Available online at www.sciencedirect.com
ScienceDirect
Journal of Computational Physics 225 (2007) 1271–1287
JOURNAL OF COMPUTATIONAL PHYSICS
www.elsevier.com/locate/jcp

A fully implicit, fully adaptive time and space discretisation method for phase-field simulation of binary alloy solidification

J. Rosam [a,b,*], P.K. Jimack [a], A. Mullis [b]

[a] University of Leeds, School of Computing, Leeds LS2 9JT, UK
[b] University of Leeds, Institute of Materials Research, Leeds LS2 9JT, UK

Received 20 October 2006; received in revised form 22 January 2007; accepted 24 January 2007
Available online 6 February 2007

Abstract

A fully implicit numerical method based upon adaptively refined meshes for the simulation of binary alloy solidification in 2D is presented. In addition we combine a second-order fully implicit time discretisation scheme with variable step size control to obtain an adaptive time and space discretisation method. The superiority of this method, compared to widely used fully explicit methods, with respect to CPU time and accuracy, is shown. Due to the high nonlinearity of the governing equations a robust and fast solver for systems of nonlinear algebraic equations is needed to solve the intermediate approximations per time step. We use a nonlinear multigrid solver which shows almost *h*-independent convergence behaviour.

PACS: 81.30.Fb; 02.70.−c; 02.60.Cb

Keywords: Phase-field simulation; Binary alloys; Mesh adaptivity; Fully implicit method; Nonlinear multigrid; Variable time step control

1. Introduction

The modelling of solidification microstructures has become an area of intense interest in recent years (e.g. [1–5]), especially the evolution of microstructure and segregation patterns during the solidification of alloys. In order to model and simulate crystal growth in alloys the phase-field method is one of the most popular and powerful techniques (e.g. [6–8]). However, the nature of the phase-field models leads to coupled systems of highly nonlinear and unsteady partial differential equations (PDEs). Typically, this complexity has led

* Corresponding author. Address: University of Leeds, Institute of Materials Research, Leeds LS2 9JT, UK.
E-mail addresses: j.rosam04@leeds.ac.uk (J. Rosam), pkj@comp.leeds.ac.uk (P.K. Jimack), a.m.mullis@leeds.ac.uk (A. Mullis).

0021-9991/$ - see front matter © 2007 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2007.01.027

modellers to rely primarily on relatively simple numerical methods, however in this work we aim to demonstrate that it is possible, and indeed advantageous, to make use of advanced numerical methods, such as adaptivity, implicit schemes and multigrid.

For phase-field models, in which the phase variable, ϕ, is constant in the two phases and only varies in the thin interface region, the use of mesh adaptivity is a natural choice. Adaptive mesh refinement was applied to phase-field models for pure materials solidification, e.g. [9–13], and has subsequently also been used for models of binary alloy solidification, e.g. [14–16]. This method leads to very fine mesh resolution only in the interface region and therefore allows the use of large domains to prevent boundary effects. Another important, and related, factor is the choice of a suitable time integration method. Widely used methods are explicit methods such as Euler's method (e.g. [2,3,6,8]). However, when using explicit methods a major constraint in the computation is the time-step restriction in order to assure the stability of the scheme. Implicit methods are more expensive per step than explicit ones because intermediate approximations have to be solved from a system of nonlinear algebraic equations. However, implicit methods (e.g. [13,15]) are important because of their superior stability properties, which allow larger time steps. Another class of integration schemes are semi-implicit schemes which have been used before for pure material phase-field models where the nonlinear phase equation is solved explicitly and the linear diffusion equation is solved implicitly, see [11].

In this work we use the A-stable implicit second-order Backward Differentiation Formula (BDF2) [17] for both nonlinear equations, which is combined with variable step size selection, to obtain an adaptive time and space method. Especially for the simulation of dendritic growth, variable time-stepping is valuable because of the variation in the interface velocity over time. Explicit schemes are not generally able to exploit this since the step size selected is typically the maximum stable time step: and when mesh adaptivity is used this can be very small.

Here we demonstrate the advantages of the implicit method by considering the isothermal case of the coupled heat and solute phase-field model of Ramirez et al. [3]. These authors propose that the results of this model are independent of the interface width, thus making this model especially attractive. This model is an extension of the phase-field model for pure materials [18] and binary alloys [8]. The model is described briefly in the next section before we describe, in Section 3, the proposed discretisation methods in detail. In order for the implicit time-stepping scheme to be viable it is essential that the large systems of nonlinear algebraic equations, that occur at each time step, are solved as efficiently as possible. In order to achieve this a nonlinear multigrid solver, based upon [21], has been implemented. This is demonstrated to behave almost optimally on both uniformly and locally refined grids. Finally we compare our proposed method to other discretisation methods with respect to CPU time and accuracy by comparing total time and interface positions as well as tip velocities. Some typical simulation results are also included.

2. Phase-field model

The phase-field model used here is a variation of the coupled thermal-solute model for the simulation of microstructure formation in dilute binary alloys, given in [3]. In this paper we only consider the isothermal case in which the model reduces to a pure solute model by fixing the thermal undercooling and choosing an infinitely large Lewis number. The authors in [3] showed that the simulation results for the isothermal case agree exactly with those results found by using the model given in [8]. The microstructure is represented by the phase variable ϕ which divides the liquid and the solid phase by a diffuse interface. The solid and liquid phases correspond to $\phi = 1$ and $\phi = -1$ respectively, and in the interface region ϕ varies smoothly between the bulk values. The governing equations, in dimensionless forms for vanishing kinetic effects [3], are

$$A(\psi)^2\frac{\partial\phi}{\partial t} = A^2(\psi)\nabla^2\phi + 2A(\psi)A'(\psi)\left[\frac{\partial\psi}{\partial x}\frac{\partial\phi}{\partial x}+\frac{\partial\psi}{\partial y}\frac{\partial\phi}{\partial y}\right] - \frac{\partial}{\partial x}\left(A(\psi)A'(\psi)\frac{\partial\phi}{\partial y}\right) + \frac{\partial}{\partial y}\left(A(\psi)A'(\psi)\frac{\partial\phi}{\partial x}\right) + \phi - \phi^3 - \lambda(1-\phi^2)^2(\theta_{\text{fix}} + Mc_\infty U), \tag{1}$$

$$\left(\frac{1+k}{2}-\frac{1-k}{2}\phi\right)\frac{\partial U}{\partial t}=\widetilde{D}\left(-\frac{1}{2}\left[\frac{\partial\phi}{\partial x}\frac{\partial U}{\partial x}+\frac{\partial\phi}{\partial y}\frac{\partial U}{\partial y}\right]+\frac{1-\phi}{2}\nabla^2 U\right)+\frac{1}{2\sqrt{2}}\left(\{1+(1-k)U\}\left(\frac{\partial}{\partial x}\left(\frac{\partial\phi}{\partial t}\frac{\phi_x}{|\nabla\phi|}\right)+\frac{\partial}{\partial y}\left(\frac{\partial\phi}{\partial t}\frac{\phi_y}{|\nabla\phi|}\right)\right)+(1-k)\left(\frac{\partial U}{\partial x}\left(\frac{\partial\phi}{\partial t}\frac{\phi_x}{|\nabla\phi|}\right)+\frac{\partial U}{\partial y}\left(\frac{\partial\phi}{\partial t}\frac{\phi_y}{|\nabla\phi|}\right)\right)\right)+\frac{1}{2}\left((1+(1-k)U)\frac{\partial\phi}{\partial t}\right), \tag{2}$$

where $\psi=\arctan(\phi_y/\phi_x)$ is the angle between the normal to the interface and the x-axis, $A(\psi)=1+\epsilon\cos\eta\psi$ is an anisotropy function with anisotropy strength ϵ and mode number η. The dimensionless coupling parameter is given as

