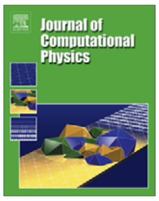
Journal of Computational Physics 228 (2009) 8945–8957

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp



Efficient thermal field computation in phase-field models

Jing-Rebecca Li [a,*], Donna Calhoun [b], Lucien Brush [c]

[a] *Projet POEMS, INRIA, Domaine de Voluceau – Rocquencourt – B.P. 105, 78153 Le Chesnay Cedex, France*
[b] *DEN/DM2S/SFME/LTMF, Commissariat a l'Energie Atomique, F-91191 Gif-sur-Yvette Cedex, France*
[c] *Department of Materials Science and Engineering, University of Washington Seattle, WA 98195, United States*

ARTICLE INFO

Article history:
Received 20 February 2008
Received in revised form 21 April 2009
Accepted 6 August 2009
Available online 1 September 2009

Keywords:
Phase-field
Crystal growth
Dendritic solidification
Diffusion equation
Unbounded domain
Integral representation
Fast solvers

ABSTRACT

We solve the phase-field equations in two dimensions to simulate crystal growth in the low undercooling regime. The novelty is the use of a fast solver for the free space heat equation to compute the thermal field. This solver is based on the efficient direct evaluation of the integral representation of the solution to the constant coefficient, free space heat equation with a smooth source term. The computational cost and memory requirements of the new solver are reasonable and no artificial boundary conditions are needed. This allows one to solve for the thermal field in a computational domain whose size depends only on the size of the growing crystal and not on the extent of the thermal field, which can result in significant computational savings in the low undercooling regime.

© 2009 Elsevier Inc. All rights reserved.

1. Introduction

Freely growing single crystals exhibit a wide range of crystal-melt interface morphologies, including faceted and dendritic structures, and as a result, have been of great interest to the engineering, scientific, and mathematical communities for decades. Researchers have attempted to understand, predict, and control solidification morphologies which ultimately impact material properties and performance [27]. Crystal growth morphologies have also been a subject of interest to researchers studying nonlinear dynamical systems because freely growing crystals are examples of rich, pattern-forming systems [29,10].

Numerical simulation of dendritic growth has become an important tool in understanding how crystal morphologies evolve in various experimental and theoretical regimes. It has attracted the attention of computational scientists motivated to develop and improve numerical methods for accurately reproducing the two- and three-dimensional tree-like structures that emerge when crystals form in an undercooled melt. As a result, several effective methods are now available, including those based on solving the physically derived phase-field equations [14,5,35,47], as well as level set and Lagrangian methods which can be used to solve the related Stefan problem [43,6,32]. For an overview of phase-field modeling and contemporary views of crystal growth, see the review papers by Karma [24], Provatas et al. [39], Boettinger et al. [2] and Glicksman and Lupulescu [16].

Regimes of theoretical and experimental importance most easily accessible by numerical computations are those for which the temperature of the liquid melt is relatively far from the equilibrium melting temperature. In such regimes, the non-dimensional undercooling parameter, the Stefan number, is large (typically $O(1)$) and crystals grow fast relative to

* Corresponding author. Tel.: +33 1 39 63 53 55; fax: +33 1 39 63 58 84.
E-mail address: jingrebecca.li@inria.fr (J.-R. Li).

the expansion of the thermal field. One can observe interesting results before the extent of the thermal field exceeds the computational domain and adequate resolution of the solid–liquid interface can be achieved at modest computational expense. Many of the early dendritic growth calculations reported in the literature were computed in this regime [46,45,3,34,25,48,33]. In another regime of theoretical importance, one assumes an infinite thermal decay rate and models the heat transport using a free-space Laplace equation. In this case, the computational mesh for the thermal field can be eliminated entirely by using front-tracking methods coupled with boundary integral techniques to directly solve for growth velocities along the solid–liquid interface [4,7,8].

By contrast, obtaining accurate numerical solutions in the regime of low undercoolings still proves to be computationally challenging. For low Stefan numbers (see Eq. (8)) crystals grow very slowly relative to the rate of expansion of the thermal front and the extent of the thermal field in the liquid can quickly exceed by several orders of magnitude the size of the growing crystal. To obtain results which can be used to confirm existing theory or be validated by experiments, long simulation times may be required. In the free growth experiments of Koss et al. carried out in NASA's USMP-4 (Ref. [18] in Provatas et al. [38]), pivalic acid was grown at a variety of undercoolings. The lowest undercooling corresponded to a melt temperature of approximately 0.5 K below the melting point, or to a Stefan number of about 0.05. Under these conditions, it can take a long time to establish steady state dendrite tip growth conditions, which has often been the goal of many experimentalists trying to verify theories of dendritic growth.

In addition, the length scale representative of the solid–liquid interface thickness is on the order of a few nanometers at most, for the primary, secondary and tertiary dendritic branches it can be up to microns and for the extent of the thermal field, centimeters. A numerical simulation which attempts to reproduce these experimental results would need to correctly compute the thermal field over long simulation times, while adequately resolving the interface thickness as well as the emerging dendritic features.

Typically, in numerical simulations using finite difference or finite elements methods, the computational domain is taken to be large enough to contain the entire thermal field. This is due to the fact that in order to match experiments, numerical simulations must not exhibit any spurious artifacts introduced by the presence of an artificial computational boundary. During the early stages of actual crystal growth, the experimental container is typically large relative to the size of the crystal. For an initially uniform undercooled melt, disturbances in the thermal field due to the growth of a crystal are damped at distances as far as the container boundary, and the thermal field is fully captured by the container. The experimental container is assumed to have negligible, if any, effect on the evolving solidification front. In fact, this is a critical requirement for experimentalists attempting to validate the steady state theories of dendrite tip velocities and tip curvatures.

For slowly growing crystals, long simulation times are required before interesting results can be observed, and as a result, the thermal field will exhibit time dependent behavior far from the crystal interface. Capturing the thermal field numerically places significant demands on computational resources.

1.1. Numerical approaches

In this paper, we focus on the efficient computation of the thermal field during free crystal growth in the low undercooling regime. The lowest undercooling considered in the numerical results section is $S = 0.025$.

A vast majority of studies on the numerical solution of the phase-field model use finite-difference or finite element schemes, which are relatively easy to implement. Such methods require artificial boundary conditions which must be imposed on the thermal field at the boundary of the computational domain.

The boundary conditions that are almost universally applied because they are easy to implement are purely Neumann conditions (see [34,3,42]). This choice of artificial boundary conditions will result in a distorted thermal field at long times and hence an incorrect solution to the free space crystal growth problem.

To reduce the adverse effects caused by imposing inappropriate artificial boundary conditions on finite computational domain, one typically increases the size of the computational domain and introduces mesh adaptivity. In [41,40] an adaptive mesh refinement (AMR) technique was used to successfully model two-dimensional free growth at the low undercoolings described in the pivalic acid experiment above.

A different approach was taken in [36] where an adaptive Monte-Carlo method was used to compute the thermal field. This approach does not require artificial boundary conditions but is probabilistic.

In our approach, we eliminate the need for artificial boundary conditions for the thermal field because we use a form of the solution which automatically satisfies the correct boundary conditions on the computational domain. In other words, these conditions do not need to be artificially imposed. The form of the solution we suppose is that it is the sum of convolution integrals involving the initial data and the source term. Our numerical method then discretizes these integrals in an efficient and accurate manner by using a recently developed method for the free space heat equation called the Fast Recursive Marching (FRM) method [31]. For the second equation of the phase-field model, the phase equation, we will use the RKC (Runge–Kutta–Chebyshev) method [44] (for the time integration of parabolic PDEs discretized by the method of lines). The RKC method is an explicit method which computes the solution of moderately stiff problems in a time interval with a small of number of stages and function evaluations while ensuring stability. We have not seen the use of the RKC method for the phase equation before in literature.

Our approach is complementary to mesh and time adaptivity. To simulate at very low undercoolings and long times, adaptivity must be incorporated into any method that solves for the thermal field. The FRM method can readily fit into an adaptive scheme because it accepts non-uniform, non-structured meshes and variable time steps. Incorporating adaptivity into the FRM method is ongoing work and adaptive FRM is not yet available at the present time. In this paper, we will present the method for uniform Cartesian meshes and explain where appropriate how to extend it to locally refined as well as unstructured meshes. We have implemented a simple adaptive enlargement of the computational domain as the crystal approaches the computational boundary. The thermal field can be computed on the enlarged computational domain in a totally natural way even though the support of the thermal field has long since expanded outside of the previous computational domain. Such an approach would not be feasible with typical finite-difference/element based methods.

Our approach is a competitor of the Monte-Carlo approach. The FRM is a deterministic method which produces the exact solution to the thermal equation modulo discretization errors. Monte-Carlo is a probabilistic method and the accuracy depends on the number of random walkers.

