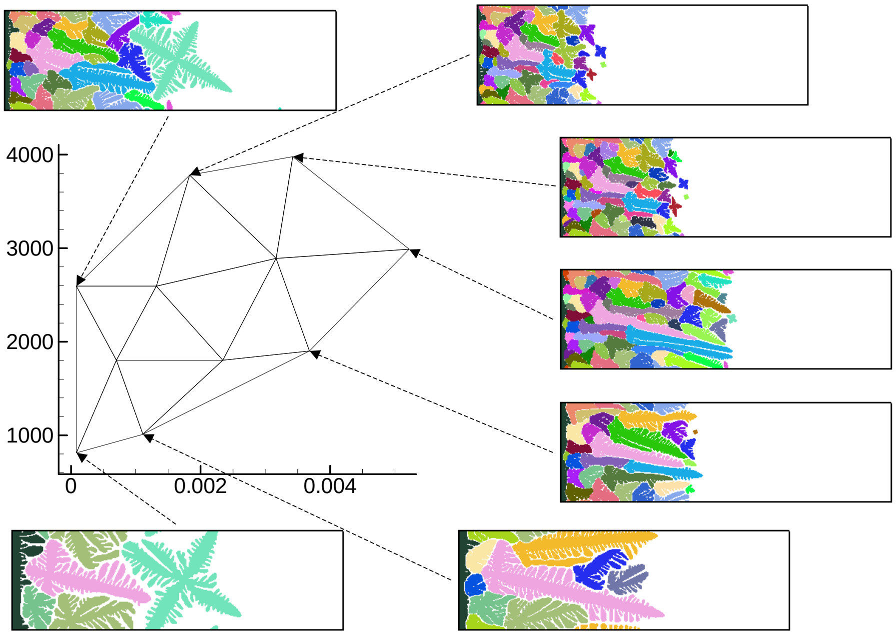


Figure 5.24: Sample problem results using the fully-resolved model.

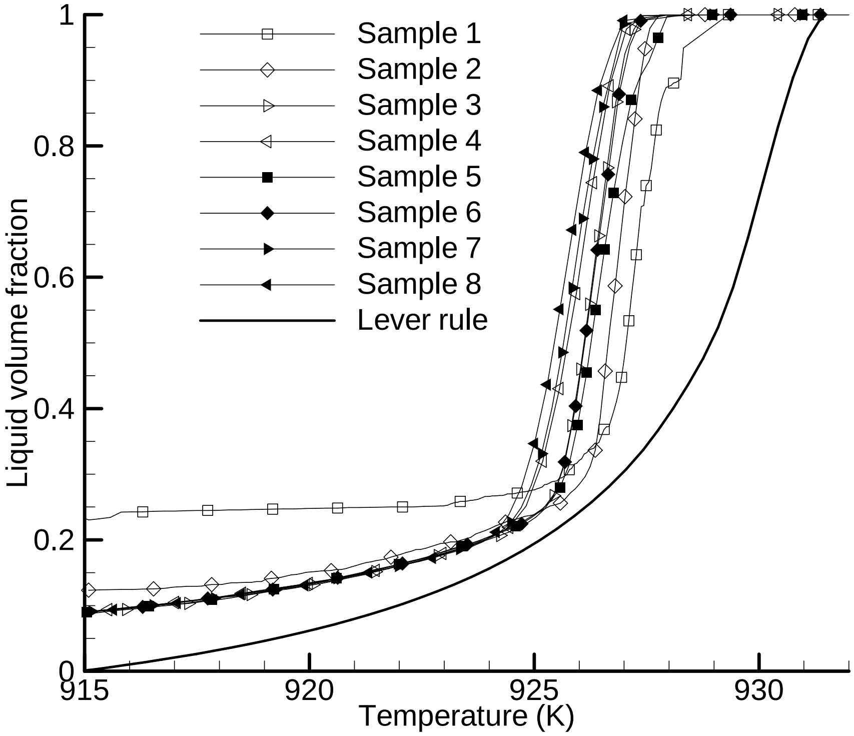


Figure 5.25: Relation of volume fraction and temperature.

