To the Graduate Council:

I am submitting herewith a thesis written by Chun Wai Chang entitled "Dendritic Growth and Ostwald Ripening in Solidification Using Phase Field Method." I have examined the final copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Physics.

Dr. Geoffrey Ganright, Major Professor

We have read this thesis and
recommend its acceptance:

Accepted for the council:

Associate Vice Chancellor and
Dean of the Graduate School

DENDRITIC GROWTH AND OSTWALD RIPENING IN SOLIDIFICATION USING PHASE FIELD MODEL

A Thesis

Presented for the

Master of Science

Degree

The University of Tennessee, Knoxville

Chun Wai Chang

May 1998

Dedicated to my family

Acknowledgment

I would like to thank all the members in my thesis advisory committee. Professor Alexiades in the Mathematics Department, Professor Canright, and Professor Quinn in the Physics Department. Dr. Canright serves as my major advisor and Dr. Alexiades is the one who guides me all the way. I have to thank them for reaching the agreement on the roles they play in the committee. I thank for Dr. Alexiades' valuable help and suggestions. I would also like to thank Dr. Quinn for serving in the committee.

I have learned a lot from the classes taught by the professors serving in the committee. The two statistical mechanics classes taught by Dr. Canright, especially the second one on phase transition and renormalization group, the series of two classes on solidification processes, and the class on mathematical modeling taught by Dr. Alexiades, build a strong background needed for my research. Dr. Quinn's solid state physics classes are also very helpful. I must admit that I gained much more understanding on solid state physics later on when I had the opportunity of grading for Dr. Quinn.

I would also like to thank other excellent professors in the Physics Department, including Dr. Harris and Dr. Breinig, from whom I have learned a lot in their classes. Most of my computations were done on the zeus machine (a Digital Alphastation 500/333 MHz) of the Mathematics Department. I would like to extend my gratitude to the people who keep the machine up and running.

This thesis work and my graduate study would not be possible without the financial support from the Physics Department, especially for the last semester.

Abstract

Dendritic growth and Ostwald ripening are observed in solidification process. The simulation of dendritic growth is very difficult using the classical approach which needs interface tracking. A new approach called phase field method could bypass such difficulty. Recently a formulation of phase field method with an entropy functional which accounts for the increase of local entropy was suggested. It has advantages over the formulation with a free energy functional. We evaluate the performance of this phase field method by examining the morphological patterns and kinetic variables in two dimensional simulations of a single dendrite growing in undercooled melt.

The simulations are done with nickel's material parameters. Isotropic and anisotropic cases are examined. For isotropic case, the solid grows branches in all directions without a dominant branch. For anisotropic case, the preferred directions is set in the x- and y-directions. With strength of anisotropy as small as 0.01, a main branch formed in the x-direction away from the boundary and some large branches in y-direction are formed. Side branches growing in diagonal directions are suppressed. When noise is added in the anisotropic cases, we see that side branches perpendicular to the main branch are formed. The tip velocity, tip temperature, and tip radius are computed. They, in dimensionless units, are $7, -0.12$, and 0.045 respectively, corresponding to $50m/s, 1702K (26K$ under the melt temperature), and $10^{-7}m$ in dimensional units.

Ostwald ripening occurs in a later stage of solidification and is also described by the same governing equations. Simulation is also done on Ostwald ripening. Initially we have two circles of solid immersed in liquid. As time evolves, the circle with

larger radius grows bigger at the expense of the smaller circle which eventually vanishes. The results we obtained show all the essential features of dendritic growth and Ostwald ripening observed in experiments. It shows that the formulation we examine is a promising theoretical and computational approach to the problem.

Table of Contents

List of Figures

Chapter 1. Introduction

Solidification occurs when a substance makes its transition from a disordered liquid state to an ordered solid state. It is a very complex process. Numerous factors determine the details of the process.

The way a substance solidifies determines many physical properties of the resulting solid. Therefore, it is an important area of science and engineering. While physicists are interested in its underlying thermodynamics and the morphological pattern of the solid formed during the process, applied mathematicians are interested in the mathematical techniques and analysis, and materials engineers are interested in the applications in manufacturing such as casting and welding.

Solidification usually begins when a liquid is cooled below its freezing point and a nucleation site is formed or introduced, such as a solid seed for crystal growth. The process continues when more and more undercooled liquid around the nucleation site transforms into solid. It can be seen as the propagation of the solid-liquid interface when the solid grows into the undercooled liquid.

When we look at the solid under a powerful microscope in the earlier stage of the solidification process, we see a complex tree-like solid. This tree-like structure is called a dendrite, derived from a Greek word which means tree. Such a description is appropriate because the structure usually has a main branch, some side branches growing from the main branch, smaller secondary side branches growing from the side branches, and so on. The side branches tend to grow into each other until there

is no more room for growth. On the whole, we see many dendritic crystals packed closely to each other.

The solidification process is important in engineering and technology for its applications. One good example is metallic materials used in our everyday life, from aluminum to steel, from soda cans to engines. These products are manufactured from the corresponding metals in liquid phase and by casting into the desired shapes. Another good example of an everyday substance that is also manufactured by solidification is plastics.

The dendrites formed during solidification affect the mechanical, electrical, and chemical properties of the solid. The shape, size, and speed of growth of dendrites are important factors affecting the final properties of the material. Thus, the different properties and responses of the material, such as yield strength, toughness, corrosion resistance, hot cracking, and etc., are related to the dendritic morphology. These properties are important aspects that engineers in the metal casting industry want to know about, because they are related to how strong the metal is, how quickly it will rust or wear out, how stretchable it is, and how easily two pieces of metal can be welded into each other. Even though the dendritic structure can be modified by heat treatment after the solidification is completed, the final properties of the material are still largely dependent on the original dendritic structure. Therefore, to obtain the desired physical properties of the material in question, it is very important to have certain understanding and control of the solidification process.

Experiments in dendritic solidification have been performed both in ground-based laboratories and in micro-gravity environment most recently, on the space shuttle. In ground-based experiments, both kinetic and morphological measurements for dendrites of various materials, especially succinonitrile (SCN), are well established. The Isothermal Dendritic Growth Experiment(IDGE), performed aboard the space shuttle Columbia in November 1997, is designed to investigate dendritic growth in micro-gravity environment. Again, kinetic and morphological measurements for SCN and pivalic acid are made in IDGE.

Scientists and engineers have gained much knowledge on dendritic growth over the years. However, under the influence of gravity on the Earth, there is a rather strong density gradient around the interface caused by the release of latent heat or the rejection of solute. This density gradient induces a convection which makes the dendritic growth process very complicated. Before we can gain knowledge on dendritic growth with such gravity induced convection, we need first to know what is happening during solidification without the influence of gravity. So far, it is almost impossible to predict what gravity will do. Therefore, recent development of theoretical formulations of the problem and subsequent numerical simulations are done under the assumption that the density of the substance is constant over the entire volume in question, including the solid, the interface, and the liquid. It is assumed that there is no convective flow of any kind, not only that under gravitational influence.

The reason the solid-liquid interface branches out and forms finger like structures is

as follows. Suppose there is a bump like solid growth on an otherwise flat interface. The thermal gradient (or concentration or other gradient) between the solid and the undercooled liquid near the bump is steeper than that around the flat interface. It is because the temperature difference is applied across a shorter length near the bump. In other words, the surface area of the bump is larger than that of a flat surface when the bump is absent . Therefore, the bump diffuses latent heat faster and it grows faster. However, if the bump becomes sharp enough, it has a tendency of melting back due to surface tension. We can see this in the light of Gibbs-Thomson relation which says that the equilibrium melting temperature is reduced by curvature.

Thus, in earlier stages of solidification, we see fingers growing into the undercooled liquid. However, in later stages the thermal gradient has greatly reduced and the surface tension effect dominates and Ostwald ripening occurs. Ostwald ripening or coarsening also plays an important role in solidification. Regions of low curvature are more favorable than regions of high curvature, which is being eliminated in coarsening. This again can be explained with the Gibbs-Thomson relation. Ostwald ripening could affect the mechanical, electrical, and chemical properties of the material, like those we described above for dendritic growth. Its impacts on the dendritic formation is is still not well known.

Theoretical modeling of dendritic growth has undergone a number of stages. Classically, the problem is viewed as a moving boundary problem. It is governed by a single differential equation with matching free boundary conditions. In computations, it is

necessary to keep track of the exact location of the moving, sharp interface. Since the dendrite has a complicated morphological structure, it is a formidable task to track the solid-liquid interface.

However, there is a recent development which introduces a phase field variable or order parameter ϕ. ϕ has a value of 0 when the material under investigation is in its bulk solid state and a value of 1 for bulk liquid. It takes on values within a continuous range from 0 to 1 for the interface. The interface is no longer viewed as a sharp surface, but rather as a thin layer with certain thickness. The early versions of the phase field method employed a free energy functional [21]. It is not satisfactory for it does not account for entropy creation during the process. An ad hoc way has to be included in the model by artificially adding a term proportional to the time derivative of the phase field variable to take care of the latent heat released. Even so, this method only applies in isothermal cases. To overcome such shortcoming, another phase field model with an entropy functional was developed [17]. It takes care of the entropy production, complying with irreversible thermodynamics, and it applies also to non-isothermal cases.

The contents of this thesis are as follows. In Chapter 2 we present the formulation of the new phase field model, following [12], [17], [22], which leads to a pair of coupled parabolic partial differential equations. In Chapter 3 we describe the discretization of the PDEs and the numerical methodology we will employ. The results of the numerical computations are presented in Chapter 4, and the conclusions at the end

of the chapter.

Chapter 2. Formulation of the Phase Field Model

2.A. Introduction

During solidification a material undergoes a first order phase transition. The amount of latent heat released at the interface when solid is formed has to be accounted for. The classical theoretical approach of solving the problem is through a single differential equation for the temperature field with the evolving solid-liquid interface as a moving boundary. At the interface, boundary conditions for the temperature and its derivative have to be satisfied. Therefore the location of the interface has to be found all the time during its evolution in order to solve the problem. As the morphological pattern of the interface is very complicated, it is a formidable task to keep track of it.

In the early 1980's, a theoretical approach to the problem, called phase field model, using a free energy functional was suggested, [4],[5],[6],[21]. It introduces a phase field variable or order parameter ϕ to represent the solid, liquid, and the interface, for example, the region with $\phi = 0$ is the bulk solid, $\phi = 1$ is the bulk liquid, and $0 < \phi < 1$ is the interface. The interface now is a thin layer with certain thickness, instead of being a sharp surface . ϕ is governed by a partial differential equation, and in the sharp interface limit, it satisfies the interface conditions. The model also modifies the heat transport equation by adding extra terms with ϕ dependence, to account for latent heat release. The PDEs apply to the entire region under

consideration and take care of the boundary conditions at the interface. Thus, we now have two coupled partial differential equations to solve instead of only one. Nevertheless, it still has great advantages over the classical approach, because it is not necessary to explicitly keep track of the interface.

However, there are some difficulties with the phase field method using a free energy functional. It is applicable only to isothermal cases. The basic formulation does not take into account the liberation of latent heat at the interface when solid is formed, and to overcome this, a term containing $\dot{\phi}$ is artificially inserted to account for the latent heat released. Even so, the entropy production during this irreversible process is not taken care of .

Despite such shortcomings, the phase field model based on a free energy functional provided a direction and a framework for further investigation and improvement, and it led eventually to an improved phase field model based on an entropy functional [14].

In the new formulation [12],[17],[22], there are still two coupled partial differential equations for the temperature and phase field, but they are now derived from a single entropy functional which guarantees local positive entropy production. It is consistent with the second law of thermodynamics. It is applicable not only to isothermal cases but also to non-isothermal ones. It again satisfies that the phase field variable takes on fixed values in bulk solid and liquid. We take $\phi = 0$ for solid, $\phi = 1$ for liquid, and $0 < \phi < 1$ for the interfacial layer, as above. In

this model, we can choose the parameters and functions entering it so as to agree with known thermophysical material parameters, such as heat capacity and surface tension. Certain relations between model parameters and physical properties are fixed by examining some asymptotic cases in the sharp interface limit.

2.B. Entropy Functional

We now describe the theoretical formulation using an entropy functional, following [12],[17],[22]. It is derived for a pure material in a closed system of volume Ω. The order parameter $\phi(x,t)$ takes on the values as described above. We assume that throughout the system the density is constant; this implies that there is no convection in the liquid.

Conservation of energy (first law of thermodynamics) requires

$$\frac{d}{dt}E + \int_{\partial V} \vec{q} \cdot \vec{n}\, dA = 0, \tag{1}$$

where E is the internal energy of an arbitrary subvolume V of Ω, ∂V is the surface of V with outward normal $\vec{n}$, and $\vec{q}$ is the flux of internal energy (crossing a unit area in unit time).

Applying Gauss theorem and equating the integrands, we have

$$\dot{e} + \bigtriangledown \cdot \vec{q} = 0, \tag{2}$$

expressing local conservation of energy, where $\epsilon = e(x,t)$ is the energy density at

location x at time t, such that

$$E = \int_V e\, dV.$$

The entropy of any sub-volume V is assumed to be

$$S = \int_V [s - \frac{1}{2}\epsilon^2 |\bigtriangledown \phi|^2]\, dV, \tag{3}$$

where $s = s(e, \phi)$ is an entropy density (for bulk phases), and ϵ is a constant, the meaning of which will be discussed later. The second term in the above equation is a gradient entropy term analogous to Landau-Ginzburg or Cahn-Hilliard gradient energy term, accounting for surface entropy.

Differentiating (3), we obtain

$$\dot{S} = \int_V \{(\frac{\partial s}{\partial e})\dot{e} + [(\frac{\partial s}{\partial \phi}) + \epsilon^2 \bigtriangledown^2 \phi]\dot{\phi} - \epsilon^2 \bigtriangledown \cdot(\dot{\phi} \bigtriangledown \phi)\}\, dV,$$

$$\dot{S} = \int_V \{\vec{q} \cdot \bigtriangledown(\frac{\partial s}{\partial e}) + [(\frac{\partial s}{\partial \phi}) + \epsilon^2 \bigtriangledown^2 \phi]\dot{\phi}\}\, dV - \int_{\partial V} [(\frac{\partial s}{\partial e})\vec{q} + \epsilon^2 \dot{\phi} \bigtriangledown \phi] \cdot \vec{n}\, dA. \tag{4}$$

According to the second law of thermodynamics, the entropy production for irreversible process is greater than 0 in any sub-volume V. This local entropy production is, subtracting the entropy flux through the surface ∂V from $\dot{S}$,

$$\dot{S} + \int_{\partial V} (\frac{\vec{q}}{T} \cdot \vec{n} + \epsilon^2 \dot{\phi} \bigtriangledown \cdot \vec{n})\, dA \geq 0, \tag{5}$$

where T is the absolute temperature, $\frac{\vec{q}}{T}$ is the entropy flux due to heat flow, and $\epsilon^2 \dot{\phi} \bigtriangledown \phi$ is an entropy flux related to changes in phase field at the boundary of V.

Substituting (4) into (5), we get

$$\int_V \{q \cdot \bigtriangledown(\frac{1}{T}) + [(\frac{\partial s}{\partial \phi})_\epsilon + \epsilon^2 \bigtriangledown^2 \phi]\dot{\phi}\}\, dV \geq 0. \tag{6}$$

where we have used

$$de = Tds + (\frac{\partial e}{\partial \phi})_s \, d\phi \tag{7}$$

to identify $(\frac{\partial s}{\partial e})_\phi = \frac{1}{T}$.

So, we can guarantee positive local entropy production by choosing

$$\vec{q} = M_T \bigtriangledown (\frac{1}{T}), \tag{8}$$

$$\tau \dot{\phi} = (\frac{\partial s}{\partial \phi})_e + \epsilon^2 \bigtriangledown^2 \phi, \tag{9}$$

where M_T and τ are positive.

Thus, equations (8) and (9) force the integrand in (6) to be a sum of two squares,

$$\frac{q^2}{M_T} + \tau \dot{\phi}^2,$$

and the local entropy production is thus positive.

So, the governing equations we have obtained are

$$\dot{e} = - \bigtriangledown \cdot [M_T \bigtriangledown (\frac{1}{T})], \tag{10}$$

$$\tau \dot{\phi} = -\frac{1}{T} (\frac{\partial e}{\partial \phi})_s + \epsilon^2 \bigtriangledown^2 \phi. \tag{11}$$

2.C. Helmholtz Free Energy

To identify the derivative in equation (11), we use the Helmholtz free energy density

$$f = e - Ts.$$

Combining the differential of the above equation and equation (7), we obtain

$$(\frac{\partial e}{\partial \phi})_s = (\frac{\partial f}{\partial \phi})_T, \tag{12}$$

$$(\frac{\partial}{\partial T}\frac{f}{T})_\phi = -\frac{e}{T^2}. \tag{13}$$

Integrating (13), keeping ϕ constant, we have

$$f(T,\phi) = T[-\int_{T_m}^{T} \frac{e(\zeta,\phi)}{\zeta^2}\, d\zeta + G(\phi)], \tag{14}$$

where $G(\phi)$ at this moment is an undetermined function of ϕ, and T_m is the melting temperature.