The extension of our method to three dimensions poses no theoretical difficulties. It awaits the development of the three-dimensional FRM method which so far is limited to two dimensions.

We summarize here the most important features of the Fast Recursive Marching method as applicable to the phase-field model. Details are fully described in [31].

- The numerical solution is based on the efficient evaluation of the integral representation of the solution to the constant coefficient, time dependent heat equation. Hence, no artificial boundary conditions are needed. The high cost typically associated with evaluating convolution integrals in time is drastically reduced. At each new time step, the solution at the current step can be computed using only values from a few previous time steps.
- An accurate solution to the evolving thermal field can be obtained on a computational domain whose size is dictated by the size of the crystal, and not by the thermal field. The value of thermal field can be adaptively computed on ever larger computational domains as the crystal grows even though the support of thermal field has exited the current computational domain at some time in the past.
- The computational time and memory requirements of the Fast Recursive Marching method are on the same order as for a standard explicit finite difference/element method with the same spatial resolution. The gain will come from the fact that the new approach allows smaller computational domains and hence fewer spatial discretization points in total. The savings increases with increasing simulation time since the diffusion of the thermal field occurs on a length scale that is $O(\sqrt{t})$, where t is the time of simulation.
- There are no numerical stability constraints on Δt. Temporal accuracy depends on the polynomial approximation order for the source term.
- The method is spectrally accurate in space.

We mention here that yet another possible approach, which does not appear to be widely used, involves formulating the exact boundary conditions for the thermal equation and then coupling these conditions to finite differences/elements solvers for the interior of the computational domain. The idea is that if heat is allowed to escape the computational domain as if the artificial boundary were not there, then the domain needs only be large enough to contain the crystal interface itself. Such transparent boundary conditions for the heat equation are described in [21,11,22,23]. However, we are not aware of any attempts to use these boundary conditions in the problem of modeling crystal growth.

This paper is a follow-up of the work presented in [31]. In that paper, a simplified phase-field model was included in the numerical results section. Here, we use a different (and anisotropic) model and incorporated the RKC method for the phase equation. More importantly, we discuss how to apply the FRM method in an appropriate way to the solution of the phase-field model. We explain how to choose the various parameters in the FRM method as well as show that one can easily incorporate adaptive enlargement of the computational domain into the phase-field computation. Much more detailed numerical results are also included.

This paper is organized as follows: In Section 2, we briefly describe the phase-field model and our discretization of the coupled thermal field and phase equations. In Section 3, we describe the Fast Recursive Marching method for solving the free space heat equation and explain how it should be applied to solve the thermal field equation. In Section 4, we show numerical results for the simulation of two-dimensional, anisotropic crystal growth. The lowest undercooling simulated is 0.025. Section 5 contains the conclusions.

2. The phase-field model for dendritic crystal growth

We will consider the solidification of a pure material. The model we will use are the phase-field equations, which consist of a coupled system of two diffusion-type equations governing a dimensionless temperature $u(x,y,t)$ (the thermal field) and the phase variable $\phi(x,y,t)$.

2.1. The model

We use the following formulation of the model equations, slightly modified from [26]:

$$u_t - \nabla^2 u = \frac{1}{2S}(h(\phi))_t, \quad \mathbf{x} = (x,y) \in \mathbb{R}^2,\ t > 0, \tag{1}$$

$$\tau_0 a(\theta)^2 \phi_t = \left[\frac{\phi(1-\phi^2)}{D} - \frac{\tau_0 S}{a_2 w_0^2} u(1-\phi^2)^2 + \frac{w_0^2}{D}\nabla_a^2(\theta)(\phi)\right], \quad \mathbf{x} = (x,y) \in \mathbb{R}^2,\ t > 0, \tag{2}$$

along with the choice of parameters,

$$h(\phi) = \phi, \quad a_2 = 0.6267, \quad \tau_0 = 1, \quad w_0 = 1.$$

Because the existing code for the FRM method requires that the coefficient of diffusion $D = 1$, we use the scaled time $t = Dt'$ in (1) and (2) instead of the time variable in [26], To normalize with respect to the undercooling parameter, we have also chosen to use the scaled thermal field $u = u'/S$ with respect to what was in [26]. Hence, the variables t and u used in (1) and (2) are related to their analogues t' and u' in [26] via the relations $t = Dt'$ and $u = u'/S$.

The thermal and the phase variables are subject to initial conditions:

$$u(x,y,0) = u_0(x,y), \quad \mathbf{x} = (x,y) \in \mathbb{R}^2, \tag{3}$$
$$\phi(x,y,0) = \phi_0(x,y), \quad \mathbf{x} = (x,y) \in \mathbb{R}^2. \tag{4}$$

The normalized temperature is $u = (T - T_M)/\Delta T$, where T_M is the equilibrium melting temperature at a planar interface and ΔT is the melt undercooling (the difference between the melting temperature and the initial temperature in the bulk liquid). Interfacial properties such as the crystal-melt interface energy and the local atomic attachment coefficient depend on interfacial orientation, i.e. they are anisotropic. In (1)–(4) the anisotropy of the material properties is included through the anisotropic Laplacian, which depends on the angle θ (for example, the angle that the normal to the iso-ϕ contours makes with the x-axis):

$$\nabla_a^2(\theta) = -\frac{\partial}{\partial x}\left(a(\theta)a'(\theta)\frac{\partial}{\partial y}\right) + \frac{\partial}{\partial y}\left(a(\theta)a'(\theta)\frac{\partial}{\partial x}\right) + \nabla\cdot(a^2(\theta)\nabla), \tag{5}$$

with θ computed by the formula

$$\theta = \tan^{-1}\left(\frac{\partial_y\phi}{\partial_x\phi}\right). \tag{6}$$

The anisotropy is typically modeled by

$$a(\theta) = 1 + \epsilon_4 \cos w\theta, \tag{7}$$

where ϵ_4 is the strength of the anisotropy and w is the crystalline symmetry (e.g., $w = 4$ for four-fold symmetry).

The phase variable varies from $\phi = -1$ (liquid) to $\phi = 1$ (solid), corresponding to the liquid and solid phases, respectively. The value of ϕ varies smoothly but sharply from -1 to 1 in a narrow diffuse interface region. The Stefan number, or dimensionless undercooling, is given by

$$S = \frac{c\Delta T}{L}, \tag{8}$$

where c is the specific heat per unit volume and L is the latent heat per unit volume. The undercooling is the parameter of primary importance in this paper. Typical simulations reported in the literature assume S to be in the range 0.05–1. However, experimental results often report values of S in the range 0.001–0.1 [28].

Since we will be using our FRM method to compute the thermal field, we do not require any numerical boundary conditions on the thermal field. For the phase-field variable, we use the RKC (Runge–Kutta–Chebyshev) method [44] which is an explicit solver for parabolic PDEs and boundary conditions on the computational boundary will be required. However, the phase-field variable ϕ can be considered constant in the bulk liquid away from the interface, and so, assuming the computational domain is large enough to contain the support of the solid–liquid interface, one can safely impose conditions compatible with $\phi \equiv -1$ on computational boundary (e.g. homogeneous Neumann conditions). Although the equation for ϕ also appears to be a diffusion-type equation, in fact, it is not a true diffusion equation. This can be seen by writing the term $\nabla_a(\theta)\nabla\phi$ in tensor notation as $\nabla L(\theta)\nabla\phi$ and noting that the resulting anisotropic tensor $L(\theta)$ is skew symmetric, with complex eigenvalues, and so does not behave as a true diffusion tensor [46]. As a consequence, the field for ϕ does not extend beyond the interface and is constant outside of this region.

The field of interest in the phase-field equations is represented by the phase variable. The thermal field is of interest near the dendritic tips but is of only secondary interest away from the crystal interface. For this reason, one would ideally like to use a computational domain for u which is the same size as needed for ϕ even at low undercoolings. We want the thermal field to diffuse out through the computational boundary as if the boundary were not there.

2.2. Discretization of coupled system

We use Algorithm 1 for the coupled equations in (1) and (2). It includes the adaptive enlargement of the computational domain.