$$\lambda=\frac{\widetilde{D}}{a_2}=\frac{a_1 W_0}{d_0}, \tag{3}$$

with the chemical capillary length d_0. Also, $a_1=5\sqrt{2}/8$ and $a_2=0.6267$ [18] to simulate the kinetic free growth with the dimensional solute diffusivity $\widetilde{D}=D\tau_0/W_0^2$, where $\tau_0=(d_0^2/D)a_2\lambda^3/a_1^2$ is a relaxation time and $W_0=d_0\lambda/a_1$ is a measure of the interface width [3]. The dimensionless concentration field U is given as

$$U=\frac{\left(\frac{2c/c_\infty}{1+k-(1-k)\phi}\right)}{1-k}, \tag{4}$$

where c_∞ is the value of the concentration c far from the interface and k is the partition coefficient. The far-field concentration c_∞ and the equilibrium liquidus concentration at system temperature, c_1^0, are related via the imposed solutal undercooling as

$$\Omega=\frac{c_1^0-c_\infty}{(1-k)c_1^0}. \tag{5}$$

In order to compare our simulation results with results given in [3], the scaled magnitude of the liquidus slope is given as $Mc_\infty=1-(1-k)\Omega$ and the fixed undercooling as $\theta_{\text{fix}}=-Mc_\infty\frac{\Omega}{1-/(1-k)\Omega}$. The system parameters are set to $\Omega=0.55$, $k=0.15$, $W_0=\tau_0=1$, $\epsilon=0.02$ and $\eta=4.0$.

The highly nonlinear nature of these two time-dependent PDEs is clearly apparent due to the anisotropy terms in the phase equation (1) and the solute trapping term [8] in the concentration equation (2), respectively.

3. Numerical methods

Due to the nature of the phase-field method, where the variables may change only in a small region relative to the computational domain, adaptive mesh refinement is a natural choice and leads to a computationally efficient method. We discretize the governing equations with a finite difference approximation based upon a quadrilateral, non-uniform, refined mesh with equal grid spacing on each level in both directions. The equal grid spacing is necessary in order to apply standard finite difference stencils. The adapted meshes are non-conforming in the sense that we allow hanging nodes [21]. We distinguish between four different types of node, see Fig. 1; internal nodes ○, boundary nodes □, hanging nodes ⊙ and interface nodes ●.

In the case of a uniformly refined mesh all nodes are either internal nodes or boundary nodes. In the case of a non-uniformly refined mesh the nodes that lie at the interface of two levels of refinement are termed as either

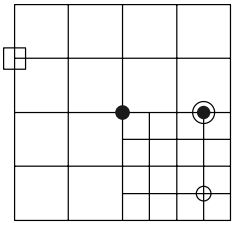


Fig. 1. Different type of grid nodes.

interface nodes or hanging nodes. Interface nodes are nodes which also exist on the next coarser grid and hanging nodes are nodes which only exist on the finer grid. This distinction is important for the understanding of the algorithms that follow.

3.1. Spatial discretisation

For all of the computational results presented in this paper second-order finite difference schemes have been used. Compact schemes are used for the phase equation in order to reduce the mesh anisotropy influence [19,20]. The mesh data is stored in a quadtree data structure, as in [11,13]. Additional to the information stored in the node list and the element tree, see [13], we also hold for each node a link to their neighbour nodes in order to facilitate the efficient application of different, and especially higher-order, finite difference stencils. Important for the stability of the numerical method is the fact that the interface is always in the refined region. To ensure this, adaptive refinement is used based upon the elementwise gradient criterion

$$E = Ch_{\mathrm{lvl}}(|\nabla\phi| + E_{\mathrm{c}}|\nabla U|), \tag{6}$$

where h_{lvl} is the element size on the actual refinement level and C, as well as E_{c}, are two user defined parameters, see [12]. E_{c} should be greater than zero to guarantee an appropriate representation of the concentration field and not only the phase field. We found that a value for E_{c} of between 0.5 and 0.75 is suitable, especially in comparison to results produced on uniform refined meshes. Two different meshes are shown in Fig. 2. On the left-hand side we show a quarter of the domain with $E_{\mathrm{c}} = 0.25$ and on the right-hand side a quarter of the domain with $E_{\mathrm{c}} = 1.0$. The mesh on the right-hand side shows much more refinement on the levels below the finest level, and this is caused by the greater influence of the concentration field. The parameter C is a more global parameter and an increase would lead to more refinement on all levels.

3.2. Time discretisation

A widely used choice, see [3,8] for example, for temporal discretisation of phase-field models such as Eqs. (1) and (2) are explicit methods such as the forward Euler scheme. If we rewrite Eqs. (1) and (2) in operator form

$$\frac{\partial\phi}{\partial t} = F_\phi(t,\phi,U), \quad \frac{\partial U}{\partial t} = F_U\left(t,U,\phi,\frac{\partial\phi}{\partial t}\right), \tag{7}$$

where F_ϕ and F_U are nonlinear differential operators, then the explicit Euler method has the following form

$$\phi^{k+1} - \phi^k = \Delta t F_\phi(t^k,\phi^k,U^k), \tag{8}$$

$$U^{k+1} - U^k = \Delta t F_U\left(t^k,U^k,\phi^k,\frac{\partial\phi}{\partial t}\right) \tag{9}$$

for $k = 0,1,2,\ldots$.

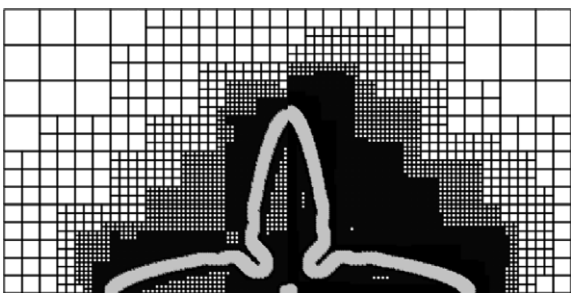


Fig. 2. Adaptive meshes after $t = 2000$ for $C = 1/2$ and left $E_{\mathrm{c}} = 0.25$ and right $E_{\mathrm{c}} = 1.00$, the finest mesh is shaded grey.

The implementation of the explicit Euler method based upon uniform grids is very straightforward, but for adaptively refined meshes their exist a number of possibilities.

Algorithm 1. Explicit Euler method for adaptively refined meshes

1. Go to the finest uniform refined level
2. Solve Eqs. (8)–(9) for all internal nodes
3. Set up the values on the internal interface nodes of the next finer level by interpolating the values from the coarser mesh
4. Go up to the next finer level and solve (8)–(9) for all internal nodes
5. IF the finest level has been reached then STOP else GOTO 3

Algorithm 1 shows the implementation used in this work for locally refined spatial meshes. The key point in any such algorithm is the handling of the internal interfaces. The internal interfaces are treated as a Dirichlet boundary for the finer level, with these values obtained by interpolating from the coarser level.

The finer level solution is then obtained only at the internal points. Simple injection is used for the interpolation of the values at the interface nodes from the coarser level, however cubic interpolation is used to obtain Dirichlet values at the hanging nodes. This higher-order interpolation is especially needed for the concentration field which is not linear in the internal interface regions. An interesting observation is that to obtain a given accuracy more refinement is needed for the explicit method than for the implicit method, even when higher-order interpolation is used at the hanging nodes. The reason is that in the nonlinear multigrid solver, which is described later in Section 3.4, the hanging nodes are updated at each cycle and the convergence is therefore guaranteed at these nodes.

As already mentioned, the explicit methods suffer from the following time step restriction

$$\Delta t \leqslant \delta h^2 \tag{10}$$

for some constant δ, where Δt is the time step and h is the minimum element size. This condition is necessary in order to ensure the stability of the discretisation scheme, and for some nonlinear systems the constant δ can be very small, thus leading to excessively small time steps. That is, the time steps are so small that the temporal error is substantially less than the spatial truncation errors.

Table 1 shows statistics for the maximum stable time step size for different meshes when using the explicit Euler method. With this information the constant δ in (10) may be approximated as about 0.13 to solve (1), (2). Note however that the maximum stable time step depends not only on the mesh size but also on the model parameters. The predicted maximum time step sizes shown in this table are computed with the same set of model parameters as used for the majority of calculations in this paper.