Assume the internal energy density has the form

$$e = e_S(T) + p(\phi)L(T) = e_L(T) + [p(\phi) - 1]L(T),$$

where $e_S(T)$ and $e_L(T)$ are the classical internal energy density of the solid and liquid phases,

$$L(T) = e_L(T) - e_S(T),$$

specifically $L_0 \equiv L(T_m)$ is the latent heat at the melt temperature, and $p(\phi)$ satisfies the restrictions $p(0) = 0$, and $p(1) = 1$. The above expression for internal energy guarantees that the correct amount of latent heat is released when solid forms at the interface. From (14), the free energy density corresponding to this internal energy is

$$f(T,\phi) = T - [\int_{T_m}^{T} \frac{e_L(\zeta)}{\zeta^2}\, d\zeta - [p(\phi) - 1]Q(T) + G(\phi)], \tag{15}$$

where

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

Note that $Q(T)$ is monotonically increasing with respect to T , and $Q(T_m) = 0$.

From (12) and (15), we get

$$(\frac{\partial e}{\partial \phi})_s = -TQ(T)p'(\phi) + TG'(\phi).$$

The governing equations thus become

$$\dot{e}_L(T) + \dot{p}(\phi)L(T) + [p(\phi) - 1]\dot{L}(T) = - \bigtriangledown \cdot [M_T \bigtriangledown (\frac{1}{T})], \tag{16}$$

$$\tau\dot{\phi} = Q(T)p'(\phi) - G'(\phi) + \epsilon^2 \bigtriangledown^2 \phi. \tag{17}$$

Since we have set $\phi = 0$ for the bulk solid and $\phi = 1$ for the bulk liquid, independent of the temperature T, f must have local minima with respect to ϕ at $\phi = 0$ and $\phi = 1$ and be continuous with respect to temperature. That is,

$$f(T_m, 0) = f(T_m, 1),$$

$$\frac{\partial f}{\partial \phi}|_{\phi=0,1} = 0,$$

$$\frac{\partial^2 f}{\partial \phi^2}|_{\phi=0,1} > 0.$$

The above requirements for f in turn impose the following restrictions on G.

$$G(0) = G(1),$$

$$[G'(0) - p'(\phi)Q(T)]|_{\phi=0,1} = 0, \tag{18}$$

$$[G''(\phi) - p''(\phi)Q(T)]|_{\phi=0,1} > 0. \tag{19}$$

To satisfy the above restrictions, $G(\phi)$ is chosen as a symmetric double well potential with minima at 0 and 1,

$$G(\phi) = \frac{1}{4a} g(\phi) = \frac{1}{4a} \phi^2 (1-\phi)^2, \tag{20}$$

where a is a positive number. Several other choices have been examined [19], and (20) seems to be the most convenient.

For simplicity, we assume that the energy density in the liquid is a linear function of temperature,

$$e_L(T) = e_L(T_m) + c \cdot (T - T_m),$$

where c is the heat capacity per unit volume, assumed to be constant here; this will lead to the classical heat equation in bulk phases.

The governing equations (16), (17) then become

$$\{c + [p(\phi) - 1]L'(T)\} \frac{\partial T}{\partial t} + L(T) p'(\phi) \frac{\partial \phi}{\partial t} = k \bigtriangledown^2 T, \tag{21}$$

$$\tau \frac{\partial \phi}{\partial t} = Q(T) p'(\phi) - \frac{1}{4a} g'(\phi) + \epsilon^2 \bigtriangledown^2 \phi, \tag{22}$$

where we have set $M_T = kT^2$, with k the thermal conductivity, which is also assumed to be constant.

2.D. Identifying the Parameters

We can relate a and ϵ to the interface thickness and surface energy by looking at a

1-D solution of equation (22) in the isothermal case $T = T_m$,

$$\epsilon^2 \frac{d^2\phi}{dx^2} - \frac{1}{4a} g'(\phi) = 0,$$

with boundary conditions $\phi \to 0$ as $x \to -\infty$ and $\phi \to 1$ as $x \to +\infty$. The solution is

$$\phi(x) = \frac{1}{2}[tanh \frac{x}{2\sqrt{2a}\epsilon} + 1].$$

From this solution, the interface has characteristic thickness

$$\delta = \sqrt{a}\epsilon.$$

Now the Helmholtz free energy functional at T_m, $F = E - T_m S$, is

$$F = \int_V [f(T_m, \phi) + \frac{1}{2}\epsilon^2 T_m (\nabla\phi)^2]\, dV.$$

The corresponding surface energy per area (a surface excess quantity because $f(T_m, \phi) = 0$ in the bulk) is

$$\sigma = \int_{-\infty}^{\infty} \epsilon^2 T_m \left(\frac{d\phi}{dx}\right)^2 dx,$$

$$\sigma = \frac{\sqrt{2}\,\delta T_m}{12a}.$$

To proceed, we undimensionalize the governing equations with the introduction of the following scales. The length scale w represents the size of system. The reference time scale is represented by the thermal diffusion time $\frac{w^2}{\lambda}$, where $\lambda = \frac{k}{c}$ is thermal diffusivity of the liquid, which was assumed to be constant above. Therefore, we write,

$$\hat{x} = \frac{x}{w},$$

$$\tilde{t} = \frac{t}{w^2/\kappa},$$
$$u = \frac{T - T_m}{T_m - T_0}.$$

T_0 is a reference temperature, which signifies the initial temperature or the magnitude of undercooling for Neumann boundary conditions and the external boundary temperature for Dirichlet conditions.

The governing equations become

$$(1 + [p(\phi) - 1]\frac{\tilde{L}'(u)}{\tilde{S}})\frac{\partial u}{\partial \tilde{t}} + \frac{\tilde{L}(u)}{\tilde{S}}p'(\phi)\frac{\partial \phi}{\partial \tilde{t}} = \nabla^2 u, \tag{23}$$

$$\frac{\tilde{\epsilon}^2}{m}\frac{\partial \phi}{\partial \tilde{t}} = \tilde{\epsilon}^2\nabla^2\phi + \tilde{\epsilon}\alpha\tilde{S}\tilde{Q}(u)p'(\phi) - \frac{1}{4}g'(\phi), \tag{24}$$

where

$$\tilde{L}(u) = \frac{L}{L_0},$$
$$\tilde{Q}(u) = \int_0^u \frac{\tilde{L}(\zeta)}{\{1 + [\frac{(T_m - T_0)}{T_m}]\zeta\}^2}\, d\zeta,$$
$$\alpha = \frac{\sqrt{2}wL_0^2}{12c\sigma T_m}, \tag{25}$$
$$\tilde{S} = \frac{c(T_m - T_0)}{L_0}, \tag{26}$$
$$\tilde{\epsilon} = \frac{\delta}{w} = \frac{\epsilon\sqrt{a}}{w}, \tag{27}$$

and

$$m = \frac{6\sqrt{2}\delta\sigma}{\tau\lambda T_m}. \tag{28}$$

We have to specify $p(\phi)$ in order to proceed further. We choose,

$$p(\phi) = \frac{\int_0^\phi g(\zeta)\, d\zeta}{\int_0^1 g(\zeta)\, d\zeta} = \phi^3(10 - 15\phi + 6\phi^2), \tag{29}$$

satisfying $p(0) = 0, p(1) = 1$, equation (18), and (19) independently of $Q(T)$ because $p'(\phi) = p''(\phi) = 0$ at $\phi = 0, 1$. That is, there is no restriction on $Q(T)$. For simplicity, we choose heat capacity per volume of solid = heat capacity per volume of liquid = c = constant. Also, we assume that $|T_m - T_0| << T_m$, so we can make the approximation

$$\tilde{Q}(u) = u,$$

and assume constant latent heat $L \equiv L_0$, so $\tilde{L} \equiv 1$. The governing equations (23) and (24) therefore become

$$\frac{\partial u}{\partial \tilde{t}} + \frac{30 g(\phi)}{\tilde{S}} \frac{\partial \phi}{\partial \tilde{t}} = \tilde{\nabla}^2 u, \tag{30}$$

$$\frac{\tilde{\epsilon}^2}{m} \frac{\partial \phi}{\partial \tilde{t}} = \tilde{\epsilon}^2 \tilde{\nabla}^2 \phi + 30 g(\phi) \tilde{\epsilon} \alpha \tilde{S} u - \frac{1}{4} g'(\phi). \tag{31}$$

The four dimensionless parameters $\alpha, \tilde{S}, \tilde{\epsilon}, m$ of the model are defined in (25)-(28) in terms of the physical properties $L_0, T_m, c, \alpha, \sigma$ of the material, the initial temperature T_0 , the phase field parameters a, τ, and the length scales $\delta = \epsilon a$ and w. Moreover, we can fix m by asymptotic analysis of the phase field equations in the limit $\tilde{\epsilon} \to 0$. We state the result as follows.

$$\frac{\partial u}{\partial \tilde{t}} = \nabla^2 u,$$

with interfacial conditions

$$u = -\tilde{\Gamma}(\tilde{\kappa} + \frac{\tilde{v}_l}{m}), \tag{32}$$

where

$$\tilde{\Gamma} = \frac{\sigma T_m}{w L_0 (T_m - T_0)}$$

is a dimensionless capillary length, and

$$[\frac{\partial u}{\partial \vec{n}}]_{sol}^{liq} = -\frac{\tilde{v}_l}{\tilde{S}}, \tag{33}$$

with $\tilde{v}_l$ dimensionless normal velocity of interface into liquid , and $\tilde{\kappa}$ dimensionless interfacial curvature (positive for convex projection into liquid).

When $\tilde{v}_l = 0$, equation (31) reduces to Gibbs-Thomson equation. The extra term in (31) is due to interface kinetic effect .

The parameter m represents the dimensionless interface kinetics coefficient. Comparing the undimensionalized equation (31) with its dimensional form

$$T = T_m - \frac{\sigma T_m}{L_0}\bar{\kappa} - \frac{v_l}{\mu},$$

gives a relation between m and μ,

$$m = \frac{\mu \sigma T_m}{\lambda L_0}. \tag{34}$$

$\bar{\kappa}$ is the (dimensional) mean curvature. (34) amounts to fixing τ, via (28), and from (24) m is independent of $\tilde{\epsilon}$ in (27).

At this point, only one parameter, $\tilde{\epsilon}$ is left undetermined. $\tilde{\epsilon}$ is chosen to be small enough for accuracy and large enough to make computation practical.

Let us summarize the parameters we have set and how they are chosen or determined.

$$\tilde{S} = \frac{c(T_m - T_0)}{L_0},$$

$$\tilde{\epsilon} = \frac{\delta}{w},$$

$$m = \frac{6\sqrt{2}\delta\sigma}{\tau\lambda T_m} = \frac{\mu\sigma T_m}{\lambda L_0},$$

and

$$\alpha = \frac{\sqrt{2}wL_0^2}{12c\sigma T_m} = \frac{\sqrt{2}w}{12d},$$

where $d = \frac{c\sigma T_m}{L_0^2}$ = capillary length.

$\tilde{S}$ is the dimensionless undercooling (Stefan number). It is the ratio between two energies, sensible to the latent heat. The parameters α and m are related to the physical parameters which characterize the interfacial dynamics such as interfacial energy and kinetic coefficient.

We see that the material parameters $c, \kappa, T_m, L_0, \sigma, \mu$, and the initial undercooling temperature T_0, determine d, m, and $\tilde{S}$. As soon as the length scale w is set, α is also determined, leaving one degree of freedom $\tilde{\epsilon}$, which characterizes the interfacial thickness.

In above, we have taken ϵ, and in turn $\tilde{\epsilon}$, to be a constant. This corresponds to an isotropic system. That is, there are no preferred directions for the growth of the solid crystal. Since many crystals have axes of symmetry, we want to incorporate anisotropy in the model.

2.E. Anisotropy

We relax the restriction that $\tilde{\epsilon}$ be constant and let it depend on the orientation of the solid-liquid interfacial [3],[12], by taking

$$\tilde{\epsilon}(\theta) = \bar{\epsilon}\eta(\theta)$$

where $\theta = tan^{-1}(\frac{\phi_y}{\phi_x})$ is the angle between the normal direction to the interface and the x-axis, and a simple representation of anisotropy through $\eta(\theta)$ is $\eta(\theta) = 1 + \gamma \cos k\theta$. For instance, if $k = 4$, the four preferred directions are the positive and negative x-directions and y-directions. The variation of the second term in the integral in equation (3) without anisotropy is

$$-\frac{\partial}{\partial\phi}(\frac{\bar{\epsilon}^2}{2}|\bigtriangledown\phi|^2) = \bar{\epsilon}^2\bigtriangledown^2\phi.$$

This has to be modified for the anisotropic case to:

$$-\frac{\partial}{\partial\phi}(\frac{[\eta(\theta)]^2}{2}|\bigtriangledown\phi|^2) = [\eta(\theta)]^2\bigtriangledown^2\phi + \text{extra terms}$$

The detailed derivation is as follows. The variation of the integral

$$I[\phi] = \frac{1}{2}\int [\eta(\theta)]^2(\bigtriangledown\phi)^2\, dV \qquad \text{is}$$

$$\begin{aligned}
\delta I &= \int\{\eta^2 \bigtriangledown\phi\cdot\bigtriangledown\delta\phi + (\bigtriangledown\phi)^2\eta\eta'\delta\theta\}\, dV \\
&= \int\{-\delta\phi\bigtriangledown\cdot(\eta^2\bigtriangledown\phi) + \eta\eta'[\phi_x\delta\phi_y - \phi_y\delta\phi_x]\}\, dV \\
&= -\int\{\eta^2\bigtriangledown^2\phi + 2\eta\eta'\bigtriangledown\theta\cdot\bigtriangledown\phi + ((\eta')^2 + \eta\eta'')[\phi_x\theta_y - \phi_y\theta_x]\}\delta\phi\, dV
\end{aligned}$$

where $\eta' = \frac{d\eta}{d\theta}$, and we have used

$$\delta\theta = \frac{\phi_x\delta\phi_y - \phi_y\delta\phi_x}{|\bigtriangledown\phi|^2}.$$

Noting that, with $g(\phi)$ defined in (20), $-\frac{1}{4}g'(\phi) = \phi(\phi - \frac{1}{2})(1-\phi)$, and that, with $p(\phi)$ defined in (29), $p'(\phi) = 30g(\phi)$, the governing equations with anisotropy become

$$\frac{\partial u}{\partial\tilde{t}} + \frac{30g(\phi)}{\tilde{S}}\frac{\partial\phi}{\partial\tilde{t}} = \bigtriangledown^2 u \tag{35}$$

$$\begin{aligned}\frac{\bar{\epsilon}^2}{m}\frac{\partial\phi}{\partial\tilde{t}} &= \phi(1-\phi)[\phi-\frac{1}{2}+30\bar{\epsilon}\alpha\tilde{S}u\phi(1-\phi)] \\ &-\bar{\epsilon}^2\frac{\partial}{\partial x}(\eta(\theta)\eta'(\theta)\frac{\partial\phi}{\partial y})+\bar{\epsilon}^2\frac{\partial}{\partial y}(\eta(\theta)\eta'(\theta)\frac{\partial\phi}{\partial x}) \\ &+\bar{\epsilon}^2\bigtriangledown\cdot(\eta^2(\theta)\bigtriangledown\phi)\end{aligned} \tag{36}$$

In some cases, we want to examine the effect of noise corresponding to thermal fluctuations. We simply add in an extra term $A_n r_n$ artificially:

$$\frac{\bar{\epsilon}^2}{m}\frac{\partial\phi}{\partial\tilde{t}} = \phi(1-\phi)[A_n r_n+\phi-\frac{1}{2}+\cdots]+\cdots,$$

where $r_n = r_n(x)$ is a random number in [-0.5, 0.5], and A_n is the amplitude of the noise.

2.F. Some Explicitly Solvable Cases

We can examine some simple isothermal cases related to Ostwald ripening before we use the above governing equations to examine dendritic growth and coarsening [8],[15].

From the interfacial boundary condition (31), we have

$$u = -\tilde{\Gamma}(\tilde{\kappa}+\frac{\tilde{v}_l}{m})$$

For a circle of radius R this becomes

$$u = -\tilde{\Gamma}(\frac{1}{R}+\frac{\dot{R}}{m}),$$

which for $u = 0$ gives:

$$\dot{R} = -\frac{m}{R} \tag{37}$$

whence

$$R(t) = \sqrt{R_0^2 - 2mt}. \tag{38}$$

This indicates that the circle shrinks until the extinction time $t^* = \frac{{R_0}^2}{2m}$ when it vanishes.

For u = constant $\neq 0$, we have for a circle,

$$-m\int_0^t dt = \int_{R_0}^{R} \frac{\tilde{\Gamma} R}{\tilde{\Gamma} + uR} dR \tag{39}$$

$$-mt = \frac{\tilde{\Gamma}}{u^2}[u(R - R_0) - \tilde{\Gamma} \ln \frac{\tilde{\Gamma} + uR}{\tilde{\Gamma} + uR_0}]$$

0.05in

$$ut = \frac{\tilde{\Gamma}}{m}[R_0 - R(t) - \frac{\tilde{\Gamma}}{u} \ln \frac{\tilde{\Gamma} + uR(t)}{\tilde{\Gamma} + uR_0}]. \tag{40}$$

When u < 0, R(t) decreases if $R_0 < \frac{1}{u}$, and R(t) increases if $R_0 > \frac{1}{u}$.