Algorithm 1. FRM/RKC time marching algorithm with adaptive enlargement of computational domain

```
Initialization:
• Choose rectangular computational domain
      $\Omega^0 = [-R_1^0, R_1^0] \times [-R_2^0, R_2^0]$,
  so that it contains both the support of $\phi^0$ (support of $\phi$ is where the phase is not liquid) and the support of $u^0$.
• Choose a spatial discretization $N_x^0 = \frac{R_1^0}{\Delta x}$ and $N_y^0 = \frac{R_2^0}{\Delta y}$. The discretization points are $\{\mathbf{x}_{i,j} = (i\Delta x, j\Delta y), i \in \{-N_x^0, \ldots, N_x^0\}, j \in \{-N_y^0, \ldots, N_y^0\}\}$.
  For steps $m = 0, 1, 2, \ldots$ on the computational domain $\Omega^m$:
1. Obtain $\phi^{m+1} \approx \phi(\cdot, (m+1)\Delta t)$ at $\{\mathbf{x}_{i,j}\} \in \Omega^m$ using the RKC method by solving
      $\phi_t = \frac{1}{\tau_0 a(\theta)^2}\left[\frac{\phi(1-\phi^2)}{D} - \frac{\tau_0 S}{a_2 w_0^2} u^m (1-\phi^2)^2 + \frac{w_0^2}{D}\nabla_a^2(\theta)(\phi)\right], \quad t \in (m\Delta t, (m+1)\Delta t),$   (9)
   subject to the initial condition
      $\phi(\mathbf{x}_{ij}, m\Delta t) = \phi^m(\mathbf{x}_{ij})$,
   and homogeneous Neumann boundary condition
      $\frac{\partial \phi}{\partial t} = 0 \quad \text{on } \partial\Omega^m$.
2. Obtain $u^{m+1} \approx u(\cdot, (m+1)\Delta t)$ at $\{\mathbf{x}_{i,j}\} \in \Omega^m$ using the FRM method by solving
      $u_t - \nabla^2 u = \left\{\frac{1}{2S}(h(\phi))_t \equiv f\right\}$,   (10)
   subject to the initial condition
      $u(\mathbf{x}, 0) = u^0(\mathbf{x})$.
3. Decide if the support of $\phi^{m+1}$ is 'close' to $\partial\Omega^m$.
   • If so,
     (a) Make $R_1^{m+1}$ larger than $R_1^m$ and $R_2^{m+1}$ larger than $R_2^m$. Set $\Omega^{m+1}[-R_1^{m+1}, R_1^{m+1}] \times [-R_2^{m+1}, R_2^{m+1}]$.
     (b) Extend $\phi^{m+1}$ to $\{\mathbf{x}_{i,j}\} \in (\Omega^{m+1} - \Omega^m)$ by the value 'liquid'.
     (c) Extend $u^{m+1}$ to $\{\mathbf{x}_{i,j}\} \in (\Omega^{m+1} - \Omega^m)$ by inverse Fourier transform.
   • Otherwise, $\Omega^{m+1} := \Omega^m$.
```

The computational domain for both (9) and (10) at $t = m\Delta t$ will be Ω_m, which is supposed to contain the support of ϕ^m. It is clear from (1) that this means that Ω^m also always contains the source for u.

We treat the coupling of Eqs. (1) and (2) by evaluating the right hand side of each equation using previously computed values of the variables. This operator-splitting approach is formally first order accurate, although in practice one can often obtain accuracy rates higher than this.

For Eq. (9) the contribution of u to the right hand side is approximated by its value at $t = m\Delta t$. The boundary conditions on $\partial\Omega_m$ are homogeneous Neumann. Since $\phi \equiv -1$ in the bulk liquid, these boundary conditions are acceptable. Spatial discretization of (9) can be done in standard ways, for example, via finite differences.

We solve (10) using the FRM method, the details of which will be given in Section 3. There is no need for boundary conditions on $\partial\Omega_m$ because the correct far field behavior of u is automatically satisfied. The right hand side f of (10) will be treated as known, using already computed values of ϕ, via the following first order in time approximation:

$$f_{m+1}(\mathbf{x}) = \frac{1}{2S}\phi_t \approx \frac{1}{2S}\frac{\phi^{m+1} - \phi^m}{\Delta t}. \tag{11}$$

Higher order methods may be formulated with (9) and (10) as the basic components but will not be discussed here because the emphasis of this paper is on the correct treatment of the thermal field equation while avoiding overly large computational domains. The current implementation is done for two dimensions but there is a natural extension to three dimensions.

3. Fast recursive marching method

In this section we outline the Fast Recursive Marching (FRM) method and explain how to apply it to solve (10). We will only point out the main features of the FRM method, details can be found in [31].

The FRM method solves the heat equation in free space with a source term:

$$\tilde{u}_t(\mathbf{x},t) - \nabla^2\tilde{u}(\mathbf{x},t) = f(\mathbf{x},t), \quad \mathbf{x}\in\mathbb{R}^d,\ t>0, \tag{12}$$

supplemented by the initial condition

$$\tilde{u}(\mathbf{x},0) = \tilde{u}_0(\mathbf{x}), \quad \mathbf{x}\in\mathbb{R}^d. \tag{13}$$

The functions $f(\mathbf{x},t)$ and $\tilde{u}_0(\mathbf{x})$ are assumed to be compactly supported for any fixed t. Because of the choice of the normalization in the phase-field model, we have $u=1$ as the base line value for the thermal field in the bulk liquid in the initial configuration. To obtain the baseline value of 0 and compact support for the initial configuration, we have defined the new variable $\tilde{u} = u-1$. We will concentrate on the case $d=2$ but will use the notation d if the statement applies to other dimensions as well.

We recall the right hand side of (10) is $f = -\frac{1}{2S}\phi_t$, and that we will use the approximation of it given in (11). Clearly, the support of $f(\cdot,t)$ is on a narrow band around the crystal-melt interface at time t. (The width of this band depends on the speed of crystalization.) Given Δt, we will compute the approximate solution at $t=\Delta t, 2\Delta t,\ldots,m\Delta t,\ldots$, using an increasing set of computational domains, $\Omega_1\subset\cdots\subset\Omega_m\cdots$, where each Ω_i contains the support of $f(\cdot,t)$, for all $t\leqslant i\Delta t$. The domain will be enlarged adaptively after the support of the phase variable ϕ, which is computed via (9), is determined. In addition, we will assume that all the computational domains contain the support of $\tilde{u}_0$.

The FRM method is applicable to problems where the source term f is k-times differentiable. Let $t_{\max}$ be an estimate of the longest simulation time of interest, (the actual simulation time may be much shorter), and let $B=\prod_{i=1}^{d}[-R_i/2, R_i/2]$ be a rectangle which contains the support of $f(\cdot,t)$, for all $t\leqslant t_{\max}$. The value $\max\{R_1,\ldots,R_d\}$ will be used to choose certain parameters in the FRM method.

We assume that $f(\mathbf{x},t)\in C^k(B\times[0,t_{\max}])$. This is satisfied because ϕ is smooth in the diffuse interface region. The fact that ϕ varies sharply in this narrow region will affect the parameters we choose in the algorithm and we will address this when we come to the relevant parameters.

From standard potential theory [20], the solution to (12) and (13) can be written as

$$\tilde{u}(\mathbf{x},t) = \int_{\mathbb{R}^d} k(\mathbf{x}-\mathbf{y},t)\tilde{u}_0(\mathbf{y})\,d\mathbf{y} + \int_0^t\int_{\mathbb{R}^d} k(\mathbf{x}-\mathbf{y},t-\tau)f(\mathbf{y},\tau)\,d\mathbf{y}\,d\tau, \tag{14}$$

where the fundamental solution for the heat equation in $\mathbb{R}^d$ is

$$k(\mathbf{x},t) := \frac{e^{-\|\mathbf{x}\|^2/4t}}{(4\pi t)^{\frac{d}{2}}}, \quad t>0. \tag{15}$$

We will refer to the first integral in (14) as an *initial potential* and the second integral as a *volume potential*. Straightforward discretization of the above integrals leads to an prohibitively expensive numerical scheme – the solution is dependent on the full space-time history of the diffusion process. If M is the number of time steps, and the computational domain is not adaptively enlarged during these M time steps and contains N spatial discretization points, it is easy to see that $O(N^2M + N^2M^2)$ work is required for those M time steps with a naive discretization.

The Fast Recursive Marching method is the accelerated computation of these integrals. There are three components to the algorithm;

- We accelerate the time convolution by doing the time stepping in the Fourier domain.
- To compute the inverse Fourier transform, we use an efficient quadrature of the inverse Fourier integral that involves many fewer Fourier nodes than a naive trapezoidal rule.
- These Fourier nodes are not equi-spaced, so to transform to and from Fourier space in an efficient manner, we use the Non-uniform FFT (NUFFT) [17,30].

In this way, we are able to reduce the total computational cost of evaluating (14) to $O(M\cdot(N_f+N_p)\log(N_f+N_p))$, where N_p is the number of discretization points in the physical domain, and N_f is the number of nodes in the Fourier domain (which depends on the desired tolerance and an estimate of the smoothness of f). In our numerical experiments, N_f is typically on the same order as N_p. For the same N_p, this algorithmic complexity is comparable to finite difference or finite elements methods. The advantage of this approach is that many fewer points in the physical domain will be needed for long-time simulation of slow solidification because of the smaller computational domain.