In order to overcome this restriction the use of implicit time integration methods is proposed in this paper. These methods may be designed to be unconditionally stable, which means that the time step size does not depend on the space step size in order to ensure stability. Our interest is in finding an optimal scheme for which it is possible to set $\Delta t = \delta h$. The second-order Backward Difference Formula (BDF2), combined with the described spatial discretisation, would lead to a second-order time and space method and so fulfil the desired criterion. This is not true for second-order explicit time integration methods, such as Runge–Kutta or the trapezoidal or midpoint rules, see [17], because the stability of these methods are also only preserved by the condition (10). Other classes of implicit higher-order time methods can be found for example in [17] or [26].

Table 1
Maximum stable time step size when using the explicit Euler method on different grid levels

Mesh size (h)	Explicit Euler method
0.781	0.079–0.080
0.391	0.019–0.020
0.195	0.004–0.005

The BDF2 method is an implicit linear 2-step method which takes the following form when solving (7):

$$\frac{1}{2}\phi^{k-1} - 2\phi^{k} + \frac{3}{2}\phi^{k+1} = \Delta t F_{\phi}(t^{k+1}, \phi^{k+1}, U^{k}), \tag{11}$$

$$\frac{1}{2}U^{k-1} - 2U^{k} + \frac{3}{2}U^{k+1} = \Delta t F_{U}\left(t^{k+1}, U^{k+1}, \phi^{k+1}, \frac{\partial\phi}{\partial t}\right) \tag{12}$$

for $k > 1$. The first-order implicit Euler method is typically used for the first time step ($k = 1$). It can be shown that the BDF2 method is A stable, see [17], and is therefore widely used for stiff systems of differential equations, for example to simulate chemical reactions or biological phenomena. The advantage over one-step second-order methods, such as the Crank Nicolson scheme is that only one nonlinear solve is required at each time step. The small price that has to be paid for this computational efficiency is that the solutions from the previous two time steps must be saved.

Fig. 3 shows a convergence study of the tip position at a fixed time, $t = 10.0$, for the explicit Euler method and the implicit BDF2 method for decreasing constant time step sizes. The computations are done on uniformly refined grids of dimension $[-100, 100]^2$, with an element size of $h = 0.39$ and the initial seed radius is chosen as $R_0 = 44d_0 \approx 12.1865$. Due to the stability restriction, the largest possible time step for the explicit method on this grid is $\Delta t = 0.01$, whereas for the BDF2 method the time step size can be chosen, theoretically, to be arbitrarily large. In practise there is a restriction on the maximum step size for the implicit scheme, either due to non-convergence of the nonlinear algebraic solver or simply due to the size of temporal error. In Fig. 3 the BDF2 time step is restricted by $\Delta t \leqslant 0.5$ due to the high nonlinearity of the problem and the choice of model parameters. It is very clear however that the BDF2 method converges with significantly larger time steps than the explicit Euler method and can provide comparable accuracy with much larger Δt.

In addition to Fig. 3, Table 2 shows the time steps for which the position of the interface has the same accuracy for the explicit and the BDF2 methods. As one can see, the BDF2 method allows Δt to be up to 80 times larger, for the same accuracy, than the explicit method for this example.

3.3. Variable step size control

The initial conditions typically considered for this problem consist of a small region of solid at the centre of the domain, known as the nucleus. The growth velocity of this initial nucleus is very high at the beginning of the simulation, before the interface becomes unstable and dendritic arms begin to grow, ultimately reaching a steady-state velocity. Consequently, an adaption of the time steps for the BDF2 method is likely to be efficient and leads to an adaptive time and space discretisation method. The adaptive time-stepping algorithm used in

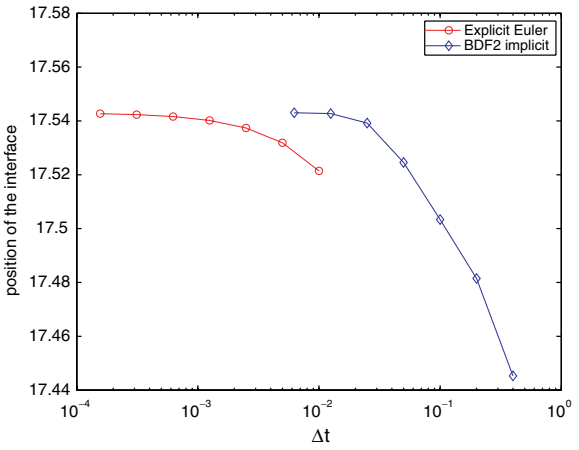


Fig. 3. Convergence study of the position of the interface for different time discretisation methods on uniform grids of element size $h = 0.39$ and at time $t = 10.0$.

Table 2
Comparison of the time step sizes for which the interface positions are the same for both methods on a uniform spatial grid with an element size of $h = 0.39$ and at a final time $t = 10.0$

Explicit method		BDF2 implicit method	
Δt	Position of the interface	Δt	Position of the interface
0.01	17.521443	0.05	17.524558
0.00125	17.540187	0.025	17.539207
0.00015625	17.542688	0.0125	17.542740

this paper is based upon the following rule: if the estimated local temporal error $D_k \leqslant Tol$ the time step is accepted and the next time step size is increased, whereas if $D_k > Tol$ the step is rejected and retaken with a smaller time step. Let

$$r = \left(\frac{Tol}{D_k}\right)^{1/(p+1)}, \tag{13}$$

where p is the order of the time discretisation scheme ($p = 2$ for the BDF2 method and, in the first time step $p = 1$ for the implicit Euler method). Then the new time step size Δt_{new} is given by

$$\Delta t_{\text{new}} = \min(r_{\max}, \max(r_{\min}, \vartheta r))\Delta t_{\text{old}}, \tag{14}$$

where the minimal and the maximal time step size growth factor are $r_{\min}$ and $r_{\max}$ respectively, and ϑ is a safety factor, see [17]. In all computations used in this paper the variables are set to $r_{\min} = 0.5$, $r_{\max} = 2.0$ and $\vartheta = 0.8$.

The local error estimate is obtained by comparing the solution of the BDF2 method and the solution obtained by using a first-order method. (In the first time step the local error is estimated by comparing the solution ϕ^1_{im} using the implicit Euler method with the solution of the explicit Euler method $\phi^1_{\text{ex}} = \phi^0 + \Delta t_0 F_\phi(\phi^0, U^0)$: the local error estimator is given then as $D^0 = \frac{1}{2}\|(\phi^1_{\text{im}} - \phi^1_{\text{ex}})\|_\infty$). In this work the implicit Euler method is used for the first-order scheme, as derived in [17], leading to the following estimate:

$$D_k = \frac{r}{1+r}\|\phi^{k+1} - (1+r)\phi^k + r\phi^{k-1}\|_\infty, \tag{15}$$

where r is the steps size ratio $\Delta t_k/\Delta t_{k-1}$. Tests show that for Eqs. (1) and (2) it is sufficient to base the time step control only on the phase variable due to the fact that the two equations are of the same type. This is different to most thermal models for the simulation of pure material solidification, see [18], where the temperature equation and the phase equation are of a different type, with different requirements on the time step size. To overcome this difficulty the authors in [11] use a second-order time discretisation scheme for the temperature equation and a first-order scheme for the phase equation.

Fig. 4 illustrates the progression of the time step size for $t = 0 \ldots 2000$, for different tolerances Tol in (13), on meshes with spacing of $h = 0.39$. One can see that a very small time step size is used right at the beginning but that this increases rapidly over time and converges to a constant value which depends upon the choice of Tol. In the figure the maximum stable time step size is also shown for the explicit Euler method. Compared to the final step size of the BDF2 method with $Tol = 1.2\text{e}{-2}$ the step size of the explicit Euler method is, for $h = 0.39$, 45 times smaller.