These explicit solutions are useful for debugging the numerical simulation program. In conclusion, the phase field model above is derived from a single entropy functional. It accounts for positive entropy production and is applicable to non-isothermal situations. Latent heat released when solid is formed at the interface is accounted for with a function of the order parameter. The governing equations are a pair of coupled partial differential equations. We will run numerical simulations with the discretized form of the governing equations.

Chapter 3. Numerical Method

3.A. Simulation Methodology

The objective of the simulations of two dimensional dendritic growth and Ostwald ripening is to examine how the phase field model performs. One great advantage of the phase field model in numerical simulations is that the solid-liquid interface is a thin layer instead of a sharp interface and so the model does not need to keep track of the solid-liquid interface during the computations. The trade-off is that now we have to solve two coupled partial differential equations instead of only one and that we need to sufficiently resolve the gradients of the phase field variable over the thin interfacial layer.

In the simulations, the two governing equations are solved numerically in a two dimensional rectangular grid which is uniformly divided into smaller rectangles of spacing ΔX and ΔY in the x and y directions respectively. The rectangular domain has to be large enough for the dendrite, including that of the side branches, to grow. However, the size of the sub-rectangles has to be small enough to resolve the thin interface. In all our computations, we impose vanishing Neumann boundary conditions for both ϕ and u, that is, there are no fluxes flowing across the external boundaries.

The governing equations are spatially discretized using finite volumes and temporally discretized with explicit time steps. The advantage of such discretization schemes is that they express conservation at the discrete level and are relatively simple to

implement.

It is found that the phase field ϕ does not need to be update numerically as often as the temperature u. Therefore, we include this feature in our code to optimize the efficiency. Since in the region far away from the liquid-solid interface the variation of the phase field ϕ is essentially zero, we only have to compute the flux of ϕ and update ϕ for a small region encompassing the interface. This technique is referred to as **masking** and it further improves the efficiency. The masking applies only to computation of ϕ , not the computation involving the temperature u.

Often, the efficiency of explicit time scheme is limited because the time step has to be kept small to satisfy stability requirements, as we will discuss later. However, we employ a technique called super-time-stepping to improve the efficiency of our explicit scheme. A brief description of super-time- stepping will be given section 3D.

The initial conditions set for dendritic growth computations are that a small solid seed ($\phi = 0$ and u=0) is located in an otherwise undercooled liquid ($\phi = 1$ and $u = -1$) which occupies the entire domain. The large gradient of ϕ between the solid and the liquid in the early stage of the computations could contribute certain error due to the polynomial dependence on ϕ in the phase field equation. In a mesh node, ϕ numerically takes on a certain value which is constant over that entire volume of the node, in a given time step. However, the value of ϕ actually has spatial variation within the node. This variation is more severe when a large gradient of ϕ is present across a small number of nodes. So, the numerical approximation of ϕ is not

very good when the gradient of ϕ is large across the interface in the early stage of the simulations. To circumvent such short-coming, we use a finer mesh in the earlier stage of the computation. With similar idea to the masking technique, we only do computation for part of the entire domain. This is an adaptive mesh, however simple it is. A finer mesh would require much more computation time (halving the size of the mesh nodes would mean the computation time is about 16 times longer because the time steps have to be one fourth of the original). We only use the finer mesh for a comparatively short time. Also the large gradient of both phase field variables, especially the temperature u in the early stage may cause instability or fluctuation. To circumvent this, it is sufficient to simply apply only a few smaller time steps at the beginning of the computations.

The initial condition for the temperature is that the liquid is undercooled to a large degree. As mentioned above, we have to use a large region with fine mesh to facilitate our dendritic growth computation. It would take tremendous amount of computation time if the undercooling is mild. It is because the diffusion of heat would be slow, and since the diffusion of the phase field is even slower than that of heat it makes the situation even worse.

In our two-dimensional simulation performed, the following material parameters for pure nickel are used.

$\sigma = 3.7 \times 10^{-5}\ J/cm^2$,

$T_m = 1728\ K$,

$L = 2350\ J/cm^3$,

$c = 5.42\ J/Kcm^3$, and

$\lambda = 0.155\ cm^2/s$.

The above values are well established. However, the kinetic coefficient μ in equation (32) lies in an estimated range of values. In most cases, we use $\mu = 285cm/Ks$ which corresponds to m=0.05. We also choose $\tilde{\epsilon}$ in equation (25) to be 0.005, and mesh node size 0.0075 for both ΔY and ΔX in those cases.

Nickel is chosen because the large degree of undercooling is experimentally obtainable. Also, the studies of solidification of an alloy consisting of nickel and another metal are important in metallurgical science; therefore, better understanding of pure nickel is desirable. Initially, a solid ellipse is introduced. The long term behavior of the dendrite is not much dependent upon the initial shape.

In Ostwald ripening, we use a much smaller degree of undercooling since it occurs in the later stage of solidification when temperature gradients have smoothened out. Initially there are two solid circles (spheres). They more or less maintain their shape and the growth or shrinking rate is mild throughout the simulation. It is sufficient to use a coarser grid.

The tip velocity v_{tip} of the main branch of the dendrite in the x-direction is computed as

$$v_{tip} = \frac{x_{tip}(t_b) - x_{tip}(t_a)}{(t_b - t_a)} ,$$

where $x_{tip}(t)$ is the position of the tip at some time t.

The position of the tip can be defined as some fixed value of ϕ between 0.1 and 0.9,

the range we will use for plotting the interface. We arbitrarily choose $\phi = 0.8$. Since ϕ takes on discrete values numerically and the interfacial thickness is about 5 nodes across, we settle for the value as close as possible to but less than 0.8. $(t_b - t_a)$ is an integral multiple of a (super) time step; it is chosen so that the tip advances at least one node in its growth direction for all time, except the very end of the computation when the tip is very close to the external boundary and stops growing.

The radius of curvature R of the tip is estimated with $\frac{1}{R} \approx \frac{\phi_{yy}}{\phi_x}$ in the discretized form.

3.B. Discretization

We describe the discretization of the differential equations as follows. Let the rectangular domain $[0, X] \times [0, Y]$ be divided into $M_x \times M_y$ rectangles (control volumes) with M_x and M_y subdivisions in x- and y-direction respectively.

Let x_i denote the distance from the origin to the center of i^{th} subdivision (node) in the x-direction for $i = 0, 1, 2, ..., M_x + 1$. The 0^{th} and $M_x + 1^{th}$ nodes coincide with the external boundaries, so these two nodes have zero volume (length). Let $x_{i-\frac{1}{2}} = 0.5 \times (x_{i-1} + x_i)$ for $i = 2, 3, ...M_x - 1$, be the left face of the i^{th} subinterval, taken half way between the nodes x_{i-1} and x_i. Also, $x_{\frac{1}{2}} = x_0 =$ left boundary and $x_{M_x+\frac{1}{2}} = x_{M_x+1} =$ right boundary.

In a similar manner, we write y_j, $j = 0, 1, 2, ..., M_y + 1$ for the y-nodes, and $y_{j-\frac{1}{2}}$ the y-faces. We denote $\Delta x_i = x_i - x_{i-1}$, and $\Delta y_j = y_j - y_{j-1}$.

The faces of the (i,j) control-volume are

$Ax_{i-\frac{1}{2},j} = y_{j+\frac{1}{2}} - y_{j-\frac{1}{2}}$, $i = 1, 2, \cdots, M_x + 1, j = 1, 2, \cdots, M_y$ for those perpendicular to the x-axis, and

$Ay_{i,j-\frac{1}{2}} = x_{i+\frac{1}{2}} - x_{i-\frac{1}{2}}$, $i = 1, 2, \cdots, M_x, j = 1, 2, \cdots, M_y + 1$ for those perpendicular to the y-axis.

The volume of the rectangular control-volume is $V_{i,j} = [x_{i+\frac{1}{2}} - x_{i-\frac{1}{2}}] \cdot [y_{j+\frac{1}{2}} - y_{j-\frac{1}{2}}]$.

In general, integer subscripts refer to nodal values, and half integer subscripts to facial values.

The discretization of equation (34) is as follows.

First, we write

$$-(\eta\eta'\phi_y)_x + (\eta\eta'\phi_x)_y = \bigtriangledown \cdot [\eta(\theta)\eta'(\theta)(-\phi_y, \phi_x)]$$

where $(-\phi_y, \phi_x) = \vec{\tau}$ is the tangential direction to $\phi(x, y)$= constant .

Integrating the above equation over $V_{i,j}$,

$\int_{V_{i,j}} \bigtriangledown \cdot (\eta\eta'\vec{\tau})\, dV = \int_{\partial V_{i,j}} \eta\eta'\vec{\tau} \cdot \vec{n}\, dA = \int_{\partial V_{i,j}} [(-\eta\eta'\phi_y)n_x + (\eta\eta'\phi_x)n_y]\, dA$

$= -\int_{Ax_{i+\frac{1}{2},j}} \eta\eta'\phi_y\, dAx_{i+i\frac{1}{2},j} + \int_{Ax_{i-\frac{1}{2},j}} \eta\eta'\varphi_y\, dAx_{i-\frac{1}{2},j}$

$\quad + \int_{Ay_{i,j+\frac{1}{2}}} \eta\eta'\phi_x\, dAy_{i,j+\frac{1}{2}} + \int_{Ayi,j-\frac{1}{2}} \eta\eta'\phi_x\, dAy_{i,j-\frac{1}{2}}$

$\approx -Ax_{i+\frac{1}{2},j}\eta_{i+\frac{1}{2},j}\eta'_{i+\frac{1}{2},j}(\phi_y)_{i+\frac{1}{2},j} + Ax_{i-\frac{1}{2},j}\eta_{i-\frac{1}{2},j}\eta'_{i-\frac{1}{2},j}(\phi_y)_{i-\frac{1}{2},j}$

$\quad + Ay_{i,j+\frac{1}{2}}\eta_{i,j+\frac{1}{2}}\eta'_{i,j+\frac{1}{2}}(\phi_x)_{i,j+\frac{1}{2}} - Ay_{i,j-\frac{1}{2}}\eta_{i,j-\frac{1}{2}}\eta'_{i,j-\frac{1}{2}}(\phi_x)_{i,j-\frac{1}{2}}$,

Now, we need an approximation to $(\varphi_y)_{i+\frac{1}{2},j}, (\phi_y)_{i-\frac{1}{2},j}, (\phi_x)_{i,j+\frac{1}{2}}$, and $(\varphi_x)_{i,j-\frac{1}{2}}$ (at faces) in terms of nodal values of ϕ.

Having chosen the faces midway between nodes, we simply interpolate:

$$(\phi_y)_{i-\frac{1}{2},j} = \frac{(\phi_y)_{i-1,j} + (\phi_y)_{i,j}}{2}.$$

and we approximate the nodal values of these derivatives by centered differences:

$$(\phi_y)_{i,j} \approx \frac{\phi_{i,j+1} - \phi_{i,j-1}}{y_{j+1} - y_j}.$$

However, when we discretize the PDE (36) by integration over each control volume $V_{i,j}$, the natural discrete value of ϕ will be its average over $V_{i,j}$, denoted $\Phi_{i,j}$,

$$\Phi_{i,j} = \frac{1}{V_{i,j}} \int_{V_{i,j}} \phi \, dV.$$

Therefore, we assume the nodal value $\phi_{i,j}$ represents the mean value $\Phi_{i,j}$:

$$\phi_{i,j} \approx \Phi_{i,j},$$

and so we get

$$(\phi_y)_{i-\frac{1}{2},j} \approx \frac{1}{2}[\frac{\Phi_{i-1,j+1} - \Phi_{i-1,j-1}}{y_{j+1} - y_{j-1}} + \frac{\Phi_{i,j+1} - \Phi_{i,j-1}}{y_{j+1} - y_{j-1}}].$$

Other $(\phi_x)_y$ and $(\phi_y)_x$ terms at their corresponding faces are derived in a similar manner, and so, we have

$$\begin{aligned}
&\int_{V_{i,j}} \nabla \cdot (\eta\eta'\vec{\tau}) \, dV \approx \\
&-\frac{1}{2} Ax_{i+\frac{1}{2},j} \, \eta_{i+\frac{1}{2},j} \, \eta'_{i+\frac{1}{2},j} [\frac{\Phi_{i,j+1} - \Phi_{i,j-1}}{y_{j+1} - y_{j-1}} + \frac{\Phi_{i+1,j+1} - \Phi_{i+1,j-1}}{y_{j+1} - y_{j-1}}] \\
&+\frac{1}{2} Ax_{i-\frac{1}{2},j} \, \eta_{i-\frac{1}{2},j} \, \eta'_{i-\frac{1}{2},j} [\frac{\Phi_{i-1,j+1} - \Phi_{i-1,j-1}}{y_{j+1} - y_{j-1}} + \frac{\Phi_{i,j+1} - \Phi_{i,j-1}}{y_{j+1} - y_{j-1}}] \\
&+\frac{1}{2} Ay_{i,j+\frac{1}{2}} \, \eta_{i,j+\frac{1}{2}} \, \eta'_{i,j+\frac{1}{2}} [\frac{\Phi_{i+1,j} - \Phi_{i-1,j}}{x_{i+1} - x_{i-1}} + \frac{\Phi_{i+1,j+1} - \Phi_{i-1,j+1}}{x_{i+1} - x_{i-1}}] \\
&-\frac{1}{2} Ay_{i,j-\frac{1}{2}} \, \eta_{i,j-\frac{1}{2}} \, \eta'_{i,j-\frac{1}{2}} [\frac{\Phi_{i+1,j-1} - \Phi_{i-1,j-1}}{x_{i+1} - x_{i-1}} + \frac{\Phi_{i+1,j} - \Phi_{i-1,j}}{x_{i+1} - x_{i-1}}].
\end{aligned} \tag{40}$$

For convenience in notation, let us represent (36) as

$$\frac{\bar{\epsilon}^2}{m}\phi_t = W(\phi, u) - \bar{\epsilon}^2(F_2)_x + \bar{\epsilon}^2(F_1)_y + \bar{\epsilon}^2 \bigtriangledown \cdot F \tag{41}$$

where $W(\phi, u)$ represents the first term on the right hand side of equation (36), $F_2 = \eta\eta'\phi_y, F_1 = \eta\eta'\phi_x$, and $F = \eta^2 \bigtriangledown \phi$. Integrating (41) over the control volume $V_{i,j}$ and with respect to time over the n^{th} time step with duration Δt_n, and denoting by $\Phi^n_{i,j}$ the mean value of ϕ over $V_{i,j}$ at time t_n , we have

$$\frac{\bar{\epsilon}^2}{m}[\Phi^{n+1}_{i,j} - \Phi^n_{i,j}]V_{i,j} = \Delta t_n V_{i,j} W^n_{i,j} + \Delta t_n \bar{\epsilon}^2 \sum(fluxes),$$

whence we obtain the explicit updating of $\Phi_{i,j}$ at time t_{n+1}

$$\Phi^{n+1}_{i,j} = \Phi^n_{i,j} + \frac{m}{\bar{\epsilon}^2}\Delta t_n \cdot W^n_{i,j} + \frac{m\Delta t_n}{V_{i,j}}\sum(fluxes).$$

Here $\sum(fluxes)$ is the sum of discretized fluxes due to F_2, F_1, and F.