3.1. Fourier representation of the solution

We propose time stepping using the Fourier transform of $\tilde{u}$:

$$\hat{u}(\mathbf{s},t) = \int_{\mathbb{R}^d} e^{i\mathbf{s}\cdot\mathbf{x}}\tilde{u}(\mathbf{x},t)\,d\mathbf{x}, \tag{16}$$

where the inverse Fourier transform is given by

$$\tilde{u}(\mathbf{x},t)=\frac{1}{(2\pi)^d}\int_{\mathbb{R}^d}e^{-i\mathbf{s}\cdot\mathbf{x}}\hat{u}(\mathbf{s},t)\,d\mathbf{s}. \tag{17}$$

It is straightforward to show that $\hat{u}(\mathbf{s},t)$ satisfies the ordinary differential equation

$$\frac{d\hat{u}}{dt}(\mathbf{s},t)=-\|\mathbf{s}\|^2\hat{u}(\mathbf{s},t)+\hat{f}(\mathbf{s},t), \tag{18}$$

where

$$\hat{f}(\mathbf{s},t)=\int_{\mathbb{R}^d}e^{i\mathbf{s}\cdot\mathbf{x}}f(\mathbf{x},t)\,d\mathbf{x}. \tag{19}$$

Thus, time stepping for $\hat{u}(\mathbf{s},t)$ can take the form

$$\hat{u}(\mathbf{s},t)=e^{-\|\mathbf{s}\|^2\Delta t}\hat{u}(\mathbf{s},t-\Delta t)+\Phi(\mathbf{s},t,\Delta t), \tag{20}$$

where

$$\Phi(\mathbf{s},t,\Delta t)=\int_{t-\Delta t}^{t}e^{-\|\mathbf{s}\|^2(t-\tau)}\hat{f}(\mathbf{s},\tau)\,d\tau. \tag{21}$$

Thus, in the Fourier domain, time convolution can be accelerated from quadratic complexity to linear complexity: $\hat{u}(\mathbf{s},t)$ is simply *damped* and *updated* by a local (in time) computation at each time step, as indicated in (20). We will refer to $\Phi(\mathbf{s},t,\Delta t)$ in (21) as the *update integral*. The observation in (20) is not new, but for it to be useful, one must be able to choose a small set of Fourier modes which can adequately represent the function $\tilde{u}$ and one must be able to move easily between Fourier and physical domains.

3.2. Choice of Fourier nodes

The choice of Fourier nodes s depends on our choice of quadrature for evaluating

$$\tilde{u}(\mathbf{x},t)=\frac{1}{(2\pi)^d}\int_{\mathbb{R}^d}e^{-i\mathbf{s}\cdot\mathbf{x}}\hat{u}(\mathbf{s},t)\,d\mathbf{s}. \tag{22}$$

An optimal choice for the Fourier nodes is one that results from a quadrature for (22) which meets the desired accuracy requirements with as few nodes as possible.

Let us recall that the source is smooth: $f(\cdot,t)\in C^k(B)$. Thus, $\hat{f}(\mathbf{s},t)=O(\|\mathbf{s}\|^{-k})$ for large $\mathbf{s}$. A straightforward calculation shows that if ϵ is the error tolerance, then the evaluation of $\hat{f}(\mathbf{s},t)$ in (19) needs to be done only for $\|\mathbf{s}\|\leqslant P$, where $P=O(1/\epsilon)^{\frac{1}{k}}$. We will refer to P as the *high-frequency cutoff.*

For the phase-field application, f is a function of ϕ. Since we will be using methods which are of low order accuracy in time, we choose an error tolerance of $\epsilon=10^{-4}$ because we do not expect to get more than 4 digits of accuracy. We make an estimate for the smoothness of ϕ. For example $(1/\epsilon)^{\frac{1}{2}}=100,(1/\epsilon)^{\frac{1}{3}}\approx 22$. For our simulations we typically set $P=20$ but this can be modified if it is found to be inadequate.

To evaluate the finite Fourier integral

$$\tilde{u}(\mathbf{x},t)\approx\frac{1}{(2\pi)^d}\int_{\|\mathbf{s}\|<P}e^{-i\mathbf{s}\cdot\mathbf{x}}\hat{u}(\mathbf{s},t)\,d\mathbf{s}, \tag{23}$$

we use a mesh which is clustered toward the origin in $\mathbf{s}$-space on dyadic intervals that is capable of resolving the integrand for all time to any desired precision ϵ. Intuitively, the reason for this exponential clustering at the origin can be understood from considering the Fourier transform of the free space Green's function $\frac{e^{-\|\mathbf{x}\|^2/4t}}{(4\pi t)^{\frac{d}{2}}}$ itself, namely the function $e^{-\|\mathbf{s}\|^2 t}$. For large t, this function is sharply peaked near $\mathbf{s}=0$ and the accurate resolution of this function is what ensures that the lowest frequency terms diffuse into the infinite region correctly.

A slight modification of Theorem 2.1 in [18] yields error bounds *in one dimension.*

Theorem 1. adapted from [18]*Let $\epsilon>0$ be the desired precision, let $L_{\min}=-\log(1/\epsilon)$ and let $L_{\max}=\lceil\log P\rceil$, where P is the high-frequency cutoff. Further, let $\{s_{j,1},\ldots,s_{j,n(j)}\}$ and $\{w_{j,1},\ldots,w_{j,n(j)}\}$ be the nodes and weights for the $n(j)$-point Gauss–Legendre quadrature on the interval $[2^j,2^{j+1}]$, where $n(j)=\max(L_x2^{j+3/2},8\log(1/\epsilon))$. Then, in one space dimension,*

$$\left|\int_{|s|<P}e^{-is\cdot x}\hat{u}(s,t)\,ds-\sum_{j=L_{\min}}^{L_{\max}}\sum_{k=1}^{n(j)}(e^{is_{j,k}x}+e^{-is_{j,k}x})\hat{u}(s_{j,k},t)w_{j,k}\right|=O(\epsilon) \tag{24}$$

for $|x|\leqslant L_x$, for $t\in(0,\infty)$.

We will denote by $Q = Q(\epsilon, P, L_x)$ the number of nodes required on the positive axis, in one dimension. Using a tensor product of this one-dimensional rule, and the fact that the physical quantities are real, the number of nodes required in the Fourier domain is $N_f = (2Q)^d/2$, which is of the order $N_f = O((\log(1/\epsilon) + L_xP)^d)$ (see [31]). If instead of this quadrature rule, we had used the trapezoidal rule for (23), then $O(\frac{L_xP}{\epsilon})^d$ nodes would have been required in order to resolve $e^{-i\mathbf{s}\cdot\mathbf{x}}e^{-\|\mathbf{s}\|^2t}$ for all time $t > 0$, which is a significantly larger number of nodes, especially when ϵ is small.

The parameter L_x is the largest magnitude of the evaluation points $\mathbf{x}$. Since we have supposed that the largest computational domain needed will be contained in $B = \prod_{i=1}^d[-R_i/2, R_i/2]$, we will set $L_x = \max\{R_1/2, \ldots, R_d/2\}$.

3.3. The forward transform

To compute $\hat{f}(\mathbf{s}, t)$ from the data $f(\mathbf{x}, t)$, since we have assumed that the data is supported in the rectangle B, we need to compute the finite integral:

$$\hat{f}(\mathbf{s}, t) = \int_{-R_1/2}^{R_1/2} \cdots \int_{-R_d/2}^{R_d/2} e^{i\mathbf{s}\cdot\mathbf{x}} f(\mathbf{x}, t)\, d\mathbf{x}, \tag{25}$$

where $\mathbf{x} = (x_1, \ldots, x_d),\ \mathbf{s} = (s_1, \ldots, s_d)$.

Let us recall that $P = O(1/\epsilon)^{\frac{1}{k}}$ is the *high-frequency cutoff*. This bounds the oscillatory behavior of the term $e^{i\mathbf{s}\cdot\mathbf{x}}$ in the integrand of (25). Together with the fact that $f(\mathbf{x}, t)$ is smooth, it follows that the trapezoidal rule applied to each dimension in (25) with N points will yield $O(N^{-k})$ accuracy [9]. In dimension i, the error will begin decaying rapidly once N_i is of the order $O(PR_i)$, meaning that the integrand is well-resolved. If $f(\mathbf{x}, t)$ is given on a uniform mesh with N_i points in each dimension, the trapezoidal rule yields

$$\hat{f}(\mathbf{s}, t) \approx \prod_{i=1}^{d}\left(\frac{R_i}{N_i}\right) \sum_{n_1=1}^{N_1} \cdots \sum_{n_d=1}^{N_d} e^{i\mathbf{s}\cdot\mathbf{x}_n} f(\mathbf{x}_n, t), \tag{26}$$

where $\mathbf{x}_n = (-R_1/2 + n_1(R_1/N_1), \ldots, -R_d/2 + n_d(R_d/N_d))$.