3.4. Nonlinear multigrid solver

When using implicit time discretisation methods it is necessary to solve a system of nonlinear algebraic equations at each time step. Multigrid methods are among the fastest available solvers for large sparse systems of linear or nonlinear algebraic equations and are based upon two principles; the coarse grid principle and the smoothing principle, see for example [21–23]. For the coarse grid correction one has to define grid transfer operators to transfer the solution and the residual from the fine to the coarse grid, and the solution from the coarse to the fine grid. In the examples given here bilinear interpolation is used for the coarse to fine grid

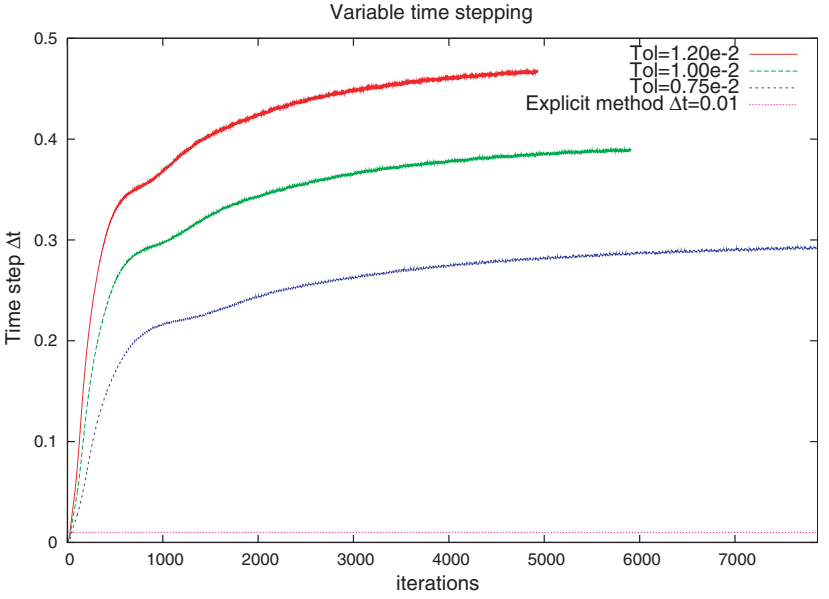


Fig. 4. The evolution of the time step size for $t = 0 \ldots 2000$, and a minimal element size of $h = 0.39$, for different tolerances and in comparison to the constant step size $\Delta t = 0.01$ for the explicit method.

transfers and injection is used for the fine to coarse grid transfers. For the smoothing principle a basic iteration method for smoothing the error is used. One of the simplest possibilities is a pointwise nonlinear weighted Jacobi smoother, which is used here for the phase equation:

$$\phi_{ij}^{k+1,n+1} = \phi_{ij}^{k+1,n} - \omega \frac{\left(F_\phi^\star(\phi_{ij}^{k+1,n}, U_{ij}^k) - \left(\phi_j^k + \frac{1}{4}\phi^{k-1}\right)\right)}{\frac{\partial}{\partial \phi_{ij}} F_\phi^\star\left(\phi_{ij}^{k+1,n}, U_{ij}^k\right)}, \tag{16}$$

where

$$F_\phi^\star(\phi^{k+1}, U^k) = -\frac{\Delta t}{2} F_\phi(t^{k+1}, \phi^{k+1}, U^k) + \frac{3}{4}\phi^{k+1}, \tag{17}$$

which follows from (11). The advantage of using a Jacobi smoother is that ψ in (1) has to be calculated only once per iteration. For the concentration equation a pointwise nonlinear weighted Gauss–Seidel smoother is used:

$$U_{ij}^{k+1,n+1} = U_{ij}^{k+1,n+1} - \omega \frac{F_U^\star\left(U_{ij}^{k+1,n+1}, \phi_{ij}^{k+1}, \frac{\partial \phi_{ij}^{k+1}}{\partial t}\right) - \left(U_j^k + \frac{1}{4}U^{k-1}\right)}{\frac{\partial}{\partial U_{ij}} F_U^\star\left(U_{ij}^{k+1,n+1}, \phi_{ij}^{k+1}, \frac{\partial \phi_{ij}^{k+1}}{\partial t}\right)}, \tag{18}$$

where

$$F_U^\star\left(U^{k+1}, \phi^{k+1}, \frac{\partial \phi^{k+1}}{\partial t}\right) = -\frac{\Delta t}{2} F_U\left(t^{k+1}, U^{k+1}, \phi^{k+1}, \frac{\partial \phi^{k+1}}{\partial t}\right) + \frac{3}{4}U^{k+1}. \tag{19}$$

For both smoothers the derivatives of the discretisation operators with respect to the system variable at each point is needed. In order to simplify these derivatives, central difference schemes are used to approximate the first and second derivatives in both directions, so that the derivative is zero, e.g.

$$\frac{\partial}{\partial \phi_{ij}}\left(\left[\frac{\partial \phi}{\partial x}\right]_{ij}\right) = 0 \quad \text{with} \quad \left[\frac{\partial \phi}{\partial x}\right]_{i,j} = \frac{1}{2h}(\phi_{i+1,j} - \phi_{i-1,j}).$$

The derivative of the right-hand side of (17) with respect to ϕ_{ij} is therefore given as

$$\frac{\partial F^{\star}_{\phi}(\phi,U)}{\partial \phi_{ij}} \approx \frac{-\Delta t}{2A^2(\psi_{ij})}\left\{A(\psi_{ij})^2\frac{\partial}{\partial \phi_{ij}}(\nabla_h^2\phi_{ij}) + 1 - 3\phi_{ij}^2 - 2\lambda(\theta_{\text{fix}} + Mc_{\infty}U_{ij})(-\phi_{ij}+2\phi_{ij}^3)\right.$$
$$\left. +(A'^2(\psi_{ij}) + A(\psi_{ij})A''(\psi_{ij}))\frac{1}{|\nabla_h\phi_{ij}|^2}\left\{\phi_x^2\frac{\partial\phi_{yy}}{\partial\phi_{ij}} + \phi_y^2\frac{\partial\phi_{xx}}{\partial\phi_{ij}} - 2\phi_x\phi_y\frac{\partial\phi_{xy}}{\partial\phi_{ij}}\right\}\right\} + \frac{3}{4},$$

where ∇_h^2 is the discrete Laplace operator, in this case the second-order compact nine point stencil, see for example [27]. The same procedure applied to the right hand-side of (19) gives

$$\frac{\partial F^{\star}_{U}(\phi,U)}{\partial U_{ij}} \approx \frac{-\Delta t}{1+k-(1-k)\phi_{ij}}\left\{D\left(\frac{1-\phi_{ij}}{2}\left(\frac{\partial}{\partial U_{ij}}(\nabla_h^2 U_{ij})\right)\right) + \frac{1}{2\sqrt{2}}\left((1-k)\left(\frac{\partial}{\partial x}\left(\frac{\partial\phi}{\partial t}\frac{\phi_x}{|\nabla_h\phi_{ij}|}\right)\right.\right.\right.$$
$$\left.\left.\left. + \frac{\partial}{\partial y}\left(\frac{\partial\phi}{\partial t}\frac{\phi_y}{|\nabla_h\phi_{ij}|}\right)\right)\right) + \frac{1}{2}(1-k)\frac{\partial\phi}{\partial t}\right\} + \frac{3}{4},$$

where, for example, ϕ_x is the notation for the first derivative $\frac{\partial\phi_{ij}}{\partial x}$.

On the basis of the described smoothers and transfer operators a multigrid solver for adaptive refined meshes has been developed based upon the Full Approximation Scheme (FAS) for resolving the nonlinearity, see [24], and the adaptive multigrid approach of [21]. Another multigrid method for local refined meshes is the Fast Adaptive Composite Grid method which is described in [23,25]. Note that although the smoothers (16) and (18) have been written separately, the nonlinear system that is solved is a single system for all unknowns ϕ_{ij}^{k+1} and U_{ij}^{k+1}. The number of pre- and post-smooths applied is typically two, however other alternatives are presented in Table 3. Note that the multigrid convergence rate depends on a number of factors, including: the transfer operators; the smoother; the number of post- and pre-smooths; and also on the iteration form. Table 3 shows convergence rates for different iteration forms and different pre- and post-smoothing steps at a fixed time and a constant time step size of $\Delta t = 0.05$ on uniform grids with size $h = 0.78$. The notation $V(2,1)$, for example, represents a V-cycle with 1 post- and 2 pre-smoothing steps. The convergence rate is calculated by iterating until a residual of less than 1e−10 is reached and then the proportion of the residual of ϕ at the penultimate and last steps is calculated, measured in the infinity-norm. The number of iterations needed to reach a residual of less than 1e−10 is equal to the number of cycles in the Table. As one can see the V-cycle form with 2 post and 2 pre smoothing steps performed best in terms of convergence rate, number of cycles and execution time.