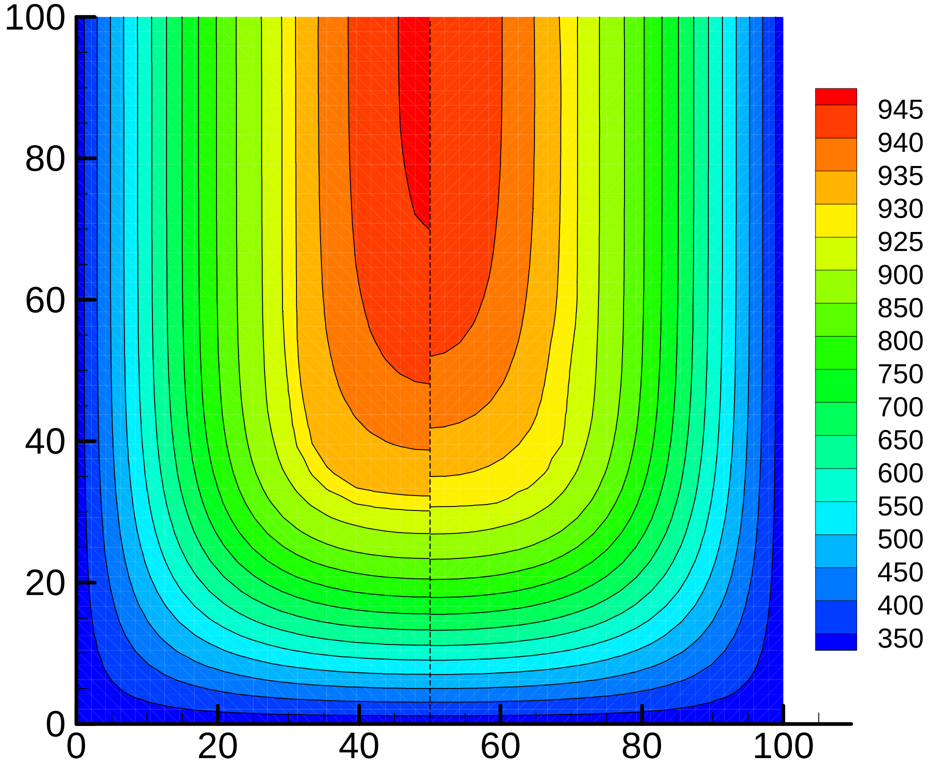


Figure 5.26: Temperature field at time 40s. Left: predicted by Lever rule, Right: predicted by the database approach

is used to greatly reduce the number of sample problems needed. Numerical results of the sample problems using the microscale model are stored using only three fields, solidification time, temperature at solidification time, and orientation angle at solidification time. The proposed multiscale framework for solidification is demonstrated in two cases with the same irregular domain but different boundary conditions. The time consuming microscale model is also solved to allow comparison with the computational results from the multiscale method. The predicted microstructure type (CET location), microstucture size, and volume fraction using the multiscale method compares well with the microscale model results.

This is the first study of using a database approach for solidification by quantitatively exploring similarities between sample problems and interested problems. The proposed