We have used the uniform trapezoidal rule in (26) to obtain our numerical results. But we emphasize here that it is desirable to use a locally refined spatial discretization of (25) instead, putting a finer discretization of $\mathbf{x}$ points near the crystal-melt interface and a coarser discretization away from it. Given a code which computes ϕ on an adaptive mesh, for example on locally refined rectangles or triangles, it is easy to generate the quadrature weights to get a modified rule for the Fourier transform in (25). The only restriction is that the density of the discretization in physical space still needs to satisfy $\Delta x_i \leqslant O\left(\frac{1}{PR_i}\right)$ globally to resolve the oscillations in $e^{i\mathbf{s}\cdot\mathbf{x}}$. Spatial refinement is then used to resolve the other term, $f(\mathbf{x}, t)$. We denote the number of points $\mathbf{x}$ in physical space by N_p (with $N_p = \prod_{i=1}^d N_i$ if refinement is not used).

3.4. The non-uniform FFT

Let us denote by $s_1, \ldots, s_Q$ and $w_1, \ldots, w_Q$ the full set of one-dimensional quadrature nodes and weights described in the preceding section. Then, we need to compute the discrete transform

$$\tilde{u}(\mathbf{x}, t) \approx \frac{1}{(2\pi)^d} \sum_{j_1=1}^{Q} \cdots \sum_{j_d=1}^{Q} e^{-i\mathbf{s}_j\cdot\mathbf{x}} \hat{u}(\mathbf{s}_j, t) w_{j_1} \cdots w_{j_d}, \tag{27}$$

where $\mathbf{s}_j = (s_{j_1}, \ldots, s_{j_d})$.

The summation in (27) must be carried out for each evaluation point $\mathbf{x}$. Direct evaluation at each of the N_p points $\mathbf{x}$ defined in Section 3.3 would require $O(N_f \cdot N_p)$ work. Likewise, the sum (26) must be evaluated at each of the N_f spectral nodes, also requiring $O(N_f \cdot N_p)$ work.

Since the $\{\mathbf{s}_j\}$ are not uniformly spaced, the classical Fast Fourier Transform (FFT) cannot be applied to accelerate this computation. In the last decade, however, suitable fast algorithms have been developed that we refer to as Non-Uniform Fast Fourier Transforms (NUFFTs). Detailed descriptions and additional references can be found in [1,12,13,15,17,30,37]. Like the FFT, these algorithms evaluate sums of the form (26) and (27) in $O((N_f + N_p)\log(N_f + N_p))$ work, for any fixed precision.

When the $\mathbf{x}$ points are uniformly spaced, the sums in (26) and (27) are computed by what are called type-1 and type-2 NUFFTs [30]. When spatial refinement in physical space is used, the target and source points in (26) and (27) are both non-uniformly spaced, and the sums are computed by what is called a type-3 NUFFT.

3.5. Computing the update integral

The update integral (21) can be computed to high order accuracy using a standard product integration approach [9]. We first approximate $\hat{f}(\mathbf{s}, t)$ by a polynomial in time, and then integrate the resulting polynomial analytically. Linear approximation of $\hat{f}(\mathbf{s}, t)$ yields second order accuracy and the rule [19]

$$\Phi(\mathbf{s},t,\Delta t) = W_0(\mathbf{s},\Delta t)\hat{f}(\mathbf{s},t) + W_1(\mathbf{s},\Delta t)\hat{f}(\mathbf{s},t-\Delta t), \tag{28}$$

$$W_0(\mathbf{s},\Delta t) = \frac{e^{-z}-1+z}{z^2}\Delta t, \quad W_1(\mathbf{s},\Delta t) = \frac{1-e^{-z}-ze^{-z}}{z^2}\Delta t, \tag{29}$$

where $z = \|\mathbf{s}\|^2\Delta t$.

For a cubic approximation of $\hat{f}(\mathbf{s},\tau)$, yielding fourth order accuracy in time, we have

$$\Phi(\mathbf{s},t,\Delta t) = W_0(\mathbf{s},\Delta t)\hat{f}(\mathbf{s},t) + W_1(\mathbf{s},\Delta t)\hat{f}(\mathbf{s},t-\Delta t) + W_2(\mathbf{s},\Delta t)\hat{f}(\mathbf{s},t-2\Delta t) + W_3(\mathbf{s},\Delta t)\hat{f}(\mathbf{s},t-3\Delta t), \tag{30}$$

where

$$W_0(\mathbf{s},\Delta t) = \frac{(6+2z^2-6z)e^{-z} + (6z^3-11z^2-6+12z)}{6z^4}\Delta t,$$
$$W_1(\mathbf{s},\Delta t) = \frac{(-z^2+2z^3+6-4z)e^{-z} + (-6z^2-6+10z)}{-2z^4}\Delta t,$$
$$W_2(\mathbf{s},\Delta t) = \frac{(2z+2z^2-6)e^{-z} + (-8z+3z^2+6)}{-2z^4}\Delta t,$$
$$W_3(\mathbf{s},\Delta t) = \frac{(z^2-6)e^{-z} + (6+2z^2-6z)}{6z^4}\Delta t.$$

Some care needs to be taken in computing the weights above. If z is small, the formulae above are subject to significant cancellation error. In that case, the weights can be computed by Taylor expansion of the exponentials, carried out to a sufficient number of terms. A reasonable compromise is to switch from the analytic expression to the Taylor series if $z < 10^{-3}$. Four terms in the Taylor series are then sufficient for 12 digit accuracy.

3.6. Adaptive enlargement of the computational domain

The computational domain for $\tilde{u}$ must contain the crystal-melt interface. As this interface grows in size, we can enlarge the computational domain. Suppose at $t = m\Delta t$, the computational domain for both $\tilde{u}$ and ϕ is Ω_m and at the next time step, we would like to enlarge it to $\Omega_{m+1} \supset \Omega_m$. Since we compute the Fourier coefficients of $\tilde{u}$ at $t = (m+1)\Delta t$, which are the values $\hat{u}(\mathbf{s},(m+1)\Delta t)$, to obtain $\tilde{u}(\mathbf{x},(m+1)\Delta t)$ on the enlarged domain Ω_{m+1}, we simply perform the inverse Fourier transform in (27) at the set of points $\mathbf{x} \in \Omega_{m+1}$ instead of only in Ω_m. All the information about $\tilde{u}$ (hence u) is stored in the Fourier coefficients and no information is lost even though the thermal field has expanded far outside of Ω_m. This is in contrast to a finite difference/element method where even if exact artificial boundary conditions were imposed on $\partial\Omega_m$, it is usually not possible to accurately reconstruct the solution u outside of the domain Ω_m.

3.7. The numerical scheme

With the full complement of tools in place, we can now provide an informal description of the Fast Recursive marching (FRM) method. For lth order accuracy in time, we refer to the method as FRM(l). The method for a second order accurate scheme is described in Algorithm 2.

```
Algorithm 2. The Fast Recursive Marching algorithm FRM(2) for solving the heat equation in $\mathbb{R}^d$
Initialization:
 1. Choose error tolerance $\epsilon$ and high-frequency cutoff $P$.
 2. Estimate the size of the computational domain, which must contains the source, during the simulation time of
    interest. Let it be the box $[-R_1,R_1]\times[-R_2,R_2]$.
 3. Obtain $Q(\epsilon,P,\max\{R_1,R_2\})$ quadrature nodes and weights in Fourier space.
 4. Select the time step $\Delta t$.
 5. Compute weights $W_0, W_1$ from (29).
Transform initial data:
 1. Initialize $\hat{u}(\mathbf{s},0)$ at all $N_f = Q^d$ spectral nodes by computing the forward NUFFT of $\tilde{u}_0(\mathbf{x})$.
 2. Compute $\hat{f}(\mathbf{s},0)$ at all $N_f$ spectral nodes by computing the forward NUFFT of $f(\mathbf{x},0)$.
For $m = 1,\ldots$:
 1. Compute $\hat{f}(\mathbf{s},m\Delta t)$ at all $N_f$ spectral nodes by computing the forward NUFFT of $f(\mathbf{x},m\Delta t)$ using the data on $\Omega_{m-1}$.
 2. Update $\hat{u}(\mathbf{s},m\Delta t)$ according to (20):
       $\hat{u}(\mathbf{s},m\Delta t) = e^{-\|\mathbf{s}\|^2\Delta t}\hat{u}(\mathbf{s},(m-1)\Delta t) + W_0(\mathbf{s},\Delta t)\hat{f}(\mathbf{s},m\Delta t) + W_1(\mathbf{s},\Delta t)\hat{f}(\mathbf{s},(m-1)\Delta t).$
 3. Compute $\tilde{u}(\mathbf{x},m\Delta t), \mathbf{x}\in\Omega_m$, from $\hat{u}(\mathbf{s},m\Delta t)$ using the (inverse) NUFFT. The domain $\Omega_m$ may be larger than $\Omega_{m-1}$.
```

The total computational cost is $O(M\cdot(N_f+N_p)\log(N_f+N_p))$, where N_f is the number of Fourier nodes and N_p is the number of $\mathbf{x}$ points in the physical domain. Two NUFFT calculations are required per time step.