We have already written down the fluxes resulting from the term $-F_2 + F_1$, see (40). The discretized form for the other flux term and W are straight forward. Thus we obtain the explicit updating formula

$$\begin{aligned}
\Phi^{n+1}_{i,j} \;=\; & \Phi^n_{i,j} + \frac{m\Delta t_n}{V_{i,j}}\{\frac{1}{\bar{\epsilon}^2}W^n_{i,j}V_{i,j} \\
& -\frac{1}{2}Ax_{i+\frac{1}{2},j}\,\eta_{i+\frac{1}{2},j}\,\eta'_{i+\frac{1}{2},j}\,[\frac{\Phi_{i,j+1}-\Phi_{i,j-1}}{y_{j+1}-y_{j-1}} + \frac{\Phi_{i+1,j+1}-\Phi_{i+1,j-1}}{y_{j+1}-y_{j-1}}] \\
& +\frac{1}{2}Ax_{i-\frac{1}{2},j}\,\eta_{i-\frac{1}{2},j}\,\eta'_{i-\frac{1}{2},j}\,[\frac{\Phi_{i-1,j+1}-\Phi_{i-1,j-1}}{y_{j+1}-y_{j-1}} + \frac{\Phi_{i,j+1}-\Phi_{i,j-1}}{y_{j+1}-y_{j-1}}] \\
& +\frac{1}{2}Ay_{i,j+\frac{1}{2}}\,\eta_{i,j+\frac{1}{2}}\,\eta'_{i,j+\frac{1}{2}}\,[\frac{\Phi_{i+1,j}-\Phi_{i-1,j}}{x_{i+1}-x_{i-1}} + \frac{\Phi_{i+1,j+1}-\Phi_{i-1,j+1}}{x_{i+1}-x_{i-1}}] \\
& -\frac{1}{2}Ay_{i,j-\frac{1}{2}}\,\eta_{i,j-\frac{1}{2}}\,\eta'_{i,j-\frac{1}{2}}\,[\frac{\Phi_{i+1,j-1}-\Phi_{i-1,j-1}}{x_{i+1}-x_{i-1}} + \frac{\Phi_{i+1,j}-\Phi_{i-1,j}}{x_{i+1}-x_{i-1}}].
\end{aligned} \tag{42}$$

$$+Ax_{i+\frac{1}{2},j}\,\eta^2_{i+\frac{1}{2},j}\,\frac{\Phi^n_{i+1,j}-\Phi^n_{i,j}}{x_{i+1}-x_i}-Ax_{i-\frac{1}{2},j}\,\eta^2_{i-\frac{1}{2},j}\,\frac{\Phi^n_{i,j}-\Phi^n_{i-1,j}}{x_i-x_{i-1}}$$
$$+Ay_{i,j+\frac{1}{2}}\,\eta^2_{i,j+\frac{1}{2}}\,\frac{\Phi^n_{i,j+1}-\Phi^n_{i,j}}{y_{j+1}-y_j}-Ay_{i,j-\frac{1}{2}}\,\eta^2_{i,j-\frac{1}{2}}\,\frac{\Phi^n_{i,j}-\Phi^n_{i,j-1}}{y_j-y_{j-1}}\},$$

$i=1,2,...,M_x,\ j=1,...,M_y,$

where

$$W^n_{i,j}=\Phi^n_{i,j}(1-\Phi^n_{i,j})[\Phi^n_{i,j}-\frac{1}{2}+30\bar{\epsilon}\alpha\tilde{S}U^n_{i,j}\Phi^n_{i,j}(1-\Phi^n_{i,j})]$$

with $U^n_{i,j}$ = mean dimensionless temperature u over $V_{i,j}$ at time t_n.

The discretization of equation (35) is obtained similarly. It results in the following explicit updating of $U^n_{i,j}$:

$$\begin{aligned}U^{n+1}_{i,j} &= U^n_{i,j}-\frac{30}{\tilde{S}}g(\Phi^n_{i,j})[\Phi^{n+1}_{i,j}-\Phi^n_{i,j}]\\ &+\frac{\Delta t_n}{V_{i,j}}\{Ax_{i+\frac{1}{2},j}\frac{U^n_{i+1,j}-U^n_{i,j}}{x_{i+1}-x_i}-Ax_{i-\frac{1}{2},j}\frac{U^n_{i,j}-U^n_{i-1,j}}{x_i-x_{i-1}}\\ &+Ay_{i,j+\frac{1}{2}}\frac{U^n_{i,j+1}-U^n_{i,j}}{y_{j+1}-y_j}-Ay_{i,j-\frac{1}{2}}\frac{U^n_{i,j}-U^n_{i,j-1}}{y_j-y_{j-1}}\}.\end{aligned}$$

3.C. Stability Condition

We have mentioned above that there are stability requirements to be met. We now describe these stability restrictions on the magnitude of time step for both ϕ and u. It is found that the restriction is more severe on u than on ϕ (u diffuses faster than ϕ); therefore, ϕ does not need to be updated numerically as often.

For ϕ updating, we rewrite (42) as

$$\Phi^{n+1}_{i,j}=\Phi^n_{i,j}+\frac{m\Delta t_n}{V_{i,j}}[\frac{1}{\bar{\epsilon}^2}W^n_{i,j}-Ax_{i+\frac{1}{2},j}\,\eta_{i\frac{1}{2},j}\,\eta'_{i,+\frac{1}{2},j}[\ldots]+\ldots]$$

$$+\frac{m\Delta t}{\Delta X^2}[(\eta^2_{i+\frac{1}{2}}+\eta^2_{i-\frac{1}{2},j}+\eta^2_{i,j+\frac{1}{2}}+\eta^2_{i,j-\frac{1}{2}})\Phi^n_{i,j}+\ldots],$$

noting that $\frac{Ax}{V}$ and $\frac{Ay}{V}$ gives ΔX and ΔY, and $\Delta X = \Delta Y$ with our uniform grid. Recall that $\eta = 1+\gamma\cos(k\theta)$, that is $\eta^2 < (1+\gamma)^2$. If we ignore the nonlinear term involving $W^n_{i,j}$, the following inequality has to be satisfied for stability, noting that the terms not involving in $\Phi^n_{i,j}$ on the RHS do not affect the stability restriction:

$$[1-4\frac{m(1+\gamma)^2}{\Delta X^2}\Delta t]\Phi^n_{i,j} > 0.$$

That is, $\Delta t < \frac{\Delta X^2}{4m(1+\gamma)^2} \approx \frac{\Delta X^2}{4m}$.

If the nonlinear term is included, we get

$$\Delta t < \frac{\Delta X^2}{4m(1+\gamma)^2 - \frac{m}{\epsilon^2}f(\Phi^n_{i,j})\Delta X^2},$$

where $\Phi^n_{i,j}f(\Phi^n_{i,j}) = W^n_{i,j}$. The effect of the nonlinear term on stability restriction is small according to the expression. and for $\Delta X = 0.0075$, we need $\Delta t \approx 2.8\times 10^{-4}$.

Similarly, for u updating,

$$U^{n+1}_{i,j} = U^n_{i,j} - \frac{30}{\tilde{S}}g(\Phi^n_{i,j})[\Phi^{n+1}_{i,j}-\Phi^n_{i,j}] + \frac{\Delta t_n}{\Delta X^2}[-4U^n_{i,j}+\ldots\ ].$$

However, from (42), $\Phi^{n+1}_{i,j} - \Phi^n_{i,j} = \frac{m}{\epsilon^2}\ \Delta t_n[\Phi^n_{i,j}(1-\Phi^n_{i,j})]^2\cdot 30\bar{\epsilon}\alpha\tilde{S}U^n_{i,j}+\ldots$

So, we get

$$U^{n+1}_{i,j} = U^n_{i,j}[1-\frac{4\Delta t}{\Delta X^2}] - \frac{m}{\epsilon^2}(g(\Phi^n_{i,j}))^2\Delta t+\cdots.$$

Noting that the maximum of $g(\Phi^n_{i,j})$ is $\frac{1}{4}$, we get

$$\Delta t < \frac{\Delta X^2}{4+\frac{900}{16}\frac{m\alpha}{\epsilon}\Delta X^2}.$$

For $\Delta X = 0.0075$, this implies $\Delta t \approx 1.2 \times 10^{-5}$.

However, in computation we cannot just use the maximum Δt allowed. We find by trial and error that a time step separation about $0.8\Delta t$ is practical.

We see that the restrictions on Δt for ϕ and u are not the same, that is the motivation for different magnitudes of time steps in updating different field variables.

3.D. Super-time-stepping

A brief description of super-time-stepping is as follows, see [1]. The method can be used in parabolic problems, like the one we have here. The demand that stability has to be fulfilled at every time step in an explicit scheme is relaxed. Stability is required only at the end of every N steps. The N steps make up one super time step and each of the N steps is a sub-step of the super time step. The sub-steps are no longer uniform steps. The total time of the N steps is about N^2 times the original explicit time step.

We write the explicit time scheme for a parabolic problem as

$$U^{n+1} = U^n - \Delta t A U^n,$$

where n indicates the n^{th} step, with $n = 0, 1, 2, 3....$, and A is a matrix operator of the discretized partial differential equation in the parabolic problem.

Suppose that Δt_{expl} is the time step with maximum time duration allowable by stability condition for an explicit scheme. We choose the N substeps of a superstep as:

$$\tau_i = \Delta t_{expl} [(\mu - 1) \cos(\frac{2i-1}{N} \frac{\pi}{2}) + 1 + \mu]^{-1},$$

where $i = 1, 2, ..., N$ and μ is some number between 0 and $\frac{\lambda_{min}}{\lambda_{max}}$, $\lambda_{min}, \lambda_{max}$ being the minimum and maximum eigenvalues of the matrix A.

The duration of a super time step is $\Delta T = \tau_1 + \tau_2 + ... + \tau_N$. Note that $\Delta T \rightarrow N^2 \Delta t_{expl}$ as $\mu \rightarrow 0$. Therefore, the super-time-stepping scheme is up to N times faster than the original scheme. The choice of N and μ is found by trial and error. In our code, we use $N = 3$ and $\mu = 0.05$. We get $\Delta T \approx 2\Delta t_{expl}$. It is not as good as some other cases without non-linear terms we have tried on, where 4 fold improvement in efficiency is obtainable. However, here we have some non-linear terms and large initial gradients. Instability sets in when we choose large N or small μ, trying to further improve efficiency. After the initial steep gradients relax, one can use larger N and smaller μ to get better efficiency.

Chapter 4. Results and Conclusions

Every morphological pattern shown is the interface between the solid and liquid. The thin layer of interface included has values of ϕ between 0.1 and 0.9. We see that the interfacial layer is about 4 or 5 mesh nodes in width. It shows that a right choice of ϵ is crucial because it has to be small enough to resolve the thin interface, but it should not be so small that it tremendously slows down the computation. Note that small ϵ means small control volume of the nodes, that is the number of nodes increases, and the duration of time step has to be decreased also.

4.A. Testing cases

Explicit solutions are known only for constant temperature u and circular shapes, as presented in equations (38) and (40).

a) Shrinking circle

We test the case for shrinking circle setting $u = 0$ as in equation (37). Initially we start with a solid circle in an otherwise undercooled liquid. The circle shrinks at the rate according to equation (38), except when its radius is almost zero due to numerical error when only a few mesh nodes are involved. The circular shape is maintained.

b) Expanding and shrinking circle

We also look at the case as in equation (39), setting u =constant $\neq 0$. Initially we

have two circles whose sizes are according to the description for equation (40), one with $R > \frac{1}{u}$ and the other with $R < \frac{1}{u}$. One circle expands and the other shrinks as expected. The circles maintain their shapes. This case is used for comparing with Ostwald Ripening presented later.

In both the cases above, the circles maintain their circular shape. This is because they are in an isothermal environment where no thermal gradient is present. Therefore, there is no driving mechanism for finger growth.

4.B. Dendritic Growth

For all cases of dendritic growth computation, the conditions are as follows.

We use a 512×512 grid with $\Delta X = \Delta Y = 0.0075$. Initially, there is a solid of half of an ellipse with major semi-axis $20\Delta X$ and minor semi-axis $10\Delta Y$. The initial temperature is $u = 0$ for solid and $u = -1$ for supercooled liquid. In the morphological pattern shown, the mesh nodes with $0.1 < \phi < 0.9$ are plotted.

We use $m = 0.05$, $\alpha = 400$, $\epsilon = 0.005, S = 0.5$, and time-step $\Delta t = 8 \times 10^{-6}$

(a) without noise (amplitude of noise A=0)

(a) (i) without anisotropy (strength of anisotropy $\gamma = 0$).

See Figure 1. [1]

We see that fingers are formed and grow in all directions because bump like struc-

[1] All figures may be found in Appendix A.

tures are preferred and there is no anisotropy or preferred directions. Tip splitting of fingers is observed. Qualitatively, we can explain this phenomenon in this way. When a finger tip grows to a certain size so that the finger tip has a relatively flat interface, and if the temperature gradient across the interface is large enough, a bump begins to form and grow there and the finger tip splits into two. Without anisotropy, the fingers undergo tip splitting and then branch out in all directions at a more or less the same rate. Therefore, no preferred directions are observed. This process goes on and more fingers are formed. A rather complex amorphous structure develops eventually.

A finger will grow at a certain rate when it is far away from external boundaries and there is room for it to grow. How far the finger will reach out also depends on the competition against other fingers around. If there are already some more dominant fingers around, it will not grow too far. So, we see some small fingers formed among other larger fingers, but they stop growing when they encounter competition from larger fingers surrounding them.

(a)(ii) with anisotropy.

For the cases with anisotropy, recall that the expression for it is $\eta(\theta) = (1+\gamma\cos k\theta)$, where γ is the strength of anisotropy. We set $k = 4$ (four-fold) for nickel; therefore, the preferred directions are along the x- and the y- directions. The data for the region near the y-axis is not as accurate as those for other regions farther away from the boundary. It does not adversely affect us because we are more interested in the

development of the main branches, its tip kinetics, and the side branches growing from it in the case with noise.

For very small values of γ, we still see tip splitting and no dominant main trunk is formed. Branches are growing in all directions. The branches in diagonal directions are not suppressed very much and the branches in the x- and y-directions do not become dominant. It is similar to the case without anisotropy. Therefore, we do not include any figure for that.

Figure 2 shows the case with anisotropy strength $\gamma = 0.01$. That is, $\eta(\theta) = 1 + 0.01\cos(4\theta)$, having a range from 0.09 to 1.01.

For γ as small as 0.01, we already see a pattern with a main branch extending in the x-direction. There are still some fingers growing at an angle to the x-axis; however they are dominated by the main trunk in the x-direction and branches in the y-direction.

In Figure 3, γ has been increased to 0.018. It is clear that a dominant main trunk in the x-direction and large branches in the y-direction are formed. The fingers growing diagonally are greatly suppressed; they reach out only a short distance and stop growing due to competition from solid near by. It should be noted that the main trunk has no side branches growing out of it for the time duration in our computation.

It is obvious that anisotropy affects the evolution of the dendrite to a great extent. For anisotropy strength as small as 0.018, we already obtain a needle shaped crystal when the dendrite tip of the main branch grows steadily and diagonal branch growth

is greatly suppressed.

For this case, we also computed the velocity, temperature, and radius of curvature of the tip of the main branch. They are not much affected when we add in noise; therefore, we only present the results obtained there.

(b) With noise

The cases we have examined so far are done without noise added in. We want to simulate the fact that there is thermal fluctuation. Therefore, we artificially put in a noise term. The amplitude of the noise is small.

(b)(i) Amplitude of noise $A = 0.018$, strength of anisotropy $\gamma = 0.018$,

see Figure 4.

The upshot is that there are side branches protruding from the main trunk. These side branches are about perpendicular to the main branches due to the four-fold anisotropy. The noise we introduced artificially corresponds to thermal noise physically. The noise makes it easier to form small bumps on the main branch in their early stage. Once a small bump is formed it continues to grow and becomes a side branch. We have mentioned that there are no side branches growing from the main branch when noise is absent and there is a certain degree of anisotropy. In this case, however, we observe such side branches. It shows that the presence of noise is essential to the formation of such side branches with anisotropy. The morphological pattern we obtain in this case shows a lot of similarity compared to the patterns obtained in experiments for materials with four-fold anisotropy. All the main features

are there. A needle like main branch dominates and side branches growing from and perpendicular to it. We have noted the importance of this case; therefore , we also compute the velocity, temperature, and radius of curvature of the tip of the main branch. We will discuss these physical variables below.

However, let us look at the isotherms obtained at the same time as the dendritic pattern just shown.

(b) (ii) Temperature Gradient

Temperature gradient is expressed through several isotherms, shown in Figure 5. The isotherms in the figure are at $-0.01, -0.04, -0.07, -0.1, -0.3, -0.5, -0.7$, and -0.9, counting from left to right. We observe that the spread of the isotherms is much wider than the interfacial thickness. It confirms that the diffusion of heat is much faster than that of the phase field variable ϕ.

The morphological pattern we have examined is important. Photographs of dendrite formed during solidification processes are taken in experiments. We have already discussed the similarity between the dendritic pattern experimentally obtained and the pattern in one of our cases. However, certain important kinetic variables are also recorded in experiments.

(c) Kinetic Variables

We can compare the computed values of the kinetic variables in our simulation to the corresponding experimental values. The benchmark experimental measurement is the tip velocity of the main branch. It is a measure of the speed of growth of

the dendrite. It is important since the speed of growth is one of the factors that profoundly affect the final properties of the material. Also, knowing the time needed for the material to solidify is crucial. Another kinetic variable measured is the tip radius. Therefore, we calculate these variables in our simulation, and we also record the tip temperature.

Figures 6, 7, and 8 show the tip velocity, temperature, and radius of curvature, respectively. We note that the tip velocity is in the order of 10^1, the tip radius is of order 10^{-2}, and the tip temperature is of order -10^{-2}. From the figure, the dimensionless tip velocity is about 6.8 most of the time and is higher for some short durations. If we take 7 as the velocity, it corresponds to $50m/s$ approximately. The experimental measurement is from 37 to $48m/s$ and a calculation estimates that the range is wider for two-dimensional cases as what we have here. Our result of tip velocity is comparable to experimental data, although we note that in our simulations we have two-dimensional cases while in experiments there are three-dimensional cases.

The tip radius is about 0.045, which corresponds to $10^{-7}m$. The tip temperature is about -0.12.

We use equation (32) where $\tilde{\Gamma} = 5.976 \times 10^{-4}$ for the values of material parameters for nickel listed earlier. With the dimensionless tip velocity and radius of curvature, we calculated that the tip temperature is about -0.1 using (32). The tip temperature obtained from our numerical simulation is comparable to this value.

The tapering off of the tip velocity, the increase in the tip radius, and the quick

approach to zero of the tip temperature at the end of the simulation are because the main branch has grown to a point that the tip is very close to the external boundary.