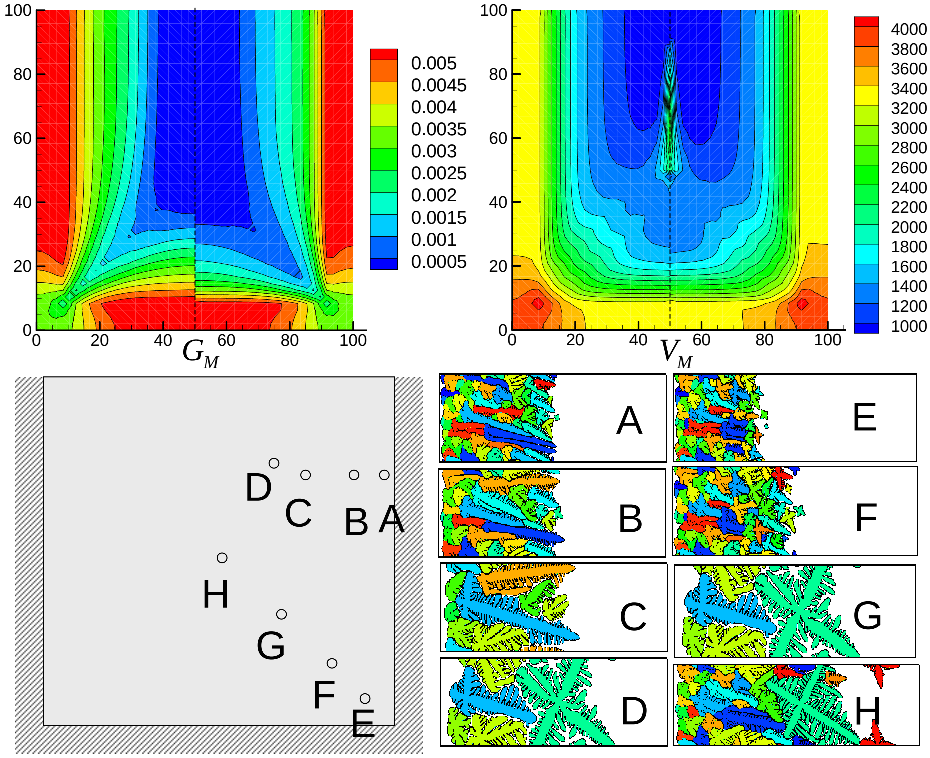


Figure 5.27: G_M and V_M fields (left half is result of the first iteration, right half is result after three iterations) and nearest microstructure in database at location A (95mm,75mm), B (90mm,75mm), C (75mm,75mm), D (60mm,80mm), E (90mm,10mm), F (80mm,20mm), G (65mm,35mm), H (50mm,50mm).

framework with a computationally efficient model to select relevant sample problems is novel and potentially applicable to problems other than solidification. Current efforts are on applying this framework to three-dimensional solidification and extending it to solidification systems with convection.

Chapter 6

Conclusions and suggestions for future research

In this thesis work, a novel model is introduced and implemented to study alloy solidification in the micro-scale. The potential of this method is demonstrated in a number of investigations. It is shown to be able to achieve converged solution with much coarser grid and much less computation effort than other numerical methods. By using adaptive meshing, domain decomposition and parallel computing, the micro-scale model is applied to study interaction between a large number of solidifying crystals. Based on the well verified micro-scale model, a database approach for multi-scale solidification is implemented by quantitatively exploring similarities between sample problems and problems of interest. A novel framework is introduced that involves a computationally efficient model to select relevant sample problems. The accuracy of the multi-scale model is demonstrated by comparing with the results obtained from a fully-resolved micro-scale model. A number of aspects of the introduced model remain to be further improved.

6.1 Multi-scale framework with convection effects

In this presented multi-scale solidification model, convection effects were not considered. The computationally efficient model that we used to identify relevant sample problems (with its analytical solution) is not applicable for problems that include convection effects. So extension of the current technique or other alternative techniques are necessary to efficiently consider convection effects in a multi-scale solidification framework.

For heat diffusion based models, the solidification process in the macro-scale can be characterized by only two parameters V_M and G_M^l, which are the solution features of a computationally efficient model. These two parameters provide a reduced representation of the solidification process. If convection effects needs to be considered, information about the flow, e.g. velocity magnitude and direction at a certain distance away from the solid-liquid interface (on the solid-liquid interface, the flow velocity is zero due to the none-slip condition). After a set of features are selected to represent the solidification process in the macro-scale, a relevant sample problem needs to be selected so that the same features can be obtained for this sample problem. In the current multi-scale framework, an analytical solution of the sample problem is utilized to find relevant sample problems. If more features (to incorporate convection effects) are used to describe the solidification process, it is probably very difficult to find relevant sample problems. Here, the relevant sample problem refers to the sample problem which leads to exactly the same features (reduced presentation of the solidification process). Interpolation or other algorithms e.g. locally weighted regression and neural network may be required to utilize the micro-scale solution of sample problems that are only approximately relevant to our problem of interest.

6.2 Consider fluid-structure interaction in micro-scale

After a crystal nucleates, the crystal would translate, rotate, come in contact with other crystals and/or break into fragments. To accurately incorporate the convection effects, the interaction between a crystal and the liquid alloy (fluid-structure interaction) must be considered in the micro-scale model. Recently, there are signifiant advances in the area of fluid-structure interaction. These advances can be used to improve the micro-scale model. However, the solidification problem here is more complicated than many of the other fluid structure interaction applications reported in the literature. On one hand, we need to preserve the crystal shape exactly when considering the movement of crystals as a basic requirement for capturing fluid structure interaction. At the same time, the crystal grows due to solidification. Meshing techniques and level set method would require significant improvements to incorporate these new abilities. Notice that, for fluid structure interaction in the solidification process, the deformation in the crystal due to flow is usually not significant. So it may be accurate enough to model the movement of crystals as rigid body translation and rotation in many cases. In other cases, the contact/impact between crystals and flow may leads to fragmentation of the dendrite arms. For efficiency in computation, we may need to use different models to model the fluid-structure interaction in solidification and smartly select the appropriate model for different crystals, for example according to the relative motion of the crystal.