A major property of the multigrid method is the h-independent convergence, which means that the convergence rate does not depend on the spatial element size. As one can see in Fig. 5(a), where the residual of ϕ is shown at a fixed time for different mesh sizes, for both uniformly and adaptively refined meshes, the convergence is of the same order and the mesh adaptivity does not affect the convergence rate even for this highly nonlinear problem.

An implication of this is that the execution time versus the number of nodes should scale linearly, and this optimal behaviour is indeed observed in Fig. 5(b).

Table 3
Statistic of the convergence rate, number of cycles and the execution time for different types of iteration form

Iteration form	Convergence rate	No. of cycles	Time (s)
$V(1,1)$	0.008789	5	2.3915
$V(2,1)$	0.000872	4	2.4667
$V(2,2)$	0.000098	3	2.3021
$W(1,1)$	0.008788	5	3.3234
$W(2,2)$	0.000097	3	3.1040

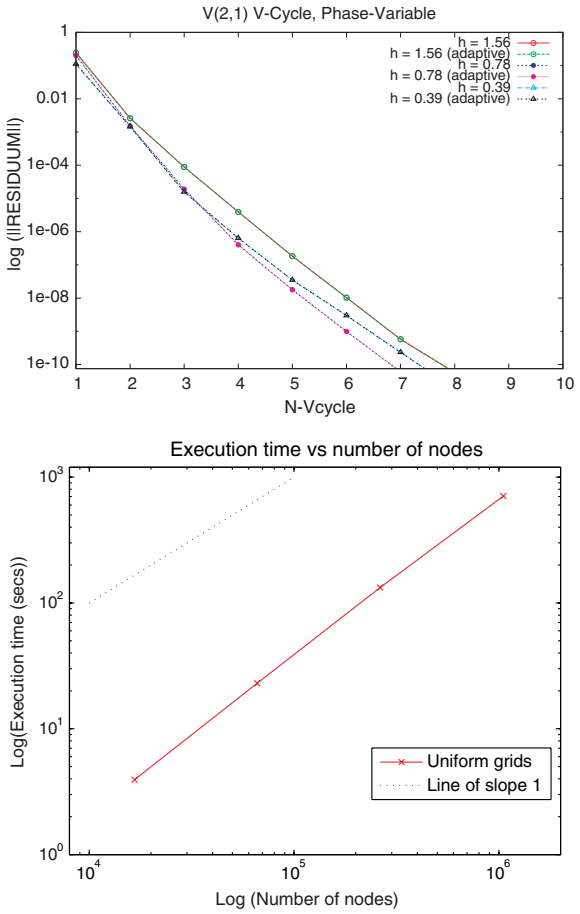


Fig. 5. (a) Residual of the phase variable at a fixed time for different mesh levels as well as uniformly and non-uniformly refined meshes; (b) shows the execution time for various system sizes.

4. Results

This section presents a selection of typical results for the solution of (1) and (2), concentrating mainly on the comparison between the explicit Euler method and the implicit BDF2 time discretisation method. The following aspects are considered:

1. The influence of the refinement on the accuracy.
2. The influence of the choice of the time step size on the accuracy.
3. The execution times of both methods.

For validation, results are compared with those presented in [8,3]. The dendritic growth simulation is undertaken with the model parameters given in Section 2. The only free parameter to choose is the coupling parameter λ, which depends on the choice of the diffusivity coefficient D, see (3). This parameter is set to $D = 2$ in these simulations, whereby it follows that $\lambda = 3.1913$ and the capillarity length $d_0 = 0.27696$. The rectangular computational domain $\mathcal{Q}$ is chosen as $\mathcal{Q} = [-400, 400]^2$ with Dirichlet boundary conditions. On the boundaries ϕ is set to be -1 and the concentration field U is considered to be zero. The phase field is initialised as $\phi = -\tanh(\beta(x^2 + y^2 - R_0^2))$, where R_0 is the radius of the initial seed and β a constant to control the steepness, and the concentration U is initialised to zero in the whole domain.

A typical simulation result for a fourfold symmetric alloy dendrite growing in an undercooled melt is shown in Fig. 6. The contour plots show, on the left-hand side, the phase variable and, on the right-hand side, the concentration field at $t = 1800$. At this time the tip velocity has reached a steady state. In those regions where the phase variable forms a very steep interface the concentration field is more slowly varying and is only very sharp in front of the tip. This illustrates the need for local mesh refinement, see Section 3 and Fig. 2. The influence that the adaptive refinement has on the simulation results is discussed in the next section.

In order to simulate a pure fourfold symmetry the radius of the initial solid seed is taken as $R_0 = 14d_0$ in all cases. Fig. 7 shows a study of how the radius of the initial seed influences the shape of the dendrite. If the initial radius is chosen to be larger than $R_0 = 28d_0$ then the dendrite no longer grows with a pure fourfold symmetry.

4.1. Adaptive remeshing

In this section we compare results obtained on uniform meshes and on adaptively refined meshes and study what influence the adaptive refinement has on different parameters for different discretisation methods. All the results shown next are for meshes with a minimum element size of $h = 0.78$. For the explicit Euler method a

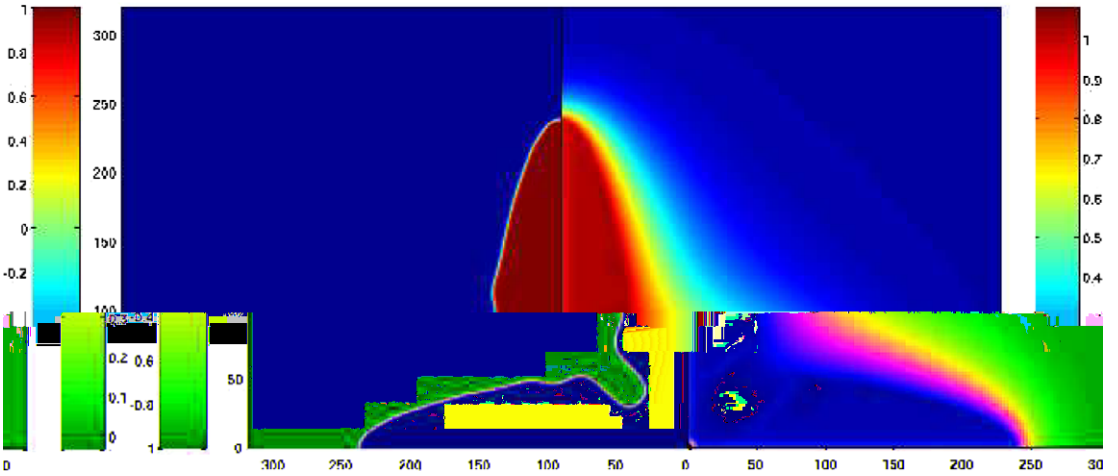


Fig. 6. The interface shape of an alloy dendrite after $t = 1800$; the left and the right box show the contours of the phase variable ϕ and the contours of the dimensionless concentration field U, respectively.