4. C. Ostwald ripening

We next turn to Ostwald ripening. In happens in the later stage of solidification. The branches would fuse together, driven mainly by the surface tension effect. The regions with large radius of curvature grow bigger and bigger, at the expense of smaller ones. However, it would take too much computation time and it is not easy to visualize the effect of Ostwald ripening if we perform the simulation of dendritic growth until the fingers merge. Therefore, we only study the case of how two circle interact and compete with each other. It is a simple case but it shows the basic essence of Ostwald ripening. In the case we examine, we set it as an isotropic case without noise. Such choices are obvious because we want to see to what degree the circular shape is maintained. Initially, the degree of undercooling, especially the difference between that of the solid and liquid, is set to be small. If we set a large difference of undercooling like in the cases of dendritic growth above, the circles grows fingers, and it destroys our purpose. Note that this is not an isothermal case and we use the same set of coupled equations as in dendritic growth above. It is not to be confused with the case of shrinking and expanding circles according to equation (40).

Figure 9 shows the case with parameters: $A = 0, m = 0.05, \alpha = 400, \epsilon = 0.005, S =$

0.5, 128×128 grid, $\Delta X = \Delta Y = 0.0075$, initial temperature for solid $u = -0.004$, for liquid $u = -0.005$.

We observe that the larger circle grows larger and larger while the smaller becomes smaller and smaller and eventually disappears (at dimensionless time = 0.43). It is consistent with the theory described above. The main driving force here is surface tension. The shapes of the circles are more or less maintained. No finger formation is observed. It is because the temperature gradient is low. It is not strong enough to counteract the surface tension effect which favors the regions with large radius of curvature. So, fingers having small radius of curvature are suppressed. The smaller circle, having smaller radius of curvature than the larger one, does not survive for long. We also note that in Figure 9d, the remaining large solid is not exactly a circle. However, it rounds out and becomes smoother at a later time. Again, it is because regions with large radius of curvature is favorable. Even a slight difference in curvature is ironed out when the parts with smaller radius of curvature flatten out. We do not include that figure because visually the difference is not very obvious.

We have tried some cases with large thermal gradient. Finger growth on the circles is observed. It confirms that finger growth appears only in high temperature gradient situations.

We want to note that in the papers we have encountered, Ostwald ripening simulation is done for an alloy where, instead of temperature gradient, a concentration gradient is present and drives the process in an isothermal environment. Density is also assumed constant there. In our case, temperature gradient drives the process.

4.D. Conclusion

We have studied dendritic growth and Ostwald ripening in supercooled solidification. A phase field method with an entropy functional formulation is used. In this formulation the positive production of entropy is accounted for and we can examine both isothermal and non-isothermal cases. We take advantage of the phase field method in which interface tracking is not necessary and employ numerical simulations in various cases of dendritic growth and Ostwald ripening. In our numerical computations, we use a finite volume method on a two dimensional rectangular grid and an explicit scheme for time stepping. We use several techniques to speed up our runs. These include super-time-stepping, different time updating for ϕ and u, and the masking method. The super-time-stepping method lessens the stability restriction and therefore improves the efficiency. The different update time of ϕ and u is exploited. ϕ does not have to be updated as often as u since u diffuses more rapidly than ϕ. As computation of the fluxes of ϕ is much more cumbersome than that of u , updating ϕ less often improves the efficiency . We also exploit the fact that far away from the interface the variation of ϕ is practically zero, that is, ϕ is either very close to 0 (bulk solid) or 1 (bulk liquid) away from the solid-liquid interface. Therefore, we only need to compute the fluxes of ϕ around the interface. We use the range of ϕ from 0.1 to 0.9 to indicate the interface. The mask we use is that for every node with ϕ between 0.1 and 0.9, 9×9 nodes (and their faces) around it are included

as the mask (4 nodes wide in both positive and negative x- and y-direction). Since the interface is thin and the area it occupies is small (especially in the early stage) compared to the entire domain, we save a lot of time by not computing fluxes of ϕ for the whole region but only for part of it.

Each of the three techniques mentioned above improves the efficiency by a factor of two roughly.

We observed that in dendritic growth without anisotropy, the solid branches out in all directions as it grows. It is because a finger like structure can diffuse heat faster than a flat surface in a negative temperature gradient as it is present when the liquid is supercooled.

We then introduce anisotropy and examine its effect on dendritic growth. We assume four-fold symmetry ($k = 4$), so that the preferred directions are in the x- and y-directions. Even with low anisotropy, a dominant branch grows in the x-direction away from the external boundary. Some branches in the y-direction are also formed. The growth of diagonal side branches is inhibited. When noise is introduced, side branches growing from and perpendicular to the main branch are observed. Only a small amplitude of noise is sufficient to induce this. Thermal noise facilitates the formation of side branches. In this case, we obtain a dendritic structure which has all the essential features observed in dendrites formed in experiments. The tip velocity and tip radius of curvature of the main branch are calculated. They are comparable to experimental measurements. The tip temperature is also noted. The isotherms in this case are recorded.

In Ostwald ripening, which happens in later stage of solidification, surface tension is the main driving mechanism when the temperature gradient is small. Solid regions with larger radius of curvature consume the solid regions with smaller radius of curvature. This is demonstrated with a simple case of two solid circles competing with each other. The larger circle survives and grows bigger at the expenses of the smaller circle.

References

1. V. Alexiades, G. Amiez, P.A. Gremaud, Communications in Numerical Methods in Engineering, **12** (1996), 31-42.

2. V. Alexiades, A.D. Solomon, Mathematical Modeling of Melting and Freezing Processes, Hemisphere, Washington D.C., 1993.

3. S.M. Allen, J.W. Cahn, Acta Metallurgica **27** (1979) 1085-95.

4. G. Caginalp, P. Fife, Physical Review B **33** (1986) 7792-94.

5. G. Caginalp, P. Fife, Physical Review B **34** (1986) 4940-43.

6. G. Caginalp, E. Socolvsky, SIAM J. Sci. Comput. **15** (1994) 106-126.

7. M.E. Glicksman, Isothermal Dendritic Growth Experiment, http://www.rpi.edu/locker/56/000756/.

8. C.M. Eillot, A.R. Gardiner, T.K. Kuhn, Research Report #96/17 (1996), U. of Sussex, UK.

9. N. Goldenfeld, Lectures on Phase Transition and the Renormalization Group, Addison Wesley, Reading, 1992.

10. D. Kahaner, C. Moler, S. Nash, Numerical Methods and Software, Prentice Hall, Engelwood Cliffs, 1989.

11. D. Kessler, J. Koplik, H. Levine, Advances in Physics **37** (1988) 255-339.

12. G.B. McFadden, A.A. Wheeler, R.J. Braun, S.R. Coriell, R.F. Sekerka, Physical Review E **48** (1993) 2016-2024.

13. B.T. Murray, A.A.Wheeler, M.E. Glicksman, J. Crystal Growth, **154** 1995) 386-400.

14. O. Penrose, P.C. Fife, Physica D **43** (1990) 44-62.

15. E.R. Rubinstein, M.E. Clicksman, J. Crystal Growth **112** (1991) 84-96, 97-110.

16. P.W. Voorhees, J. Statistical Physics **38** (1985) 231-252.

17. S.L. Wang, R.F. Sekerka, A.A. Wheeler, B.T.Murray, S.R. Coriell, R.J. Braun, G.B. McFadden, Physica D **69** (1993) 189-200.

18. S.L.Wang, R.F. Sekerka, J. Computational Physics **127** (1996) 110-117.

19. J.A. Warren, IEEE Computational Sci and Eng, Summer 1995 38-49.

20. J.A. Warren, B.T. Murray, Modelling Simul. Mater. Sci. Eng. **4** (1996) 215-229.

21. A.A. Wheeler, W.J. Boettinger, G.B. McFadden, Physical Review A **45** (1992) 7424-39.

22. A.A. Wheeler, B.T. Murray, R.J. Schaefer Physics D **66** (1993) 243-262.

Appendices

Appendix A

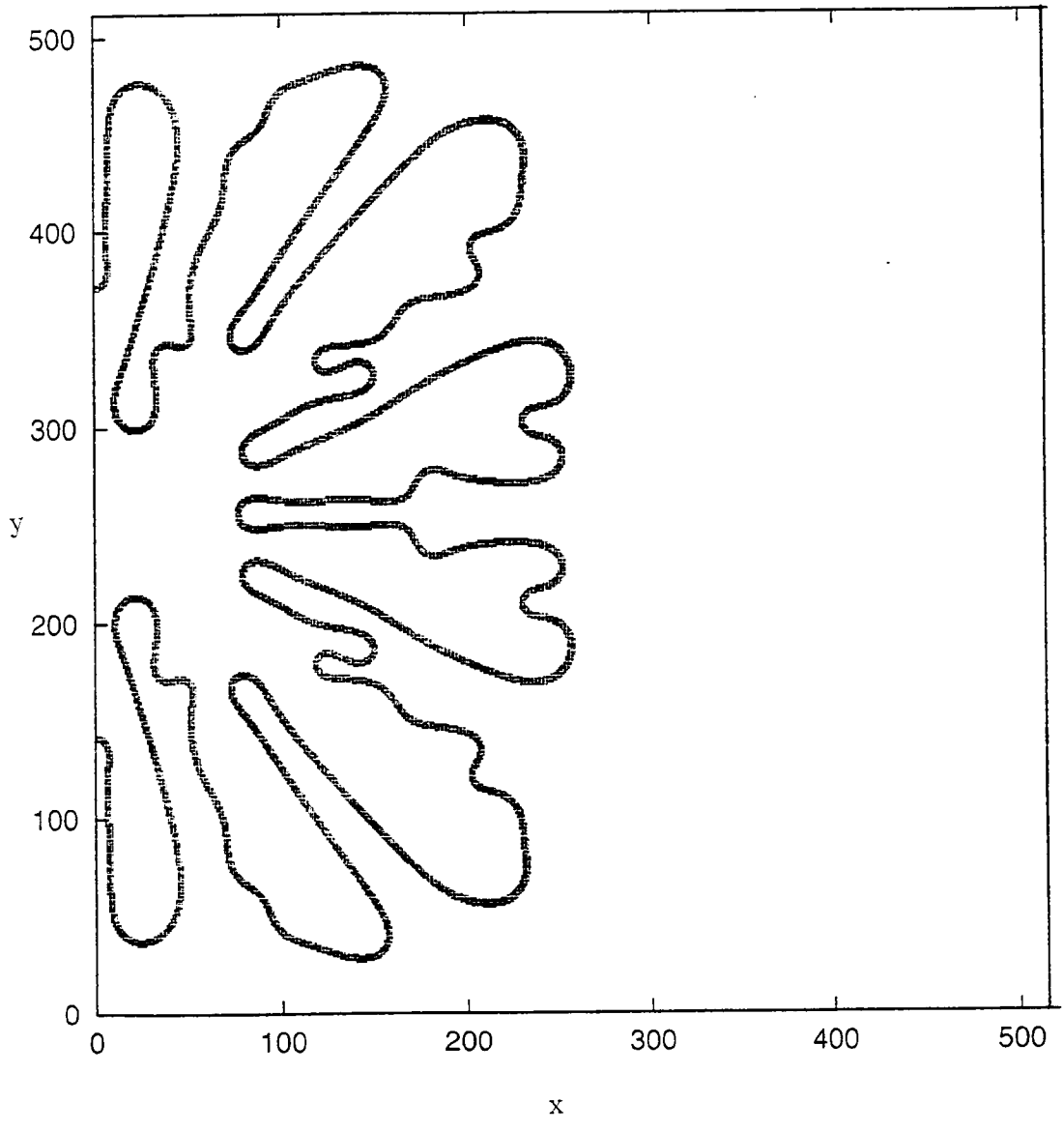

Figure 1: Dendritic growth without anisotropy ($\gamma = 0$).

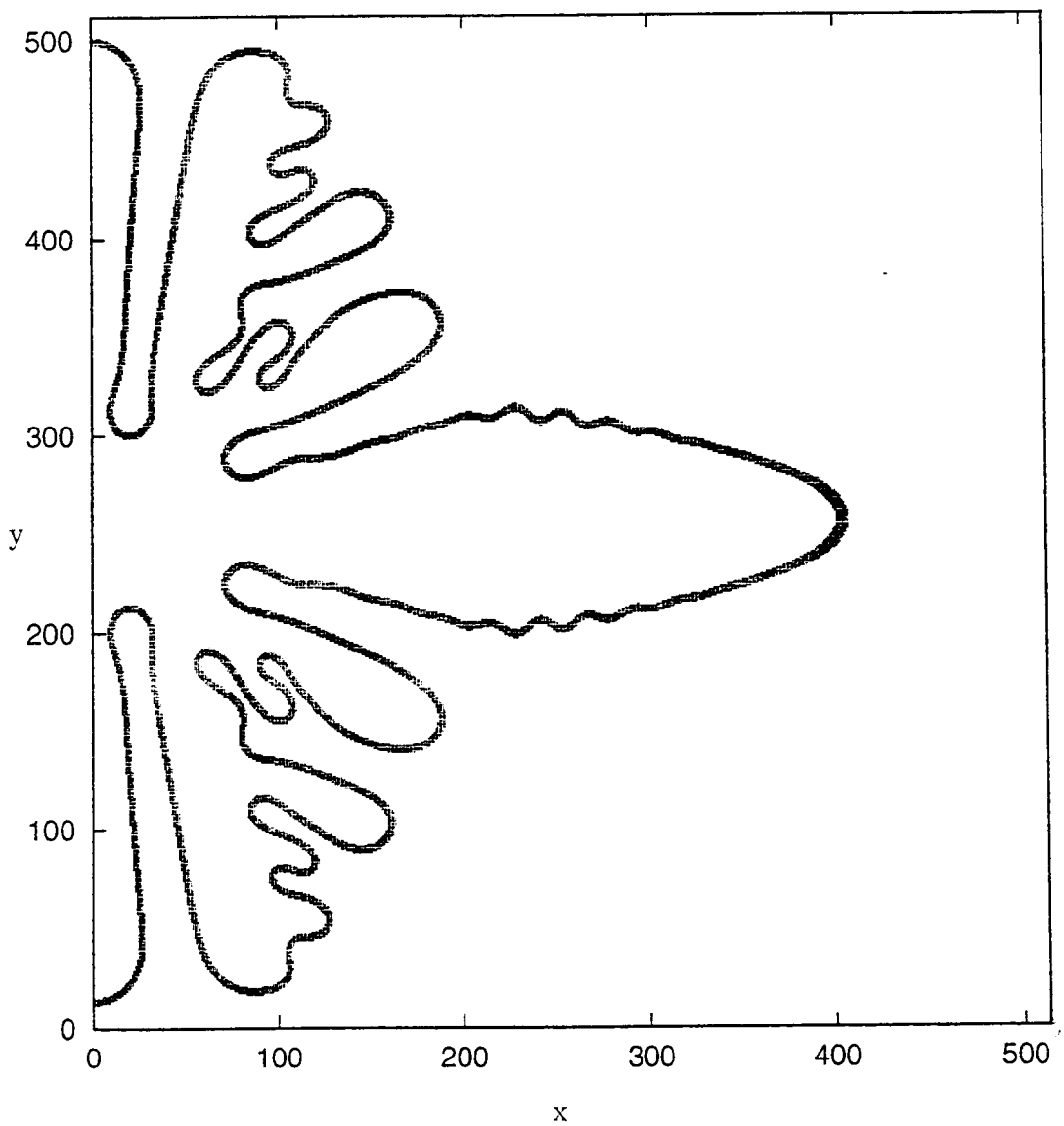

Figure 2: Dendritic growth with anisotropy strength $\gamma = 0.01$.