6.3 Atomic scale computation

Our current micro-scale model relies on inputs from the alloy phase-diagram and a number of parameters to mimic the crystal orientation anisotropy, surface tension effects, kinetic under-cooling effects and nucleation. Computation in the atomic scale and related multi-scale techniques to use atomic scale computation results are of great significance. Only with the inputs from the atomic-scale computation, the simulations in the continuum (micro-scale and macro-scale) can be accurate and of practical use. In the literature, there are mainly two types of methods to compute phase diagrams. One is based upon experimental trial-and-error methods to reduce the need for new costly experiments, such as Thermo-Calc and Pandat. Another method is completely based on first principle calculations e.g. the ATAT (Alloy Theoretic Automated Toolkit) toolkit. Phase diagrams only provide us the equilibrium phases. When and how a new phase will be nucleated must be captured using molecular dynamics computation. In our current growth model, we assume Gibbs-Thomson relation of the solid-liquid interface. The Gibbs-Thomson relation is only a phenomenological model. Molecular dynamics simulations could help us to build an accurate growth model.

6.4 Solve inverse problem to identify nucleation model

Nucleation in alloy solidification is dominated by heterogenous nucleation, which highly depends on the number and type of inclusions in the liquid alloy. In practice, , the nucleation behavior largely depends on how the liquid alloy is prepared, which might be very different in different factories. Nucleation process is so complicated that it is rather difficult to accurately capture all aspects of the nucleation process for practical alloys in real applications completely by computation. So it will be of great significance to identify parameters in the nucleation model from experimental studies. In practice, the factory usually prepares the molten alloy following the same procedure. If we can obtain microstructure information (e.g. microstructure images) with certain processing parameters, and solve an inverse problem of solidification to identify a best fit of the nucleation model, we will be able to apply the nucleation model to predict microstructure in other solidification processes using exactly the same alloy.

6.5 Solid-solid phase transformation

In our current model, only liquid to solid phase transformation is considered. Modeling solid-solid phase transformation after solidification and study of the mechanical properties of the final microstructure is an open area of research.

BIBLIOGRAPHY

[1] R. Almgren, Second-order phase field asymptotics for unequal conductivities, SIAM J. Appl. Math. 59 (1999) 2086-2107.

[2] N. Al-Rawahi, G. Tryggvason, Numerical simulation of dendritic solidification with convection: two-dimension geometry, J. Comput. Phys. 180 (2002) 471-496.

[3] N. Al-Rawahi, G. Tryggvason, Numerical simulation of dendritic solidification with convection: three-dimension flow, J. Comput. Phys. 194 (2004) 677-696.

[4] M. Apel, B. Boettger, H.J. Diepers, I. Steinbach, 2D and 3D phase field simulations of lamella and fibrous eutectic growth, J. Crys. Growth 237-239 (2002) 154–158.

[5] A. Badillo, C. Beckermann, Phase-field simulation of the columnar-to-equiaxed transition in alloy solidification, Acta Materialia 54 (2006) 2015-2026.

[6] C. Beckermann, H.-J. Diepers, I. Steinbach, A. Karma, X. Tong, Modeling melt convection in phase-field simulations of solidification, J. Comput. Phys. 154 (1999) 468–496.

[7] C. Beckermann, R. Viskanta, Mathematical modeling of transport phenomena during alloy solidification, Appl. Mech. Rev. 46 (1993) 321–336.

[8] W. J. Boettinger, J. A. Warren, C. Beckermann, A. Karma, Phase-field simulation of solidification, Annu. Rev. Mater. Res. 32 (2002) 163–194.

[9] W.J. Boettinger, J.A. Warren, Simulation of the cell to plane front transition during directional solidifiation at high velocity, J. Crys. Growth 200 (1999) 583–591.