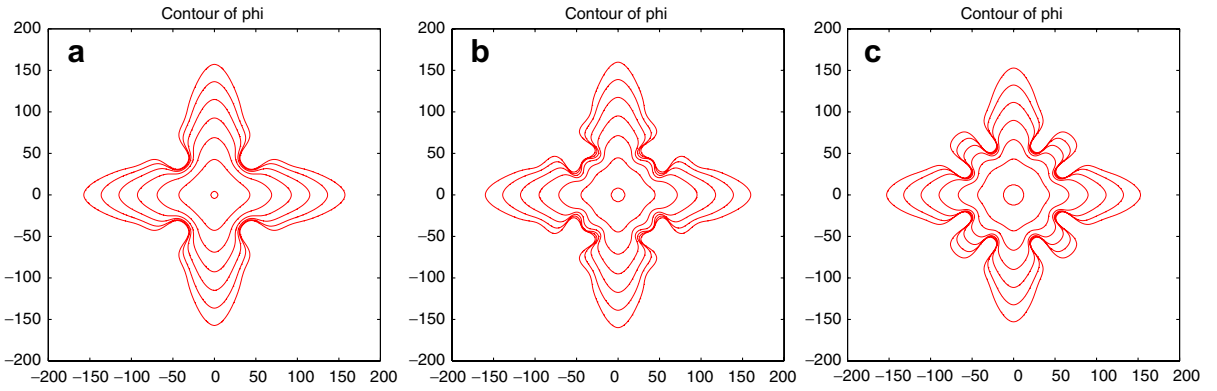


Fig. 7. The evolution of the interface for a different radius of the initial solid seed: (a) $R_0 = 14d_0$, (b) $R_0 = 28d_0$ and (c) $R_0 = 44d_0$.

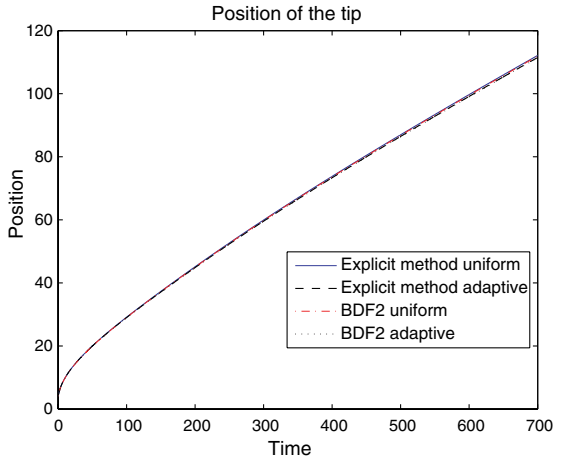


Fig. 8. Comparison of the tip position computed on uniformly and adaptively refined meshes using the explicit Euler time integration scheme and the BDF2 method.

constant step size of $\Delta t = 0.05$ was chosen, which is slightly less than the maximum stable time step, see Table 1. For the BDF2 method the error tolerance, *Tol* in (13), was fixed as 1e−2.

The first test was undertaken on a smaller domain $\mathcal{Q} = [-200, 200]^2$ to show that the adaptive refinement generally does not have an effect on the simulation results. To demonstrate the same accuracy for both uniform and adaptive refinement Fig. 8 shows the position of the tip along the x-axis as a function of time. Since the computation of the tip velocity and the curvature depends on the position of the tip it is important to demonstrate that the tip position is the same.

Fig. 8 does indeed indicate that the same results are obtained using adaptive meshes and uniform meshes for both time integration schemes. It should be noted however that the explicit scheme requires more cautious adaptivity than the BDF2 scheme in order to achieve these results. Specifically, for Fig. 8, values of $C = 2.0$, $E_c = 1.0$ were used for the adaptivity with the explicit scheme (see (6)), whereas $C = 1.0$, $E_c = 0.75$ were sufficient for the implicit time-stepping. Consequently the former yields a mesh of up to 81645 nodes whereas the latter only uses 69133 nodes. These compare to a uniform mesh of 263169 nodes. The corresponding reductions in CPU time are from 9.2 h (up to $t = 700$) to 3.9 h in the explicit case, and from 15.9 h to 3.3 h in the implicit case. Note that it is marginally faster to use the implicit method for this mesh size but for coarser meshes the explicit scheme may be preferable. For finer meshes, as shown below, the implicit scheme will provide significant further advantage.

For the rest of this paper only adaptively refined meshes will be considered. This is because it becomes excessively expensive to compute on uniform meshes as h is reduced. For example, with a minimum element size of 0.098, which is comparable to a uniform mesh with 67 million nodes, it is impossible to solve on a single workstation.

As already indicated above, the choice of the adaptive refinement parameters have an influence on the results in both methods. The explicit method is particularly sensitive to the refinement scheme due to the more complicated handling of the internal interface nodes, as discussed in Section 3.2. Even if cubic interpolation is used at the internal interfaces more refinement is needed in order to reproduce the same results as with the implicit BDF2 method, where the interface nodes are incorporated very naturally into the multigrid solver, see for example [21]. Table 4 shows a parameter study, and how the parameters influence the position and the velocity of the tip after $t = 1800$ on a domain $\mathcal{Q} = [-400, 400]^2$ with $h = 0.78$.

In Table 4 the parameter E_c is held constant and the parameter C, which is global parameter and leads to more refinement on all levels, is varied. Both methods converge to the same position but the explicit method needs more refinement than the BDF2 method. In order to compare the results for both methods the parameter values that are chosen in all later studies are: $C = 2.0$, $E_c = 1.0$ for the explicit Euler scheme, $C = 1.0$, $E_c = 0.75$ for the BDF2 method.

Table 4
Position of the tip and the tip velocity at $t = 1800$ for different refinement parameters and different time discretisation methods

Method	Parameters	Position	Velocity
Explicit Euler	$C = 1/2, E_c = 1.0$	230.92387	0.106260
	$C = 1.0, E_c = 1.0$	235.32227	0.108935
	$C = 2.0, E_c = 1.0$	236.45504	0.109486
Implicit BDF2	$C = 1/2, E_c = 0.75$	234.69686	0.109041
	$C = 1.0, E_c = 0.75$	236.50597	0.109649

4.2. Parameter studies

Before reaching the final comparison in the next section one further parameter study is undertaken, concerning the time step control and the multigrid solver tolerance for the BDF2 method. The simulation parameters are as stated in the previous section.

Table 5 shows the position of the tip and the tip velocity at $t = 1800$ for different tolerances *Tol* in the step size control. When the tolerance is small then the time steps become smaller, see Fig. 4. However, as one can see, the difference between results computed with different tolerances are marginal but the difference in the time step sizes are quite significant. For example the change of the tip position between the choice $Tol = 1.2\text{e}{-2}$ and $Tol = 0.75\text{e}{-2}$ is only 0.05% but the final time step size for $Tol = 1.2\text{e}{-2}$ is 63% larger than the final time step size when using $Tol = 0.75\text{e}{-2}$. This leads to an huge drop in the number of time steps and so in the overall execution time.

A similar conclusion can be reached by studying the dependence of the multigrid solver tolerance *Stol* on the simulation results. Table 6 shows the tip position and the tip velocity for two different solver tolerances, all other parameters held to be the same. As one can see the solver tolerance does not influence the results significantly, there is only a 0.018% difference in the position of the tip between both. Consequently, for subsequent calculations $Stol = 1\text{e}{-5}$ is chosen since the number of multigrid iterations, which depends on the chosen solver tolerance, does have a significantly impact on the total execution time.

After studying the different parameters which could have an effect on the simulation results we come now to a final comparison of the explicit and the implicit methods.