4. Numerical results

We show numerical results for solving the phase-field model equations in (1) and (2) using Algorithm 1 which computes ϕ using the RKC method and computes u using the FRM method. The anisotropy parameters we have chosen are

$$w = 4, \quad \epsilon_4 = 0.05.$$

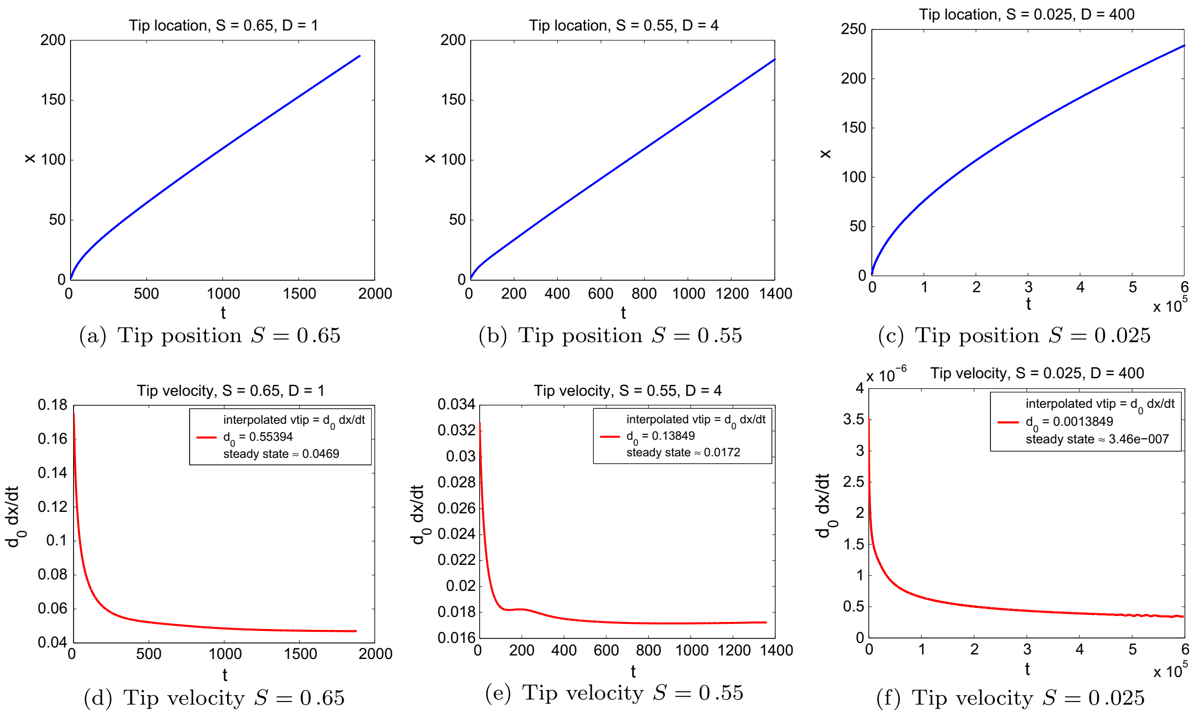
The two important parameters which will vary in the numerical experiments will be the diffusion coefficient D and the undercooling parameter S. Steady-state tip velocities were calculated for the following cases:

1. $S = 0.65, D = 1$,
2. $S = 0.55, D = 4$,
3. $S = 0.025, D = 400$.

We compare our results with those obtained in [26]. The spatial discretization for our algorithm was $\Delta x = \Delta y = 0.4$, the same as in [26]. We were able to use larger time steps: $\Delta t = 0.1$ for case 1, $\Delta t = 1$ for case 2, and $\Delta t = 10$ for case 3, than that used in [26] due to the fact the RKC method admits large time steps and the FRM is unconditionally stable. The time step used in [26] was $\Delta t = 0.016$ which is equivalent to $\Delta t = 0.016$ for case 1, $\Delta t = 0.064$ for case 2, and $\Delta t = 6.4$ for case 3. For case 1 we obtained $v_{tip} = 0.0469$ compared with $v_{tip} = 0.0465$ in [26]. For case 2 we obtained $v_{tip} = 0.0172$ compared with $v_{tip} = 0.0174$ in [26].

We also computed a case of very low undercooling, case 3, where $S = 0.025$ and $D = 400$. This undercooling is much lower than the cases computed in [26], where S ranges from 0.65 to 0.25. We obtained a steady state tip velocity of 3.46×10^{-7}. We were not able to compute a Green's function steady-state solution for this case as was done in [26] for higher undercoolings due to the lack of a publicly available code to make this calculation. Tip position and velocity as function of time for these simulations are shown in Fig. 1.

We now discuss case 3, where $S = 0.025$ and $D = 400$, in more detail. Snapshots of the crystal at various time points for the low undercooling case, $S = 0.025$ and $D = 400$, are plotted in Fig. 2. The crystal and the thermal field are shown inside the computational domain Ω^m at each time point. The computational domain goes from $[-10, 10] \times [-10, 10]$ at $t = 0$ to $[-256, 256] \times [-256, 256]$ at $t = 600,000$. Since all the information about $u(\cdot, m\Delta t)$ is stored in the Fourier coefficients, $\hat{u}(\cdot, m\Delta t)$, enlarging Ω_m just involves computing the inverse Fourier transform at a larger set of $\mathbf{x}$ points at the current time step. This is in contrast to finite difference/elements methods where one must enlarge Ω_m as soon as the support of u_m approaches $\partial\Omega_m$; in our approach, we do not have to enlarge Ω_m until the support of ϕ_m approaches $\partial\Omega_m$, a great advantage at low undercoolings. In Fig. 3 we show the support of the thermal field at $t = 300,000$ which is much larger than the computational domain we used ($[-174, 174] \times [-174, 174]$ at $t = 300,000$).



(a) Tip position $S = 0.65$ (b) Tip position $S = 0.55$ (c) Tip position $S = 0.025$

(d) Tip velocity $S = 0.65$ (e) Tip velocity $S = 0.55$ (f) Tip velocity $S = 0.025$

Fig. 1. Steady state tip velocity.

We made an estimate that the crystal will be bounded on the box $[-300, 300] \times [-300, 300]$ for all simulation time of interest, meaning that steady state will be reached before the crystal reaches this size. This means that $R_1 = R_2 = 300$ in the FRM method. With our choice of the high-frequency cutoff $P = 20$ and the error tolerance $\epsilon = 1e-4$, we have 1903 positive Fourier nodes in one dimension, this means that there are $N_f = 2(1903)^2 = 7,242,818$ total nodes in the Fourier domain in two dimensions. For case 3 at $t = 600,000$ there were $N_p = 1280^2 = 1,638,400$ nodes in the physical domain. In this example, both FRM(2) (for u) and RKC (for ϕ) use the same number of (uniformly spaced) discretization points in physical space.

RKC method is an explicit method where in each time interval Δt a number of stages are taken where the right hand side of Eq. (9) is evaluated several times. Each evaluation of the right hand side requires $O(N_p)$ work. The number of evaluations per time step Δt ranges from two to ten. In each time step of FRM(2), two non-uniform FFTs are computed. Typically, a NUFFT takes three times as long as one uniform FFT. We expect that for the same N_p, one iteration of FRM(2) will take six times as long as one right hand side evaluation of RKC. If there are two right hand size evaluations in one step of the RKC method, then the ratio between the timings of one iteration of the FRM(2) method and one iteration of the RKC method will be three to one. The ratio will be smaller if the number of right hand side evaluations per step of the RKC method is higher than two. We show in Fig. 4 the timings of the RKC method and the FRM(2) method over a range of time steps near the end of the simulation. The average time of the FRM method is about twice that of the RKC method per time step. The number of spatial discretization points (normalized by dividing by max N_p) is also plotted in the figure. It is clear that the timing for RKC increases as N_p increases. The largest prime factor of N_p (normalized by dividing by max {largest prime factor of N_p}) is also plotted (labelled 'LPF N_p'). It is clear that the timing of FRM(2) is influenced by the size of the largest prime factor of N_p, which is expected.