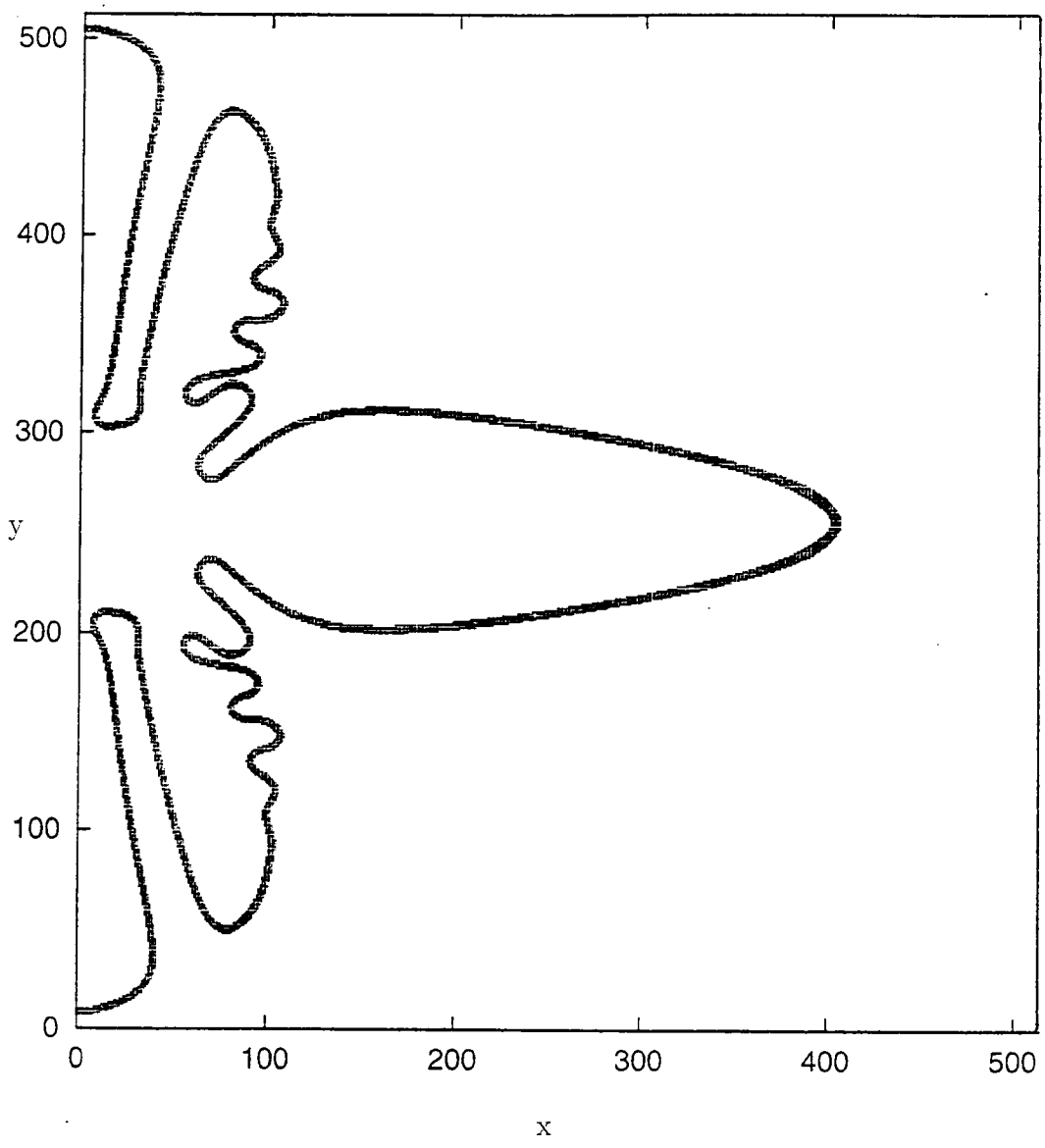

Figure 3: Dendritic growth with anisotropy strength $\gamma = 0.018$.

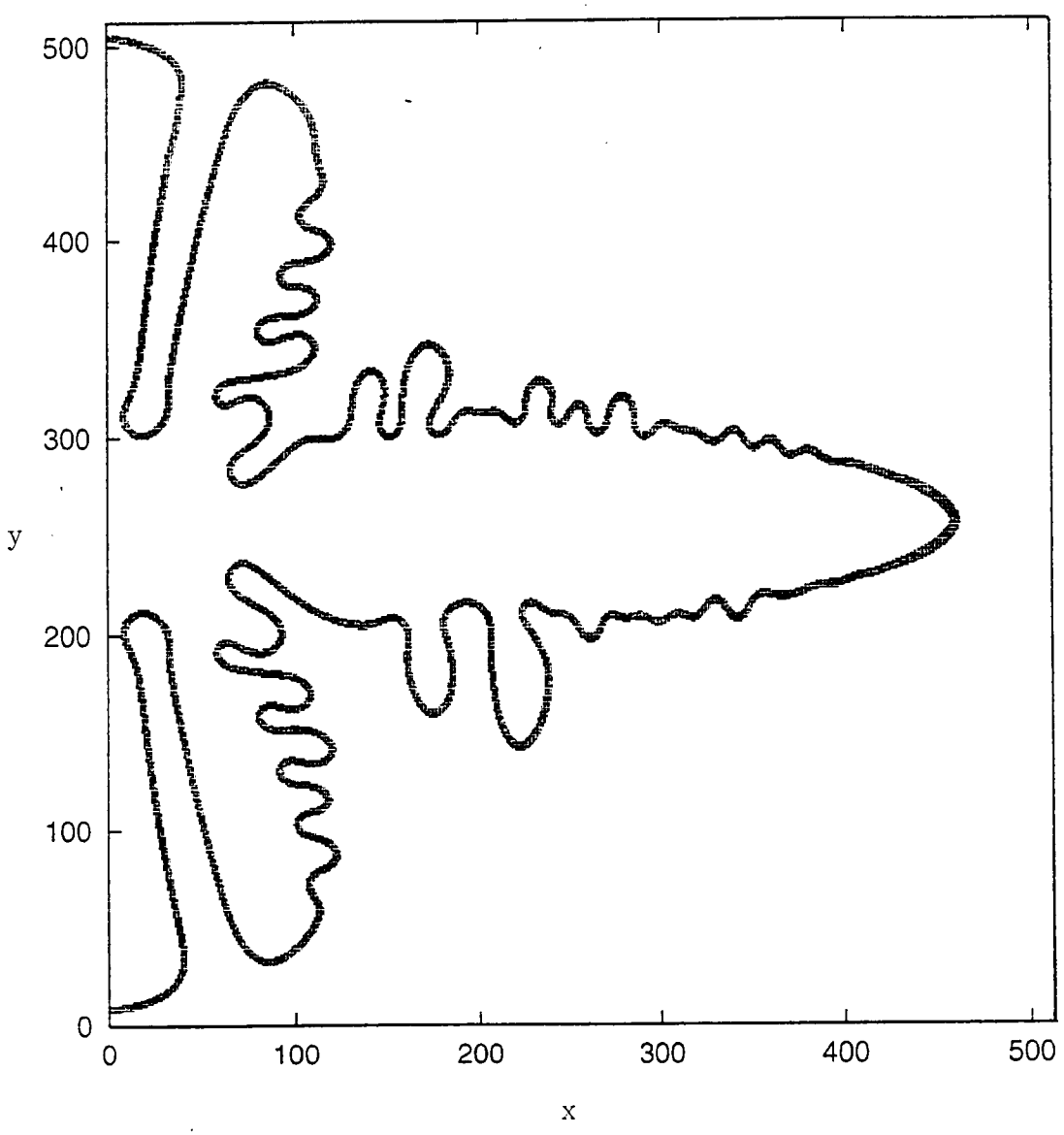

Figure 4: Dendritic growth with noise ($A = 0.018$) and anisotropy strength $\gamma = 0.018$.

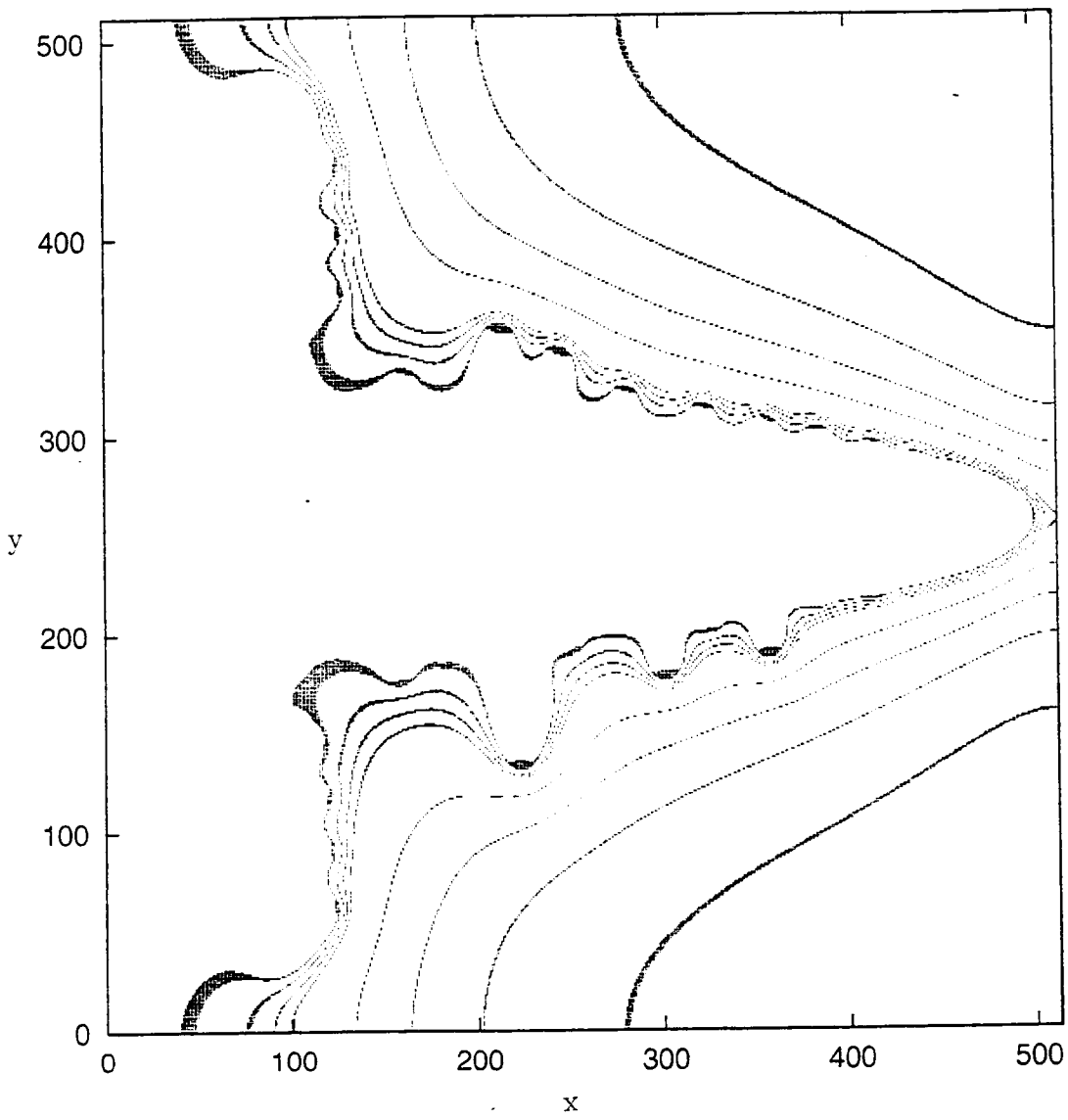

Figure 5: Temperature isotherms around the dendrite of Fig.4. The isotherms shown are $u = -0.01, -0.04, -0.07, -0.1, -0.3, -0.5, -0.7$. from left to right.

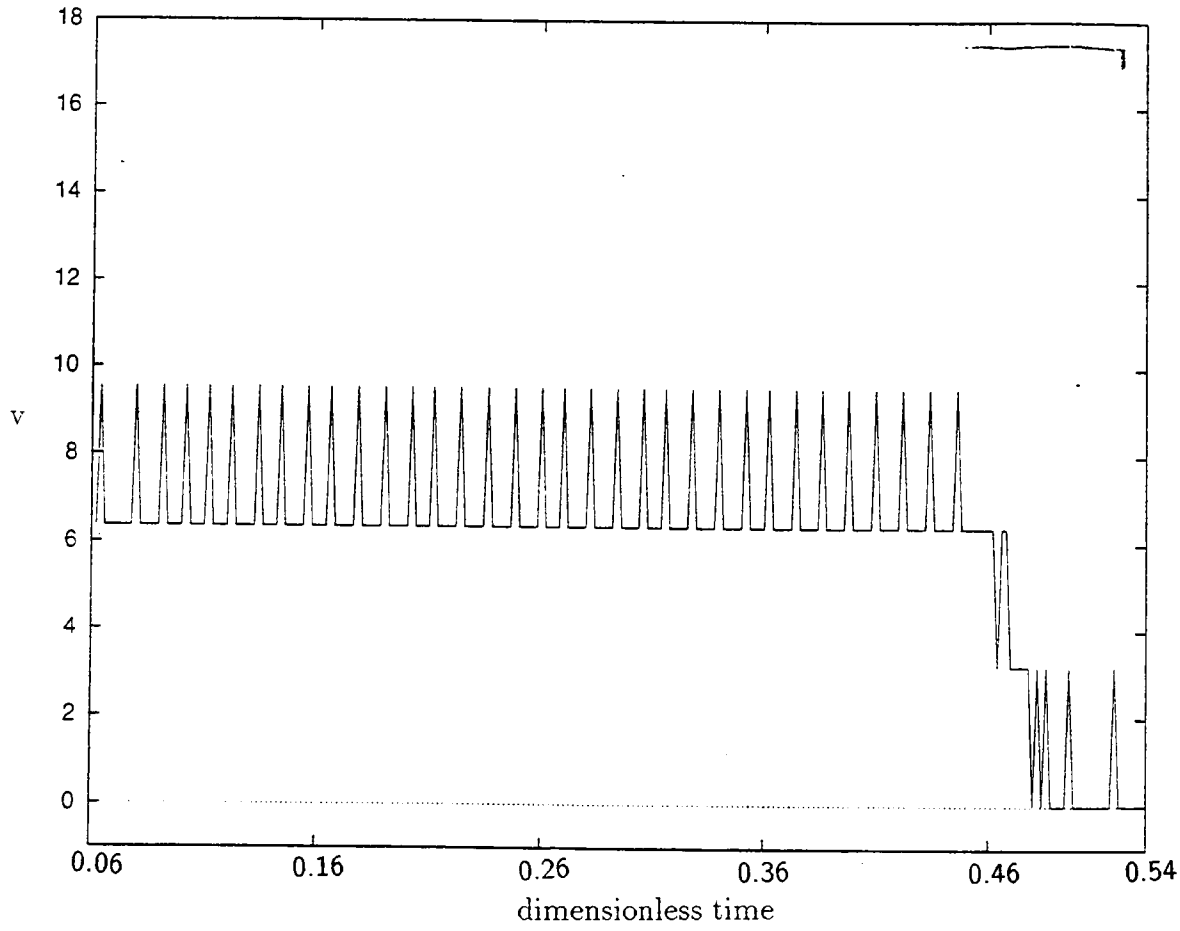

Figure 6 : Tip velocity (in x-direction) of the main finger of Fig.4, versus dimensionless time .

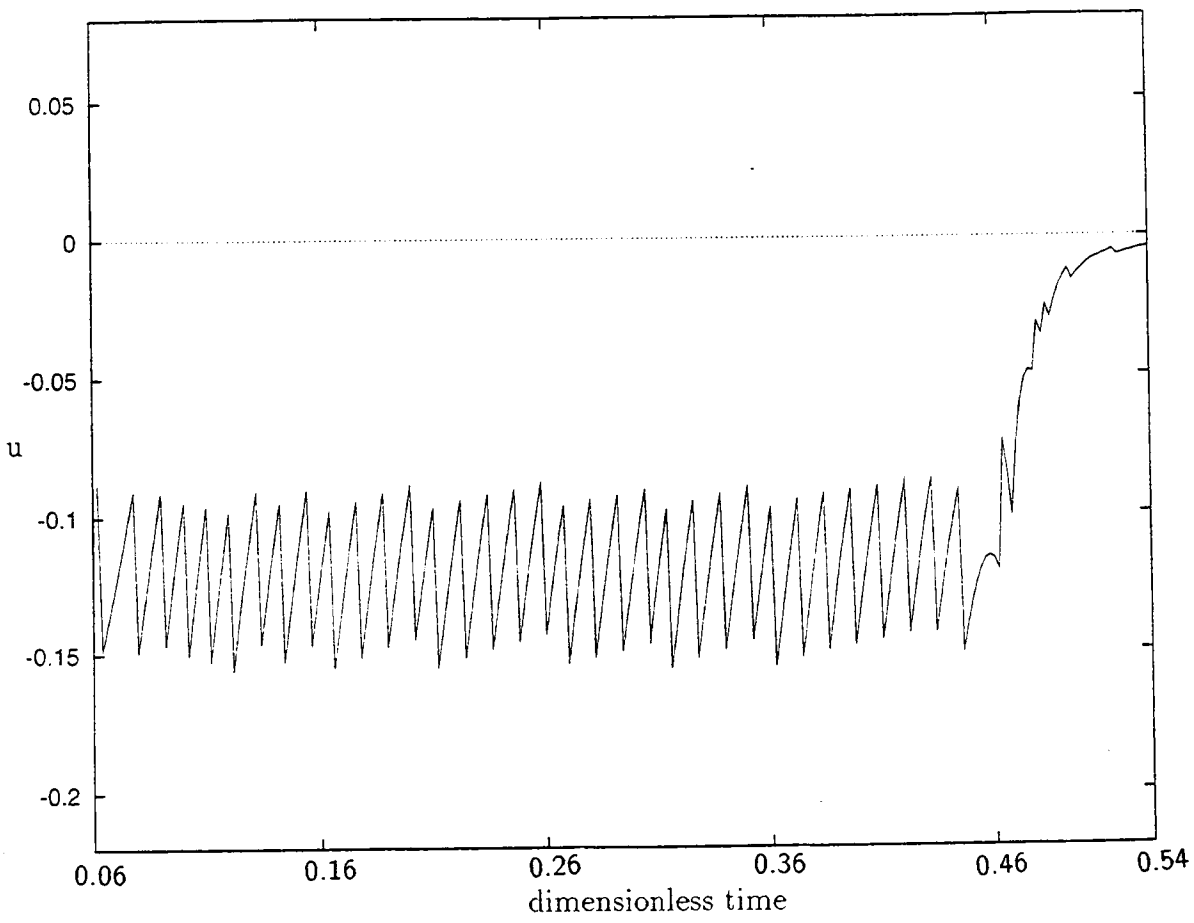

Figure 7 : Dimensionless temperature of the tip of the main finger of Fig.4 versus dimensionless time.