[10] G. Caginalp, Dynamical renormalization group calculation of a two-phase sharp interface model, Physical Review E 60 (1999) 6267–6270.

[11] D.L. Ceynar, C. Beckermann, Measurement of the density of succinonitrile-acetone alloys, J. Crys. Growth 222 (2001) 380-391.

[12] S. Chen, B. Merriman, S. Osher, P. Smereka, A simple level set method for solving Stefan problems J. Comput. Phys. 135 (1997) 8–29.

[13] J. Chessa, P. Smolinski, T. Belytschko, The extended finite element method (XFEM) for solidification problems, Int. J. Num. Meth. Eng. 53 (2002) 1959–1977.

[14] H.J. Diepers, C. Beckermann, I. Steinbach, Simulation of convection and ripening in a binary alloy mush using the phase field method, Acta Mater. 47 (1999) 3663–3678.

[15] W. E, B. Engquist, The Heterogeneous Multi-scale Methods, Comm. Math. Sci. 1 (2003) 87–133.

[16] C. Eck, P. Knabner, S. Korotov, A two-scale method for the computation of solid-liquid phase transitions with dendritic microstructure, J. Comput. Phys. 178 (2002) 58-80.

[17] R. Fedkiw, T. Aslam, B. Merriman, S. Osher, A non-oscillatory Eulerian approach to interfaces in multimaterial flows (The Ghost Fluid Method), J. Comput. Phys. 152 (1999) 457–492.

[18] S.D. Felicelli, D.R. Poirier, J.C. Heinrich, Macrosegregation patterns in multicomponent Ni-base alloys, J. Crys. Growth 177 (1997) 145–161.

[19] W.M. Feng, P. Yu, S.Y. Hu, Z.K. Liu, Q. Du, L.Q. Chen, Spectral implementation of an adaptive moving mesh method for phase-field equations, J. Comput. Phys. 220 (2006) 498-510.

[20] C.A. Gandin and M. Rappaz, A 3D Cellular Automaton algorithm for the prediction of dendritic grain growth, Acta Materialia 45 (1997) 2187–2195.

[21] W.L. George, J.A. Warren, A parallel 3D Dendritic growth simulator using the phase field method, J. Comput. Phys. 177 (2002) 264–283.

[22] F. Gibou, R. Fedkiw, R. Caflisch, S. Osher's, A level set approach for the numerical simulation of dendritic growth, Journal of Scientific Computing, Vol.19 (2003) 183–199.

[23] U. Hecht, L. Granasy, T. Pusztai, B. Bottger, M. Apel, V. Witusiewicz, L. Ratke, J. De Wilde, L. Froyen, D. Camel, B. Drevet, G. Faivre, S.G. Fries, B. Legendre, S. Rex, Multiphase solidification in multicomponent alloys, Mater. Sci. and Eng. R 46 (2004) 1–49.

[24] J.D. Hunt, Steady state columar and equiaxed growth of dendrites and eutectic, Mater. Sci. Eng. 65 (1984) 75–83.

[25] F.P. Incropera, W.D. Bennon, A continuum model for momentum, heat and species transport in binary solid-liquid phase change systems - I. model formulation, Int. J. Heat Mass Transfer 30 (1987) 2161–2170.

[26] Y. Jaluria, K. E. Torrance, Computational Heat Transfer, second edition, published by Taylor & Francis, 2003, pp. 49-54.

[27] D. Jasnow, J. Vinals, Dynamical scaling during interfacial growth following a morphological instability, Physical Review A 40 (1989) 3864–3870.

[28] J. Jeong, N. Goldenfeld, J. Dantzig, Phase field model for three-dimensional dendritic growth with fluid flow, Phy. Rev. E 64 (2001) 0416021–0416024.

[29] D. Juric, G. Tryggvason, A front-tracking method for dendritic solidification, J. Comput. Phys. 123 (1996) 127–148.

[30] A. Karma, W.J. Rappel, Quantitative phase-field modelling of dendritic growth in two and three dimensions, Phys. Rev. E 57 (1998) 4323–4349.

[31] A. Karma, W.J. Rappel, Phase field model for computationally efficient modelling of solidification with arbitrary interface kinetics, Phys. Rev. E 53 (1996) 3017–3020.

[32] Y.T. Kim, N. Goldenfeld, J. Dantzig, Computation of dendritic microstructures using a level set method, Physical Review E 62 (2000) 2471–2474.