4.3. Comparison of the explicit Euler method and the implicit BDF2 method

A graphical comparison of a selection of results on adaptively refined meshes is shown in Fig. 9. The top left figure shows the position of the tip of a dendrite growing along the x-axis versus time. The top right figure is the tip velocity versus time and the bottom left graph shows the tip radius versus time. Finally the bottom right figure shows the evolution of the time step size for the BDF2 method. All simulations are evaluated until

Table 5
Position of the tip and the tip velocity at $t = 1800$ for different error tolerances in the time step control as well as the time step size

Tol	Position	Velocity	Time step
1.20e−2	234.72354	0.109029	≈0.49
1.00e−2	234.69686	0.109041	≈0.40
0.75e−2	234.84339	0.109166	≈0.30

Table 6
Position of the tip and the tip velocity for different solver tolerances after $t = 1800$

Stol	Position	Velocity
1e−7	234.74028	0.109042
1e−5	234.69686	0.109041

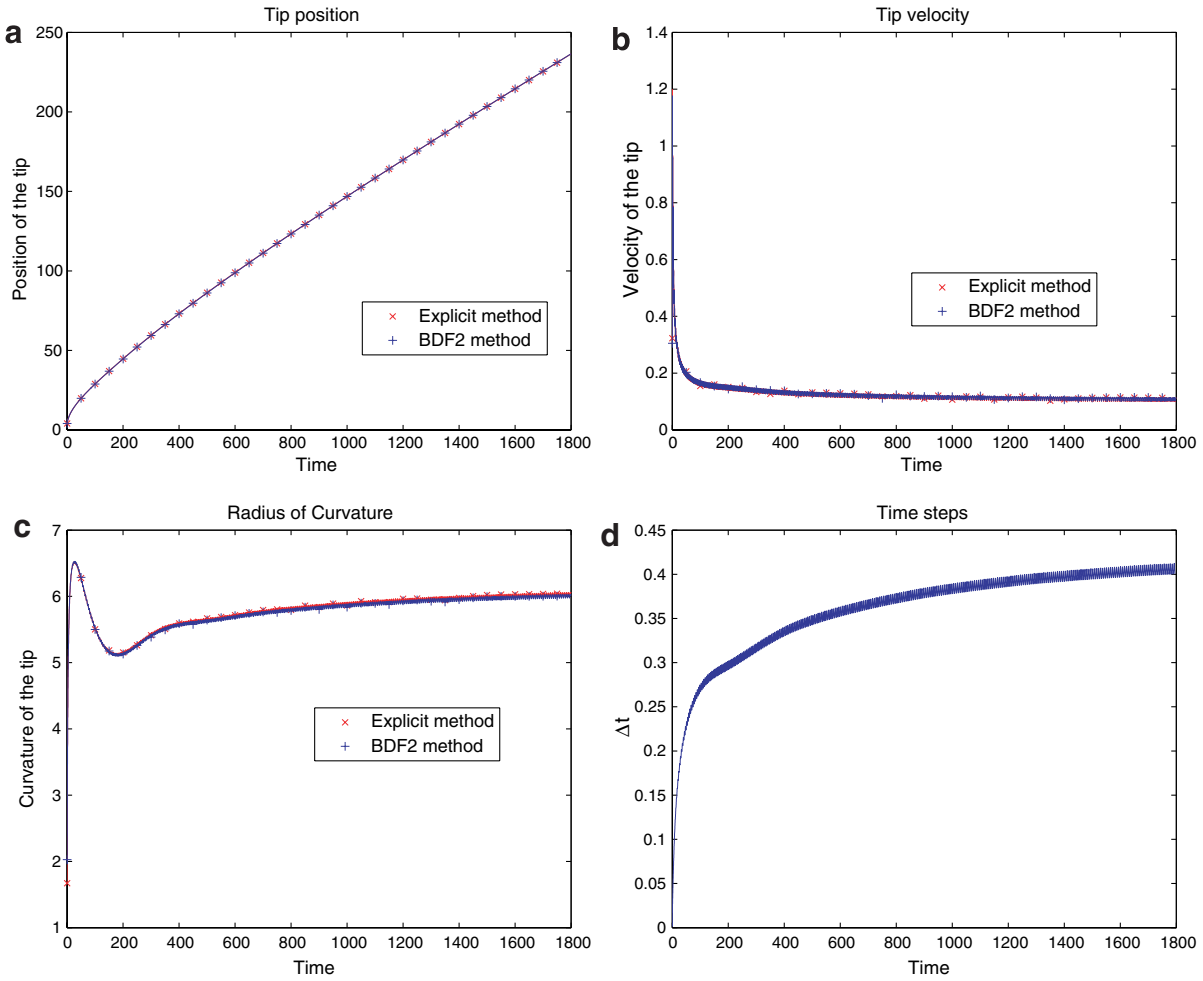


Fig. 9. (a) Position of the tip versus time, (b) velocity of the tip versus time, (c) radius of the tip curvature versus time and (d) evolution of the time steps of the BDF2 method.

$t = 1800$ where a steady-state tip velocity is reached, which is equivalent to a dimensionless time of $tD/d_0^2 \approx 47,000$. As one can see both methods produce the same results, with the respective curves lying on top of each other. This correlation is strengthened by a direct comparison of the steady-state results in Table 7. The total execution time is additionally shown in this table. From Table 5 it is known that the time step at the end of the simulation is $\Delta t \approx 0.4$ for the BDF2 scheme in comparison to the constant time step of $\Delta t = 0.05$ necessary for stability of the explicit Euler method. In order to make the time comparison as fair as possible therefore mesh refinement is only undertaken every 10 time steps with the explicit method, compared to each time step in the BDF2 method.

The results clearly demonstrate that both methods produce the same simulation results. Furthermore, the spatial mesh level with a minimum element size of $h = 0.78$ is the first level for which the implicit BDF2

Table 7
Comparison of the position, velocity and radius of the tip on the x-axis after $t = 1800$ and the total execution time

	Position	Velocity	Curvature	Total time (h)
Explicit Euler	236.45504	0.109486	6.034250	11.8
Implicit BDF2	236.50597	0.109649	6.004577	11.5

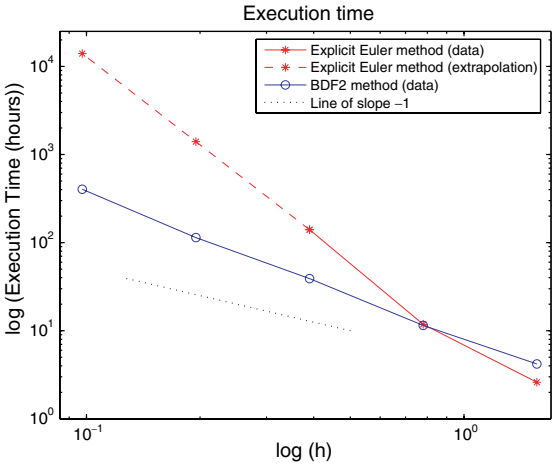
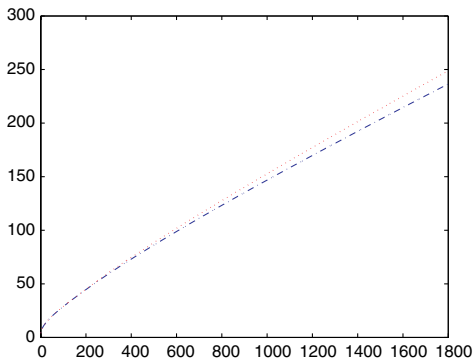


Fig. 10. The execution time for both methods and various system sizes.

method is faster than the explicit Euler method on adaptively refined meshes. However, the convergence study of the steady-state tip velocity in the next section demonstrates that a spatial step size of at least $h = 0.39$ is needed to approximate the test problem considered here with sufficient accuracy. Such a decrease of the step size has a significant impact on the total execution time for both methods but the time increase for the explicit method is much greater than for the implicit method. This is because the stability restriction for the explicit method means that one has to quarter the time step whenever the minimum element size is halved. Based on the fact that, in the adaptive meshing, the number of nodes only doubles or triples every time the minimum element size is halved, the total execution time for the explicit Euler method should go up by a factor of 8–12. However, for the BDF2 method the execution time only increases by factor of at most 4–6 because of the variable step size control.

Fig. 10 shows the execution time for both methods on meshes of minimum element size $h = 1.56, 0.78, 0.39, 0.19$ and 0.097. The times for the explicit method for $h = 0.19$ and 0.097 are extrapolated based upon an approximation of the execution times of the other step sizes $h = 0.78$ and 0.39, in order to complete the picture. It is assumed that the execution time grows by a factor of 10, although the execution time increases from $h = 078$ to $h = 0.78$ by a factor of 11.9. To complete an explicit simulation on meshes with a step size of $h = 0.097$ would require an execution time of more than 12000 h. For the same system size the BDF2 method needs a little more than 400 h, which is about 30 times less.