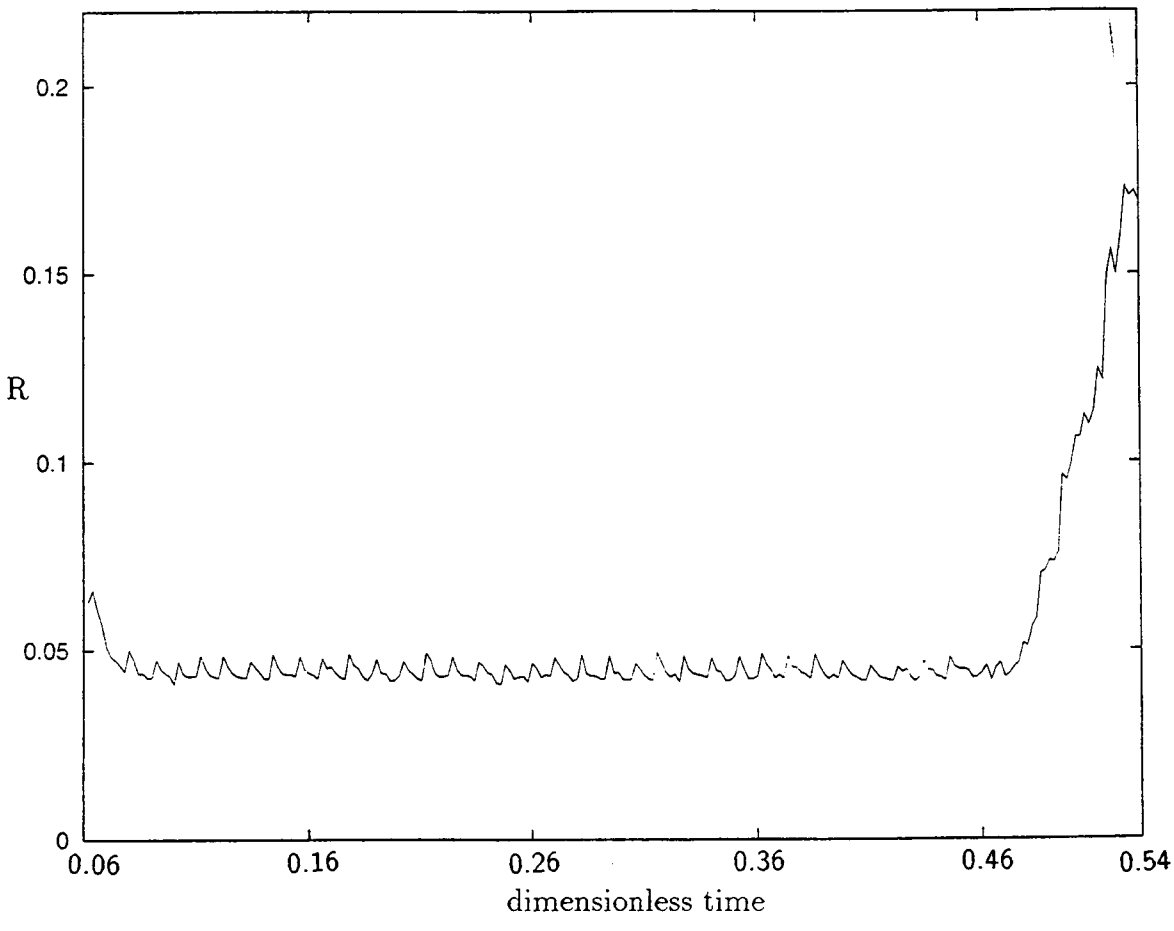

Figure 8 : Radius of curvature of the tip of the main finger of Fig.4 versus dimensionless time.

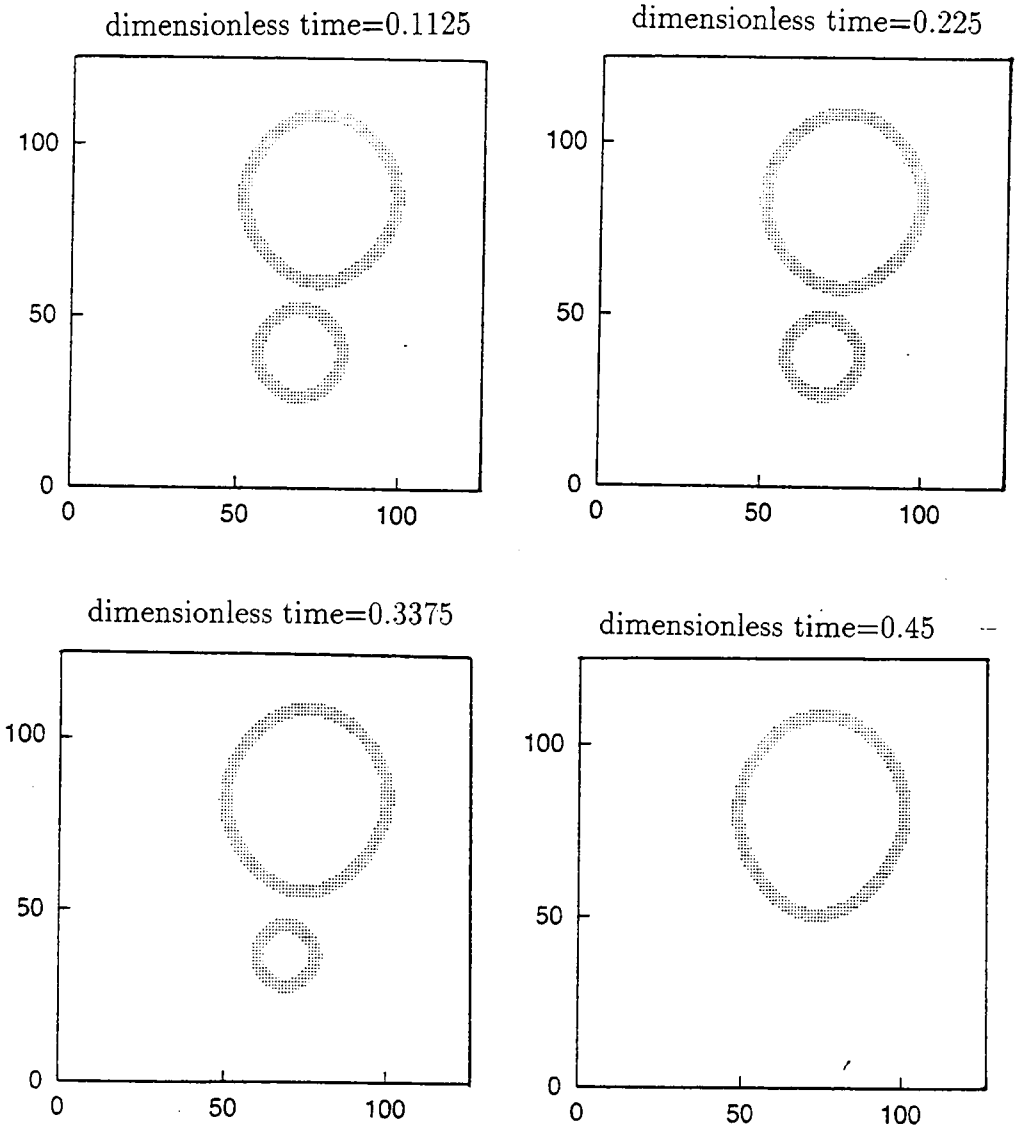

Figure 9 : Ostwald Ripening (non-isothermal). The bigger sphere grows at the expense of the smaller one.