[33] K. Kremeyer, Cellular Automata Investigations of Binary Solidification, J. Comput. Phys. 142 (1998) 243–262.

[34] W. Kurz, D.J. Fisher, Fundamentals of solidification, third edition, Trans Tech Publications Ltd, Switzerland, 1989.

[35] M. Lai, C. Peskin, An immersed boundary method with formal second-order accuracy and reduced numerical viscosity, J. Comput. Phys. 160 (2000) 705–719.

[36] Y. Lee, R. Ananth, W. Gill, Selection of a length scale in unconstrained dendritic growth with convection in the melt Journal of Crystal Growth, Journal of Crystal Growth 132 (1993) 226–230.

[37] P.D. Lee, A. Chirazi, R.C. Atwood, W. Wang, Multiscale modelling of solidification microstructures, including microsegregation and microporosity, in an Al-Si-Cu alloy, Materials Science and Engineering A 365 (2004) 57–65.

[38] Y. Lu, C. Beckermann, J.C. Ramirez, Three-dimensional phase field simulations of the effect of convection on free dendritic growth, J. Crystal Growth 280 (2005) 320–334.

[39] B. Merriman, J. Bence and S. Osher, Motion of multiple junctions: a level set approach, J. Comp. Phys. 112 (1994) 334–363.

[40] W.W. Mullins, R.F. Sekerka, Stability of a planar interface during solidification of a dilute binary alloy, J. Appl. Phys. 35 (1964) 444–451.

[41] B. Nestler, D. Danilov, P. Galenko, Crystal growth of pure substances: Phase-field simulations in comparison with analytical and experimental results, J. Comput. Phys. 207 (2005) 221–239.

[42] J. Ni, C. Beckermann, A volume-averaged two-phase model for transport phenomena during solidification, Metall. Trans. B 22 (1990) 349–361.

[43] S. Osher, R. Fedkiw, Level set Methods and Dynamic Implicit Surfaces, Springer-Verlag, New York, Inc. 2003.

[44] S. Osher, R. P. Fedkiw, Level Set Methods: An Overview and Some Recent Results, J. Comput. Phys. 169 (2001) 463–502.

[45] N. Palle, J. Dantzig, An adaptive mesh refinement scheme for Mullins Sekerka theory [solidification problems], Metal. Mater.Trans. A 27 (1996) 707–717.

[46] M. Plapp, A. Karma, Multiscale random-walk algorithm for simulating interfacial pattern formation, Physical Review Let. 84 (2000) 17401-17443.

[47] D.R. Poirier, P. Nandapurkar, S. Ganesan, The energy and solute conservation equations for dendritic solidification, Metall. Trans. B 22 (1990) 889–900.

[48] N. Provatas, N. Goldenfeld, J. Dantzig, Efficient computation of dendritic microstructures using adaptive mesh refinement, Phys. Rev. Letters 80 (1998) 3308–3311.

[49] N. Provatas, N. Goldenfeld and J. A. Dantzig, Adaptive mesh refinement compu-

tation of solidification microstructures using dynamic data structures, J. Comput. Phys. 148 (1999) 265–290.

[50] H. Rafii-Tabar, A. Chirazi, Multiscale computational modelling of solidification phenomena, Physics Reports 365 (2002) 145–249.

[51] M. Rappaz and C.A. Gandin, Probabilistic modelling of microstructure formation in solidification processes, Acta Metall. 41 (1993) 345-360.

[52] K.A. Rathjen, L.M. Jiji, Heat conduction with melting or freezing in a corner, J. Heat Transfer ASME 93 (1971) 101–109.

[53] J.C. Ramirez, C. Beckermann, Examination of binary alloy free dendritic growth theories with a phase field model, Acta Mater. 53 (2005) 1721–1736.

[54] D. Samanta, N. Zabaras, Modeling melt convection in solidification processes with stabilized finite element techniques, Int J. Numer. Meth. Eng. 64 (2005) 1769-1799.

[55] D. Samanta, N. Zabaras, Macrosegregation in the solidification of Aluminum Alloys on uneven surfaces, Int. J. Heat Mass Transfer 48 (2005) 4541-4556.

[56] J.A. Sethian, Level set Methods (Evolving Interfaces in Geometry, Fluid Mechanics, Computer Vision, and Material Science, published by the Press Syndicate of the University of Cambridge, 1996.