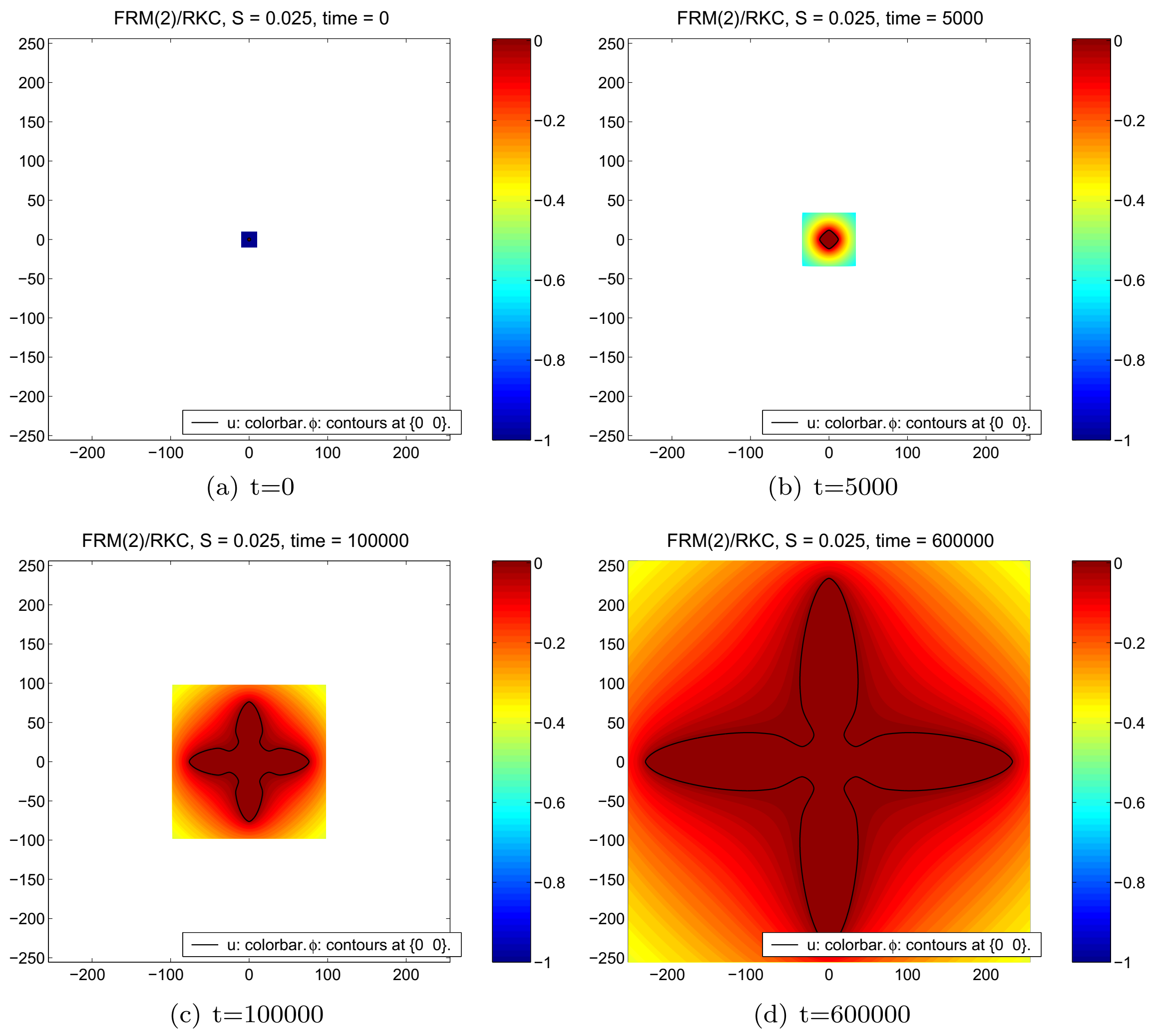


Fig. 2. The thermal field and phase plotted on the adaptive computational domain. The thermal field is computed via the FRM(2) method. The phase is computed via the RKC method. When the thermal field is computed with the FRM method heat diffuses out of the computational domain correctly and simulation can continue until the solidification front reaches the computational boundary.

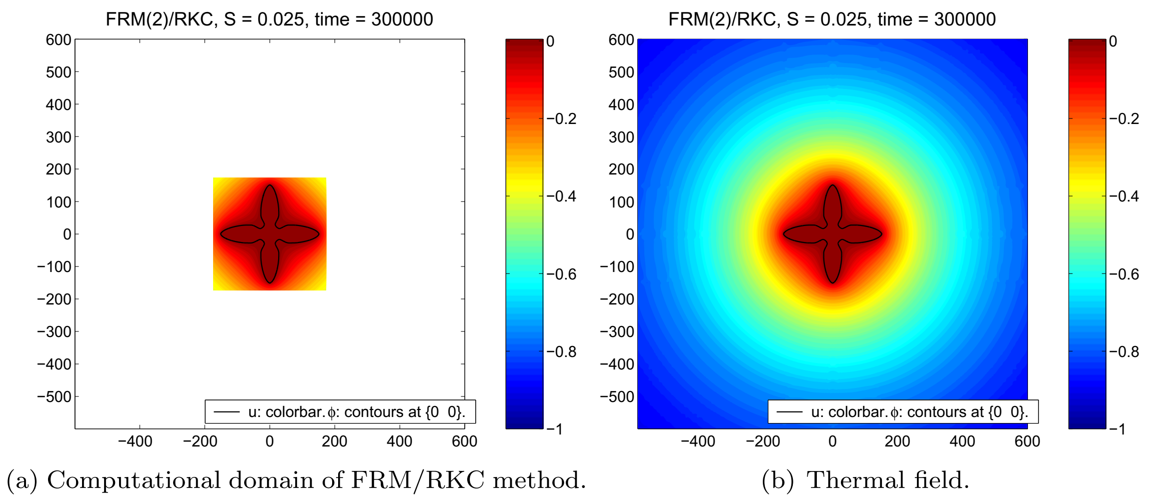


Fig. 3. The computational domain required by the FRM/RKC method can be much smaller than the thermal field. Competing finite difference or finite elements methods require a computational domain that contains the entire thermal field.

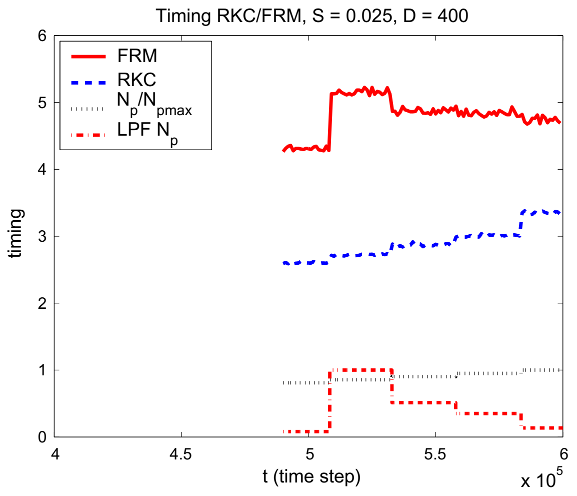


Fig. 4. Timings of the RKC method and the FRM(2) method over a range of time steps. The average time of the FRM method is about twice that of the RKC method per time step. The timing for RKC increases as N_p increases. The timing of FRM(2) is influenced by the size of the largest prime factor of N_p (labelled 'LPF N_p').

We emphasize that for the case of low undercooling, the increase in work and timing of the FRM(2) method compared to using, for example, the RKC method, for the thermal field is small compared to the additional work that would have been required if we were forced to compute u on a much larger computational domain due to the wrong artificial boundary conditions. In that case, the domain would have had side length $O(\sqrt{t})$, where t is the simulation time, which translates to $N_p = O(\sqrt{t}^d)$ discretization points. See Fig. 3 to see the support of the thermal field at $t = 300,000$ for this test case.

5. Conclusions

We used the Fast Recursive Marching method, a Fourier-based method for the solution of the heat equation in free space with a smooth source term, to compute the thermal field in the phase-field model of dendritic solidification. In the case of low undercoolings, this algorithm facilitates efficient and accurate long-time simulations because it allows the use of smaller computational domains for the thermal field.

Future work needs to be done to incorporate spatial adaptivity into the code, so that the mesh is locally refined to resolve the micro-structures that emerge at the solid–liquid interface. The FRM method can also be used on more general types meshes: logically Cartesian, triangular, or unstructured meshes. There is also a natural extension to three dimensions.

While we have demonstrated our solver on the phase-field equations, it can also be incorporated into solvers for the sharp interface model. This would involve solving an initial boundary value heat equation instead of the free space heat equation with a source term.