Appendix B

```
c     Phase Field Model
c
c     Dendritic Growth in Solidification Process Using a Phase Field Method
c     The process simulated here is the solidification of a pure material
c

      implicit double precision(a-h,o-z)
      parameter(Mp=512)

      dimension x(0:Mp+1),y(0:Mp+1)

      dimension Ax(Mp+1,Mp),Ay(Mp,Mp+1),V(Mp,Mp)
      dimension U(0:Mp+1,0:Mp+1)
      dimension Ux(Mp+1,Mp), Uy(Mp,Mp+1)
      dimension P(0:Mp+1,0:Mp+1),Px(Mp+1,Mp),Py(Mp,Mp+1)
      dimension PyAx(Mp+1,Mp), PxAy(Mp,Mp+1)
      dimension etaAx(Mp+1,Mp), etaAy(Mp,Mp+1)
      dimension etapAx(Mp+1,Mp), etapAy(Mp,Mp+1)
      dimension Msk(0:Mp+5,0:Mp+5)
      dimension sd(17), tipv(800), tipu(800), tipr(800)
c     U:normalized temperature Tmelt:melting temp  Tref:reference temp
c     P:order parameter (phase field), P(solid)=0 & P(liquid)=1
c     theta:angle between the normal vector and the x-axis
c
c     For P and U (if applicable):
c     subscripts x, y, and t are partial derivatives w.r.t. x, y, and time
c
c     For eta:
c     the subscript p is the 1st derivtive w.r.t. theta
c
c     Flux/quantity w/ endings Ax and Ay is the flux/quantity at that face
c     V: control volume
c     Msk: matrix for masking
c     sd: seeds of random # generation in randfib
c     li,lj: counters for sd, randfib seeds
c     krdmd: rand# generated with randmod (also as seed)

        li=17
        lj=5
      tnxtm=0.03
      nt=1
      itip=0
      jtip=0
c       an interger seed from 1 to 2**31
        read(98,*) krdmd
        call sdgenfi(krdmd,sd)

c       input another seed
        read(98,*) krdmd

      mx=512
      my=512
      dxo=0.00375
      dyo=0.00375
c     Tmelt=1728.


c     parameters
c       k=4
c       rM=0.05
c       alpha=400
c       ep=0.005
c       s=c*(Tmelt-Temp)/rL

      Nmax=350000
```

```
      tmax=0.54
      dtout=tmax/1.
c     stability condition
c      stab=0.8
c      dt=stab*dx*dx*0.25/rM for phi
      pi=acos(-1.0)

c     initialization
      call mesh(mx,my,dxo,dyo, x,y,Ax,Ay,V)
      time=0.0
      Nstep=0
      tout=dtout
      call start(mx,my, Msk,U,P)
      call boudary(mx,my, Px,Py,Ux,Uy)
      call output(0,mx,my,P,U)

      nmesh=1
      Nsup=1
      rnu=0.
      dtm=0.0000005
      do i=1,3
         dtm=dtm*2.
        do j=1,12-i*i
        call flux(mx,my,Msk,1,1,1,gamma,U,P,x,y,Ux,Uy,Px,Py,
     *            PxAy,PyAx,etaAx,etaAy,etapAx,etapAy)
        call pde(mx,my,Msk,1,1,1,Px,Py,Ux,Uy,Ax,Ay,V,U,P,
     *  dtm,dtm,PxAy,PyAx,etaAx,etaAy,etapAx,etapAy,krdmd,li,lj,sd)
        time=time+dtm
        Nstep=Nstep+1
        enddo
      enddo

      dtm=0.000002

c     begin time stepping

      nfile=1
c      input Nsup for super-time-step
      read(98,*) Nsup
c       Nsup=3
c      input Nu for super-time-step
      read(98,*) rnu
c       rnu=0.05
c      Anisotropy
      read(98,*) gamma


      Nkup=3
100   continue
      if (Nstep .gt. Nmax) goto 990
      do kup=1,Nkup
c      super-time-step
      atau=0.
      do n=Nsup,1,-1
        tau=dtm/((rnu-1)*cos(pi*(2*n-1)/(2*Nsup))+1+rnu)
        atau=atau+tau
        time=time+tau
        Nstep=Nstep+1
        call flux(mx,my,Msk,n,Nsup,kup,gamma,U,P,x,y,Ux,Uy,Px,Py,
     *            PxAy,PyAx,etaAx,etaAy,etapAx,etapAy)
        call pde(mx,my,Msk,n,Nsup,kup,Px,Py,Ux,Uy,Ax,Ay,V,U,P,
     *  tau,atau,PxAy,PyAx,etaAx,etaAy,etapAx,etapAy,krdmd,li,lj,sd)
      enddo
      enddo
      if (time .gt. 0.009) then
       if (nmesh .eq. 1) then
```

```
            call adaptmesh(mx,my, P,U,dxo,dyo,dtm)
            call mesh(mx,my,dxo,dyo, x,y,Ax,Ay,V)
            call boudary(mx,my, Px,Py,Ux,Uy)
            call mask(mx,my,Msk,P)
            nmesh=nmesh+1
          endif
          if (time .ge. tnxtm) call tipm(mx,my,tnxtm,time,atau,P,
     *                          U,itip,jtip,tipv,tipu,tipr,nt,dxo,dyo)
         endif
         if (time .ge. tmax) goto 990
         if (time .gt. tout) then
            if (nfile .le. 0) call output(nfile,mx,my,P,U)
            tout=tout+dtout
            nfile=nfile+1
         endif
         goto 100

 990    continue
c           'time has reached or gone beyond tmax'

         call output(nfile,mx,my,P,U)
         call outtip(tipv,tipu,tipr)
         end


ccccccccccccccccccccccccc
c     subroutine start
ccccccccccccccccccccccccc

         subroutine start(mx,my, Msk,U,P)

         implicit double precision(a-h,o-z)
         dimension U(0:mx+1,0:my+1),P(0:mx+1,0:my+1)
         dimension Msk(0:mx+5,0:my+5)

c     initial temperature and values of order parameter

         do j=0,my+1
            do i=0,mx+1
c             initial temperature
              U(i,j)=-1.
c             undimensionalize initial temperature
c              U(i,j)=(Temp(i,j)-Tmelt)/(Tmelt-Tref)

c             initial values of order parameter
              P(i,j)=1.
              Msk(i,j)=0
            enddo
         enddo
c     intial value of order parameter about (0,0), that is,
c     a seed is planted there at time=0
         do j=220,290
           do i=0,50
             if (i*i+4*(j-256.5)**2 .lt. 1600.) then
               P(i,j)=0.
               U(i,j)=0.
             endif
           enddo
         enddo
c        initial mask
         do j=100,412
            do i=0,200
               Msk(i,j)=1
            enddo
         enddo
         return
```

```
      end

ccccccccccccccccccccccccccccccccccccccccc
c     subroutine boudary(mx,my, Px,Py,Ux,Uy)
ccccccccccccccccccccccccccccccccccccccccc

      subroutine boudary(mx,my, Px,Py,Ux,Uy)

      implicit double precision(a-h,o-z)
      dimension Ux(mx+1,my), Px(mx+1,my)
      dimension Uy(mx,my+1), Py(mx,my+1)

c     Boundary conditions
      do j=1,my
         Ux(1,j)=0.
         Px(1,j)=0.
         Ux(mx+1,j)=0.
         Px(mx+1,j)=0.
      enddo
      do i=1,mx
         Uy(i,my+1)=0.
         Py(i,my+1)=0.
         Uy(i,1)=0.
         Py(i,1)=0.
      enddo
      return
      end

ccccccccccccccccccccccc
c     subroutine mesh
ccccccccccccccccccccccc

      subroutine mesh(mx,my,dx,dy, x,y,Ax,Ay,V)

      implicit double precision (a-h,o-z)
      dimension x(0:mx+1), y(0:my+1), xf(513), yf(513)
      dimension Ax(mx+1,my),Ay(mx,my+1),V(mx,my)

c     x:node  xf:face  mx: # of nodes in x direction
c     y:      yf       my                    y
c     Ax(i,j):area of face at xf(i),y(j)
c     Ay(i,j):area of face at x(i),yf(j)
c     V(i,j): i,j-control volume

      x(0)=0.
      y(0)=0.
      do i=1,mx
         x(i)=(i-0.5)*dx

      enddo
      do j=1,my
         y(j)=(j-0.5)*dy

      enddo
      x(mx+1)=x(mx)+0.5*dx
      y(my+1)=y(my)+0.5*dy

      do i=2,mx
         xf(i)=0.5*(x(i-1)+x(i))
      enddo
      do j=2,my
         yf(j)=0.5*(y(j-1)+y(j))

      enddo
      xf(1)=x(0)
      xf(mx+1)=x(mx+1)
```

```
      yf(1)=y(0)
      yf(my+1)=y(my+1)


      do j=1,my
         do i=1,mx+1
            Ax(i,j)=yf(j+1)-yf(j)
         enddo
      enddo
      do j=1,my+1
         do i=1,mx
            Ay(i,j)=xf(i+1)-xf(i)
         enddo
      enddo
      do j=1,my
        do i=1,mx
           V(i,j)=Ay(i,j)*(yf(j+1)-yf(j))
        enddo
      enddo
      return
      end


ccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c     subroutine adaptmesh(mx,my, P,U,dxo,dyo,dtm)
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

      subroutine adaptmesh(mx,my, P,U,dxo,dyo,dtm)

      implicit double precision (a-h,o-z)
      dimension P(0:mx+1,0:my+1), U(0:mx+1,0:my+1)

        do j=2,my,2
         do i=2,mx,2
           P(i,j)=(P(i-1,j-1)+P(i-1,j)+P(i,j-1)+P(i,j))*0.25
           U(i,j)=(U(i-1,j-1)+U(i-1,j)+U(i,j-1)+U(i,j))*0.25
         enddo
        enddo
        do j=2,my,2
           jj=j/2
           do i=2,mx,2
              ii=i/2
              P(ii,jj)=P(i,j)
              U(ii,jj)=U(i,j)
           enddo
        enddo
        do j=my/2,1,-1
          do i=1,mx/2
             P(i,j+my/4)=P(i,j)
             U(i,j+my/4)=U(i,j)
          enddo
        enddo
        do j=1,my/4
          do i=1,mx/2
            P(i,j)=1.
            U(i,j)=-1.
          enddo
        enddo
        do j=3*my/4+1,my
          do i=1,mx/2
             P(i,j)=1.
             U(i,j)=-1.
          enddo
        enddo
        do j=1,my
          do i=mx/2+1,mx
```

```
                P(i,j)=1.
                U(i,j)=-1.
              enddo
            enddo

            dxo=dxo*2.
            dyo=dyo*2.
            dtm=dtm*4.
            return
            end

ccccccccccccccccccccccccc
c     subroutine flux
ccccccccccccccccccccccccc

      subroutine flux(mx,my,Msk,n,Nsup,kup,gamma,U,P,x,y,Ux,Uy,Px,Py,
     *                PxAy,PyAx,etaAx,etaAy,etapAx,etapAy)

      implicit double precision (a-h,o-z)
      dimension P(0:mx+1,0:my+1), Px(mx+1,my), Py(mx,my+1)
      dimension U(0:mx+1,0:my+1)
      dimension Ux(mx+1,my), Uy(mx,my+1)
      dimension x(0:mx+1), y(0:my+1)
      dimension PyAx(mx+1,my), PxAy(mx,my+1)
      dimension etaAx(mx+1,my), etaAy(mx,my+1)
      dimension etapAx(mx+1,my), etapAy(mx,my+1)
      dimension Msk(0:mx+5,0:my+5)


      do j=1,my
        do i=2,mx
        dx=x(i)-x(i-1)
           Ux(i,j)=(U(i,j)-U(i-1,j))/dx
        enddo
      enddo
      do j=2,my
         dy=y(j)-y(j-1)
         do i=1,mx
            Uy(i,j)=(U(i,j)-U(i,j-1))/dy
         enddo
      enddo

      if ((n .eq. Nsup) .or. (kup .eq. 1)) then
      do j=1,my
        do i=2,mx
        dx=x(i)-x(i-1)
          if (Msk(i,j) .eq. 1) then
           Px(i,j)=(P(i,j)-P(i-1,j))/dx
          endif
        enddo
      enddo
      do j=2,my
         dy=y(j)-y(j-1)
         do i=1,mx
          if (Msk(i,j) .eq. 1) Py(i,j)=(P(i,j)-P(i,j-1))/dy
         enddo
      enddo

      do j=2,my-1
         dy=y(j+1)-y(j-1)
       do i=2,mx
         if (Msk(i,j) .eq. 1) then
          PyAx(i,j)=P(i-1,j+1)-P(i-1,j-1)+P(i,j+1)-P(i,j-1)
          PyAx(i,j)=0.5*PyAx(i,j)/dy
         endif
       enddo
```

```
      PyAx(1,j)=(P(1,j+1)-P(1,j-1))/dy
      PyAx(mx+1,j)=(P(mx,j+1)-P(mx,j-1))/dy
    enddo
    dy=y(2)-y(1)
    do i=2,mx
      PyAx(i,1)=.5*(P(i-1,2)-P(i-1,1)+P(i,2)-P(i,1))/dy
    enddo
    PyAx(1,1)=(P(1,2)-P(1,1))/dy
    PyAx(mx+1,1)=(P(mx,2)-P(mx,1))/dy
    dy=y(my)-y(my-1)
    do i=2,mx
      PyAx(i,my)=P(i-1,my)-P(i-1,my-1)+P(i,my)-P(i,my-1)
      PyAx(i,my)=0.5*PyAx(i,my)/dy
    enddo
    PyAx(1,my)=(P(1,my)-P(1,my-1))/dy
    PyAx(mx+1,my)=(P(mx,my)-P(mx,my-1))/dy

    do j=2,my
     do i=2,mx-1
      dx=x(i+1)-x(i-1)
      if (Msk(i,j) .eq. 1) then
       PxAy(i,j)=P(i+1,j-1)-P(i-1,j-1)+P(i+1,j)-P(i-1,j)
       PxAy(i,j)=0.5*PxAy(i,j)/dx
      endif
     enddo
    enddo
    do i=2,mx-1
      dx=x(i+1)-x(i-1)
      PxAy(i,1)=(P(i+1,1)-P(i-1,1))/dx
      PxAy(i,my+1)=(P(i+1,my)-P(i-1,my))/dx
    enddo
    dx=x(2)-x(1)
    do j=2,my
      PxAy(1,j)=.5*(P(2,j-1)-P(1,j-1)+P(2,j)-P(1,j))/dx
    enddo
    PxAy(1,1)=(P(2,1)-P(1,1))/dx
    PxAy(1,my+1)=(P(2,my)-P(1,my))/dx
    dx=x(mx)-x(mx-1)
    do j=2,my
      PxAy(mx,j)=P(mx,j-1)-P(mx-1,j-1)+P(mx,j)-P(mx-1,j)
      PxAy(mx,j)=0.5*PxAy(mx,j)/dx
    enddo
    PxAy(mx,1)=(P(mx,1)-P(mx-1,1))/dx
    PxAy(mx,my+1)=(P(mx,my)-P(mx-1,my))/dx

c   Definitions of scAx scAy ssfAx ssfAy
c   scAx and scAy=sin(theta)*cos(theta)
c   ssfAx and ssfAy=cos(theta)**2-sin(theta)**2

c   for k=4
    do j=1,my
      do i=1,mx+1
       if (Msk(i,j) .eq. 1) then
        sqPyAx=PyAx(i,j)**2
        denAx=Px(i,j)*Px(i,j)+sqPyAx
        if (denAx .ne. 0.) then
         scAx=Px(i,j)*PyAx(i,j)/denAx
         ssfAx=1.-2.*sqPyAx/denAx
        else
         scAx=0.
         ssfAx=0.
        endif
        etaAx(i,j)=1.+gamma*(ssfAx*ssfAx-4.*scAx*scAx)
        etapAx(i,j)=-16.*gamma*scAx*ssfAx
       endif
      enddo
```

```
      enddo

      do j=1,my+1
        do i=1,mx
         if (Msk(i,j) .eq. 1) then
          sqPy=Py(i,j)**2
          denAy=PxAy(i,j)*PxAy(i,j)+sqPy

          if (denAy .ne. 0.) then
           scAy=PxAy(i,j)*Py(i,j)/denAy
           ssfAy=1.-2.*sqPy/denAy
          else
           scAy=0.
           ssfAy=0.
          endif
          etaAy(i,j)=1.+gamma*(ssfAy*ssfAy-4.*scAy*scAy)
          etapAy(i,j)=-16.*gamma*scAy*ssfAy
         endif
        enddo
      enddo
      endif
      return
      end

ccccccccccccccccccccccccc
c     subroutine pde
ccccccccccccccccccccccccc

      subroutine pde(mx,my,Msk,n,Nsup,kup,Px,Py,Ux,Uy,Ax,Ay,V,U,P,
     *   dt,atau,PxAy,PyAx,etaAx,etaAy,etapAx,etapAy,krdmd,li,lj,sd)
      implicit double precision(a-h,o-z)

      dimension U(0:mx+1,0:my+1),P(0:mx+1,0:my+1)
      dimension Ux(mx+1,my), Uy(mx,my+1)
      dimension Ut(0:513,0:513), Pt(0:513,0:513)
      dimension Px(mx+1,my),Py(mx,my+1)
      dimension Ax(mx+1,my), Ay(mx,my+1), V(mx,my)
      dimension PyAx(mx+1,my), PxAy(mx,my+1)
      dimension etaAx(mx+1,my), etaAy(mx,my+1)
      dimension etapAx(mx+1,my), etapAy(mx,my+1)
      dimension Msk(0:mx+5,0:my+5)
        dimension sd(17)

c     parameters
c       k=4
      rM=0.05
      al=400.
      ep=.005
      s=0.5
      cnst=30.*ep*al*s

c     amplitude of random noise
        A=0.018
c     the first of the coupled pde's
      if ((n .eq. Nsup) .or. (kup .eq. 1)) then
      do j=1,my
       do i=1,mx

c      g is a symmetric double well potential
c      gr is the square root of g
c      dum is dummy

       gr=P(i,j)*(1.-P(i,j))
       if (Msk(i,j) .eq. 1) then
       dum=(etaAx(i+1,j)**2)*Px(i+1,j)-(etaAx(i,j)**2)*Px(i,j)
       dum=dum-etaAx(i+1,j)*etapAx(i+1,j)*PyAx(i+1,j)
```

```
          Pt(i,j)=(dum+etaAx(i,j)*etapAx(i,j)*PyAx(i,j))*Ax(i,j)
          dum=etaAy(i,j+1)**2*Py(i,j+1)-etaAy(i,j)**2*Py(i,j)
          dum=dum-etaAy(i,j+1)*etapAy(i,j+1)*PxAy(i,j+1)
          dum=(dum-etaAy(i,j)*etapAy(i,j)*PxAy(i,j))*Ay(i,j)
          Pt(i,j)=(Pt(i,j)+dum)/V(i,j)
          endif
c         AR=AR(x,y,randuni) is a random number
          call randuni(krdmd,li,lj,sd,rduni)
          AR=A*(rduni-0.5)
          dum=AR+P(i,j)-0.5+cnst*U(i,j)*gr
          Pt(i,j)=(Pt(i,j)+gr*dum/(ep*ep))*rM
          enddo
         enddo
         endif

c     the second of the coupled pde's
         do j=1,my
          do i=1,mx
           gr=P(i,j)*(1.-P(i,j))
          Ut(i,j)=Ax(i,j)*(Ux(i+1,j)-Ux(i,j))+Ay(i,j)*(Uy(i,j+1)-Uy(i,j))
          Ut(i,j)=Ut(i,j)/V(i,j)-30.*gr*gr*Pt(i,j)/s
          enddo
         enddo

c     update U(i,j) and P(i,j)
         do j=1,my
           do i=1,mx
             U(i,j)=U(i,j)+(Ut(i,j)*dt)
           enddo
         enddo

         if (kup .eq. 1) then
          do j=1,my
            do i=1,mx
              P(i,j)=P(i,j)+Pt(i,j)*dt
              if (P(i,j) .gt. 1.0) P(i,j)=1.0
              if (P(i,j) .lt. 0.0) P(i,j)=0.0
            enddo
          enddo
          if (Nsup .ne. 1) call mask(mx,my,Msk,P)
         else
          if ((n .eq. 1) .and. (Nsup .ne. 1)) then
          do j=1,my
            do i=1,mx
              P(i,j)=P(i,j)+Pt(i,j)*atau
              if (P(i,j) .gt. 1.0) P(i,j)=1.0
              if (P(i,j) .lt. 0.0) P(i,j)=0.0
            enddo
          enddo
          call mask(mx,my,Msk,P)
         endif
         endif
         return
         end


ccccccccccccccccccccccc
c     subroutine output
ccccccccccccccccccccccc

         subroutine output(nfile,mx,my,P,U)

         implicit double precision (a-h,o-z)
         dimension P(0:mx+1,0:my+1), U(0:mx+1,0:my+1)

         open(35)
```

```
      open(40)
      do j=0,my+1
         do i=0,mx+1
            if ((P(i,j) .gt. 0.1) .and. (P(i,j) .lt. 0.9)) then
            write(35,*) i,'  ',j,'  ',P(i,j)
            endif
            write(40,*) U(i,j)
         enddo
      enddo
      close(40)
      close(35)
      return

      end

ccccccccccccccccccccccccccccccc
c     subroutine mask
ccccccccccccccccccccccccccccccc

      subroutine mask(mx,my,Msk,P)
      implicit double precision (a-h,o-z)
      dimension P(0:mx+1,0:my+1), Msk(0:mx+5,0:my+5)

      do j=1,my
        do i=1,mx
          Msk(i,j)=0
        enddo
      enddo

      do j=1,my
        do i=1,mx
          if ((P(i,j) .gt. 0.) .and. (P(i,j) .lt. 1.)) then
            do jj=j-4,j+5
              do ii=i-4,i+5
                if ((ii .ge. 0) .and. (jj .ge. 0)) Msk(ii,jj)=1
              enddo
            enddo
          endif
        enddo
      enddo
      return
      end

ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
      subroutine tipm(mx,my,tnxtm,time,atau,P,
     *       U,itip,jtip,tipv,tipu,tipr,nt,dxo,dyo)
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

      implicit double precision (a-h,o-z)
      dimension tipv(800), tipu(800), tipr(800)
      dimension P(0:mx+1,0:my+1), U(0:mx+1,my+1)
      if (itip .eq. 0) then
        call tippos(mx,my,P,itip)
        tnxtm=time+50*atau
      else
       tnxtm=time+50*atau
       call tippos(mx,my,P,jtip)
       tipv(nt)=real(jtip-itip)*dxo/(50*atau)
       tipu(nt)=U(jtip,my/2)
      tipr(nt)=(P(jtip,my/2+1)-2*P(jtip,my/2)+P(jtip,my/2-1))/(dyo*dyo)
       tipr(nt)=((2*dxo)/(P(jtip+1,my/2)-P(jtip-1,my/2)))/tipr(nt)
       nt=nt+1
       itip=jtip
      endif
      return
      end
```

```
ccccccccccccccccccccccccccccccccccccccccc
      subroutine tippos(mx,my,P,ktip)
ccccccccccccccccccccccccccccccccccccccccc

      implicit double precision (a-h,o-z)
      dimension P(0:mx+1,0:my+1)
      ktip=1
c     tip's y-coordinate at y=my/2
      do i=1,mx
       if ((P(i,my/2) .gt. .2) .and. (P(i,my/2) .lt. .8)) then
        ktip=max(ktip,i)
       endif
      enddo
      return
      end

ccccccccccccccccccccccccccccccccccccccccc
      subroutine outtip(tipv,tipu,tipr)
ccccccccccccccccccccccccccccccccccccccccc

      implicit double precision (a-h,o-z)
      dimension tipv(800), tipu(800), tipr(800)

      open(41)
      open(42)
      open(43)
      do j=1,800
        write(41,*) j,' ',tipv(j)
        write(42,*) j,' ',tipu(j)
        write(43,*) j,' ',tipr(j)
      enddo
      close(41)
      close(42)
      close(43)
      return
      end

c     subroutine to generate seeds for randfib
ccccccccccccccccccccccccccccccccccccccccc
      subroutine sdgenfi(krdmd,sd)
ccccccccccccccccccccccccccccccccccccccccc

        implicit double precision (a-h,o-z)
        dimension ns(24), sd(17)
        do i=1,17
          sd(i)=0.
          do j=1,24
            call randmod(krdmd,rdmd)
            if (rdmd .lt. 0.5) then
              ns(j)=0
            else
              ns(j)=1
            endif
            sd(i)=sd(i)+real(ns(j))/2**j
          enddo
        enddo
        return
        end

c   All Rand# generators below produce rand#s between 0 and 1
ccccccccccccccccccccccccccccccccccccccccc
      subroutine randmod(krdmd,rdmd)
ccccccccccccccccccccccccccccccccccccccccc

        implicit double precision (a-h,o-z)
```

```
        krdmd=mod(69069*krdmd,2**31)
        rdmd=real(krdmd)/2**31
        return
        end

ccccccccccccccccccccccccccccccccccccccc
      subroutine randfib(li,lj,sd,rdfb)
ccccccccccccccccccccccccccccccccccccccc

        implicit double precision (a-h,o-z)
        dimension sd(17)
        rdfb=sd(li)-sd(lj)
        if (rdfb .lt. 0.) rdfb=rdfb+1.
        sd(li)=rdfb
        li=li-1
        lj=lj-1
        if (li .eq. 0) li=17
        if (lj .eq. 0) lj=17
        return
        end

ccccccccccccccccccccccccccccccccccccccccccccccccccccccc
      subroutine randuni(krdmd,li,lj,sd,rduni)
ccccccccccccccccccccccccccccccccccccccccccccccccccccccc

        implicit double precision (a-h,o-z)
        dimension sd(17)
        call randmod(krdmd,rdmd)
        call randfib(li,lj,sd,rdfb)
        rduni=rdmd-rdfb
        if(rduni .lt. 0.) rduni=rduni+1.
        return
        end
```

Vita

Chun Wai Chang was born in Hong Kong. He completed his high school education there. He came to the United States as an immigrant and an international student. He studied at Edmonds Community College, Lynnwood, WA. Just before he transferred to University of Washington at Seattle, WA, he became a permanent resident of this country. He graduated with two Bachelor of Science degrees, one in Physics with distinction and one in Mathematics. He then came to University of Tennessee at Knoxville for his graduate study. He became a citizen of the United States during his stay in Knoxville. He graduated from University of Tennessee with a Master of Science degree in Physics.