[57] W. Shyy, H.S. Udaykumar, M.M. Rao, R.W. Smith, Computational Fluid Dynamics with Moving Boundaries, 1st edn., Tailor & Francis, Washington, D.C., 1996.

[58] A. Spittle and S. Brown, Computer simulation of the effects of alloy variables on the grain structures of castings, Acta Metall. 37 (1989) 1803–1810.

[59] Y. Sun and C. Beckermann, Diffuse interface modeling of two-phase flow based on averaging: mass and momentum equations, Physica D 198 (2004) 281–308.

[60] T. Takaki, T. Fukuoka, Y. Tomita, Phase field simulation during directional solidification of a binary alloy using adaptive finite element method, J. Crys. Growth 283 (2005) 263–278.

[61] L. Tan, N. Zabaras, A level set simulation of dendritic solidification with com-

bined features of front-tracking and fixed-domain methods, J. Comput. Phys. 211 (2006) 36–63.

[62] L. Tan and N. Zabaras, A level set simulation of dendritic solidification of multicomponent alloys, Journal of Computational Physics, 221, (2007) 9–40.

[63] L. Tan, N. Zabaras, Modeling the interaction of multiple dendrites in solidification using a level set method, J. Comput. Phys., in press.

[64] T.E. Tezduyar, Stabilized finite element formulations for incompressible flow computations, Adv. Appl. Mech. 28 (1992) 1–43.

[65] J. Tiaden, Phase field simulations of the peritectic solidifiation of Fe-C, J. Crys. Growth 198/199 (1999) 1275–1280.

[66] R. Tonhardt, G. Amberg, Phase-field simulation of dendritic growth in a shear flow, Journal of Crystal Growth, 194 (1998) 406–425.

[67] S.L. Wang, R. F. Sekerka, A.A. Wheeler, B. T. Murray, S. R. Coriell, Thermodynamically consistent phase field models for solidification, Physica D 69 (1993) 189–200.

[68] W. Wang, P. D. Lee and M. McLean, A model of solidification microstructures in nickel-based superalloys: predicting primary dendrite spacing selection, Acta Materialia, 51 (2003) 2971–2987.

[69] C.Y. Wang, C. Beckermann, Equiaxed dendritic solidification with convection: part I. Multiscale / Multiphase modeling, Metallurgical and materials transactions A 27 (1995) 2754–2764.

[70] J.A. Warren, W.L. George, A parallel 3D dendritic growth simulator using the phase-field method, J. Comput. Phys. 177 (2002) 264–283.

[71] A. Wheeler, W. Boettinger, G. Mcfadden, Phase-field model for isothermal phase transitions in binary alloys, Physical Review A 45 (1992) 7424-7440.

[72] N. Zabaras, D. Samanta, A stabilized volume-averaging finite element method for flow in porous media and binary alloy solidification processes, International J. for Numerical Methods in Engineering, 60 (2004) 1103–1138.

[73] N. Zabaras, B. Ganapathysubramanian and L. Tan, Modeling dendritic solidifica-

tion with melt convection using the extended finite element method (XFEM) and level set methods, J. Comput. Phys. 218 (2006) 200–227.

[74] P. Zhao, J. C. Heinrich, Front-tracking finite element method for dendritic solidification, J. Comput. Phys. 173 (2001) 765–796.

[75] P. Zhao, M. Venere, J. Heinrich, D. Poirier, Modeling dendritic growth of a binary alloy, J. Comput. Phys. 188 (2003) 434–461.

[76] P. Zhao, J.C. Heinrich, Numerical approximation of a thermally driven interface using finite elements, International Journal for Numerical Methods in Engineering 56 (2003) 1533–1547.

[77] H. Zhao, T. Chan, B. Merriman, S. Osher, A Variational level set method for multiphase motion, J. Comput. Phys. 127 (1996) 179–195.

[78] Tecplot: Plotting and Data Visualization Software - Tecplot 10, `http://www.tecplot.com/`.

[79] K. Schloegel, G. Karypis, and V. Kumar, ParMetis: Parallel static and dynamic multi-constraint graph partitioning, Concurrency and Computation: Practice and Experience 14 (3) (2002) 219–240.

[80] PETSc: http://www-unix.mcs.anl.gov/petsc/petsc-as/index.html