References

[1] G. Beylkin, On the fast Fourier transform of functions with singularities, Appl. Comput. Harmonic Anal. 2 (1995) 363–383.
[2] W. Boettinger, J.A. Warren, C. Beckermann, Phase-field simulation of solidification, Ann. Rev. Mat. Res. 32 (2002) 163–194.
[3] R.J. Braun, B.T. Murray, Adaptive phase-field computations of dendritic crystal growth, J. Cryst. Growth 174 (1–4) (1997) 41–53.
[4] L.N. Brush, R.F. Sekerka, A numerical study of two-dimensional crystal-growth forms in the presence of anisotropic growth-kinetics, J. Cryst. Growth 96 (2) (1989) 419–441.
[5] G. Caginalp, P. Fife, Phase-field methods for interfacial boundaries, Phys. Rev. B (1986) 7792–7794.
[6] S. Chen, B. Merriman, S. Osher, P. Smereka, A simple level set method for solving Stefan problems, J. Comput. Phys. 135 (1) (1997) 8–29.
[7] V. Cristini, J. Lowengrubb, Three-dimensional crystal growth I: linear analysis and self-similar evolution, J. Cryst. Growth 240 (2002) 267–276.
[8] V. Cristini, J. Lowengrubb, Three-dimensional crystal growth II: nonlinear simulation and control of the Mullins Sekerka instability, J. Cryst. Growth 266 (2004) 552–567.
[9] P.J. Davis, P. Rabinowitz, Methods of Numerical Integration, Academic Press, San Diego, 1984.
[10] S.H. Davis, Theory of Solidification, Cambridge University Press, Cambridge, England, 2001, Cambridge Monographs on Mechanics.
[11] E. Dubach, Artificial boundary conditions for diffusion equations: numerical study, J. Comput. Appl. Math. 70 (1) (1996) 127–144.
[12] A. Dutt, V. Rokhlin, Fast Fourier transforms for nonequispaced data, SIAM J. Sci. Comput. 14 (1993) 1368–1383.
[13] J.A. Fessler, B.P. Sutton, Nonuniform fast Fourier transforms using min–max interpolation, IEEE Trans. Signal Proc. 51 (2003) 560–574.
[14] G. Fix, Phase field models for free boundary problems, in: A. Fasano, M. Primicerio (Eds.), Free Boundary Problems, Theory and Application, 1983, pp. 580–589.
[15] K. Fourmont, Non-equispaced fast Fourier transforms with applications to tomography, IEEE Trans. Signal Proc. 9 (2003) 431–450.
[16] M.E. Glicksman, A.O. Lupulescu, Dendritic crystal growth in pure materials, J. Cryst. Growth 264 (2004) 541–549.
[17] L. Greengard, J.Y. Lee, Accelerating the nonuniform fast Fourier transform, SIAM Rev. 46 (2004) 443–454.
[18] L. Greengard, P. Lin, Spectral approximation of the free-space heat kernel, Appl. Comput. Harmon. Anal. 9 (1) (2000) 83–97.
[19] L. Greengard, J. Strain, A fast algorithm for the evaluation of heat potentials, Commun. Pure Appl. Math. 43 (8) (1990) 949–963.
[20] R.B. Guenther, J.W. Lee, Partial Differential Equations of Mathematical Physics and Integral Equations, Prentice-Hall, Inglewood Cliffs, New Jersey, 1988.
[21] L. Halpern, J. Rauch, Absorbing boundary conditions for diffusion equations, Numer. Math. 71 (2) (1995) 185–224.
[22] H. Han, Z. Huang, Exact and approximating boundary conditions for the parabolic problems on unbounded domains, Comput. Math. Appl. 44 (5–6) (2002) 655–666.
[23] H. Houde, Z. Huang, A class of artificial boundary conditions for heat equation in unbounded domains, Comput. Math. Appl. 43 (6–7) (2002) 889–900.
[24] A. Karma, Phase field modeling, in: S. Yip (Ed.), Handbook of Materials Modeling, Methods and Models, vol. 1, Springer, Netherlands, 2005, pp. 2087–2103.
[25] A. Karma, W.J. Rappel, Phase-field method for computationally efficient modelling of solidification with arbitrary interface kinetics, Phys. Rev. E (1996) R3017–R3020.
[26] Alain Karma, Wouter-Jan Rappel, Quantitative phase-field modeling of dendritic growth in two and three dimensions, Phys. Rev. E 57 (4) (1998) 4323–4349.
[27] W. Kurz, Fundamentals of Solidification, Trans Tech, Aedermannsdorf, Switzerland, 1992.
[28] J.C. LaCombe, M.B. Koss, D.C. Corrigan, A.O. Lupulescu, L.A. Tennenhouse, M.E. Glicksman, Implications of the interface shape on steady-state dendritic crystal growth, J. Cryst. Growth 206 (1999) 331–344.
[29] J.S. Langer, Instabilities and pattern formation in crystal growth, Rev. Mod. Phys. 52 (1980) 1–28.
[30] J.Y. Lee, L. Greengard, The type 3 nonuniform FFT and its applications, J. Comput. Phys. 206 (1) (2005) 1–5.
[31] J.R. Li, L. Greengard, On the numerical solution of the heat equation i: fast solvers in free space, J. Comput. Phys. 226 (2007) 1891–1901.
[32] Z. Li, B. Soni, Fast and accurate numerical approaches for Stefan problems and crystal growth, Numer. Heat Trans., Part B: Fund. 35 (1999) 461–484.
[33] B.T. Murray, W.J. Boettinger, G.B. McFadden, A.A. Wheeler, Computation of dendritic solidification using a phase-field model, ASME Heat Trans. Melting, Solidification Cryst. Growth (1993) 67–76.
[34] B.T. Murray, A.A. Wheeler, M.E. Glicksman, Simulations of experimentally observed dendritic growth behavior using a phase-field model, J. Cryst. Growth 154 (1995) 386–400.
[35] O. Penrose, P. Fife, Thermodynamically consistent models of phase-field type for the kinetics of phase transitions, Physica D (1990) 44–62.
[36] Mathis Plapp, Alain Karma, Multiscale finite-difference-diffusion-Monte-Carlo method for simulating dendritic solidification, J. Comput. Phys. 165 (2) (2000) 592–619.
[37] D. Potts, G. Steidl, M. Tasche, Fast Fourier transforms for nonequispaced data: a tutorial, in: J.J. Benedetto, P. Ferreira (Eds.), Modern Sampling Theory: Mathematics and Applications, Birkhauser, Boston, 2001, pp. 249–274.
[38] N. Provatas, N. Goldenfeld, J. Dantzig, J.C. LaCombe, A. Lupulescu, M. Koss, M. Glicksman, R. Almgren, Cross-over scaling in dendritic evolution at low undercooling, Phys. Rev. Lett. (1999).
[39] N. Provatas, M. Greenwood, B. Athreya, N. Goldenfeld, J. Dantzig, Multiscale modeling of solidification: phase-field methods to adaptive mesh refinement, Int. J. Mod. Phys. B 19 (31) (2005) 4525–4565.
[40] Nikolas Provatas, Nigel Goldenfeld, Jonathan Dantzig, Efficient computation of dendritic microstructures using adaptive mesh refinement, Phys. Rev. Lett. 80 (15) (1998) 3308–3311.
[41] Nikolas Provatas, Nigel Goldenfeld, Jonathan Dantzig, Adaptive mesh refinement computation of solidification microstructures using dynamic data structures, J. Comput. Phys. 148 (1) (1999) 265–290.
[42] J. Rosam, P.K. Jimack, A. Mullis, A fully implicit, fully adaptive time and space discretisation method for phase-field simulation of binary alloy solidification, J. Comput. Phys. 225 (2) (2007) 1271–1287.
[43] J.A. Sethian, J. Strain, Crystal growth and dendritic solidification, J. Comput. Phys. 98 (2) (1992) 231–253.
[44] B.P. Sommeijer, L.F. Shampine, J.G. Verwer, Rkc: an explicit solver for parabolic pdes, J. Comput. Appl. Math. 88 (2) (1998) 315–326.
[45] R. Tonhardt, G. Amberg, Phase-field simulation of dendritic growth in a shear flow, J. Cryst. Growth 194 (1998) 406–425.
[46] S.L. Wang, R.F. Sekerka, Algorithms for phase field computation of the dendritic operating state at large supercoolings, J. Comput. Phys. 127 (1996) 110–117.
[47] S.L. Wang, R.F. Sekerka, A.A. Wheeler, B.T. Murray, S.R. Coriell, R.J. Braun, G.B. McFadden, Thermodynamically-consistent phase-field models for solidification, Physica D (1993) 189–200.
[48] A.A. Wheeler, B.T. Murray, R.J. Schaefer, Computation of dendrites using a phase field model, Physica D (1993) 243–262.