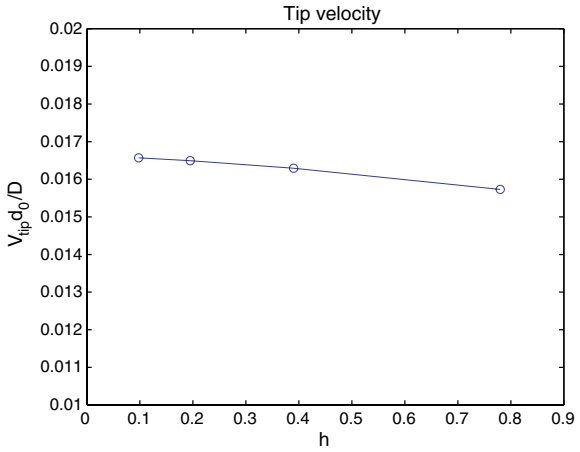


Fig. 12. Convergence study of the dimensionless tip velocity at $t = 1800$ for decreasing element sizes.

4.4. Convergence behaviour

To complete this presentation of results, this section examines the convergence behaviour of the simulations. All subsequent simulations are performed with the BDF2 method on meshes with a minimum element size of less than or equal to $h = 0.78$. Fig. 11 shows the progression of the tip position and the tip velocity over time for different maximum refinement levels. Both parameters converge as the meshes become finer. Only the results computed on meshes with a step size of $h = 0.78$ stand out at this graph resolution, thus demonstrating that a finer grid spacing is essential for accurate predictions.

A convergence study of the dimensionless steady-state tip velocity as a function of the minimum element size, is shown in Fig. 12. The computational results converge when the step size becomes sufficiently small and show a very good agreement with results published in [8,3].

5. Conclusions

This paper presents an efficient fully adaptive numerical scheme for the simulation of dendritic alloy growth in a undercooled melt in two dimensions. The phase-field model used to demonstrate the method is a variation of the coupled thermal-solute model, published in [3], for the simulation of isothermal growth. In order to solve efficiently on meshes with a very fine spatial resolution adaptive meshing and a second-order implicit time discretisation scheme are used and coupled with variable time step size control. This combination reduces the execution time drastically compared to explicit time integration methods since their is no artificial stability restriction imposed on the time step size, see Fig. 10. To solve the intermediate approximations in the implicit BDF2 method a robust multigrid solver is essential, the FAS scheme applied on the adaptive grids in this work shows excellent h-independent convergence rates.

The convergence of the steady-state tip velocity is studied and shows a very good agreement with results published in [3,8]. By using the fully implicit approach it has been possible to compute efficiently using minimum element sizes of less than 0.1. In order to achieve this accuracy on uniform meshes one would need lattice with $2^{13} \times 2^{13}$ nodes, and the use of explicit time-stepping would not be practical.

This is the first paper to couple the use of adaptivity in space and time with implicit methods and the use of multigrid solvers for the simulation of solidification using Phase-field models. Numerous other phase-field models exist and further studies may be undertaken, including the application of this numerical method to the fully coupled thermal-solute model, which exhibits diffusion effects on different time scales, thus making the potential advantages of the proposed approach even greater.

Acknowledgements

This work was supported by EPSRC Grant No. GR/T10374/01. We thank Dr. Alison Jones for providing the initial version of our mesh adaptivity routines [13].

References

[1] R. Kobayashi, Modeling and numerical simulations of dendritic crystal growth, Physica D 63 (1993) 410–423.
[2] W.J. Boettinger, J.A. Warren, C. Beckermann, A. Karma, Phase-field simulation of solidification, Annual Review of Material Research 32 (2002) 163–194, doi:10.1146/annurev.matsci.32.101901.155803.
[3] J.C. Ramirez, C. Beckermann, A. Karma, H.-J. Diepers, Phase-field modeling of binary alloy solidification with coupled thermal heat and solute diffusion, Physical Review E 69 (2004), doi:10.1103/PhysRevE.69.051607.
[4] A.M. Mullis, An extension to the Wheeler phase-field model to allow decoupling of the capillary and kinetic anisotropies, European Physics Journal B 41 (2004) 377–382, doi:10.1140/epjb/e2004-00330-7.
[5] A.M. Mullis, Quantification of mesh induced anisotropy effects in the phase-field method, Computational Materials Science 36 (2006) 345–353, doi:10.1016/j.commatsci.2005.02.017.
[6] A.A. Wheeler, W.J. Boettinger, G.B. McFadden, Phase-field model for isothermal phase transitions in binary alloys, Physical Review A 45 (1992) 7424–7440.
[7] S.G. Kim, W.T. Kim, T. Suzuki, Phase-field model for binary alloys, Physical Review E 60 (1999) 7186–7197.
[8] A. Karma, Phase-field formulation for quantitative modeling of alloy solidification, Physical Review Letters 87 (2001), doi:10.1103/PhysRevLett.87.115701.
[9] A. Schmidt, Computation of three dimensional dendrites with finite elements, Journal of Computational Physics 125 (1996).
[10] R.J. Braun, B.T. Murray, Adaptive phase-field computations of dendritic crystal growth, Journal of Crystal Growth 174 (1997) 41–53.
[11] N. Provatas, N. Goldenfeld, J. Dantzig, Adaptive mesh refinement computation of solidification microstructures using dynamic data structures, Journal of Computational Physics 148 (1999) 265–290.
[12] B. Nestler, D. Danilov, P. Galenko, Crystal growth of pure substances: phase-field simulations in comparison with analytical and experimental results, Journal of Computational Physics 207 (2005) 221–239, doi:10.1016/j.jcp.2005.01.018.
[13] A.C. Jones, A Projected Multigrid Method for the Solution of Nonlinear Finite Element Problems on Adaptively Refined Grids, Ph.D. thesis, School of Computing, University of Leeds, 2005.
[14] W.L. George, J.A. Warren, A parallel 3D dendritic growth simulator using the phase-field method, Journal of Computational Physics 177 (2002) 264–283, doi:10.1006/jcph.2002.7005.
[15] C.W. Lan, Y.C. Chang, C.J. Shih, Adaptive phase field simulation of non-isothermal free dendritic growth of a binary alloy, Acta Materialia 51 (2003) 1857–1869, doi:10.1016/S1359-6454(02)00582-7.
[16] P. Zhao, M. Vénere, J.C. Heinrich, D.R. Poirier, Modeling dendritic growth of a binary alloy, Journal of Computational Physics 188 (2003) 434–461, doi:10.1016/S0021-9991(03)00185-2.
[17] W. Hundsdorfer, J.G. Verwer, Numerical Solution of Time-Dependent Advection–Diffusion–Reaction Equations, Springer-Verlag, 2003.
[18] A. Karma, W.J. Rappel, Phase-field method for computationally efficient modeling of solidification with arbitrary interface kinetics, Physical Review E 53 (1996).
[19] A. Karma, W.J. Rappel, Quantitative phase-field modelling of dendritic growth in two and three dimensions, Physical Review E 57 (4) (1998).
[20] B. Echebarria, R. Folch, A. Karma, M. Plapp, Quantitative phase-field model of alloy solidification, Physical Review E 70 (2004).
[21] U. Trottenberg, C. Oosterlee, A. Schüller, Multigrid, Academic Press, 2001.
[22] W. Hackbusch, Multi-Grid Methods and Applications, Springer Series in Computational Mathematics, Springer-Verlag, Berlin/Heidelberg, 1985.
[23] W.L. Briggs, V.E. Henson, S.F. McCormick, A Multigrid Tutorial, second ed., SIAM, Philadelphia, 2000.
[24] A. Brandt, Multi-Level Adaptive Solutions to Boundary-Value Problems, Mathematics of Computation 31 (1977) 333–390.
[25] S. McCormick, J. Thomas, The Fast Adaptive Composite Grid (FAC) Method for Elliptic Equations, Mathematics of Computation 46 (1986) 439–456.
[26] J. Lang, Adaptive Mulitlevel Solution of Nonlinear Parabolic PDE Systems: Theory, Algorithm and Application, Springer, 2001.
[27] W. Hackbusch, Theorie und Numerik elliptischer Differentialgleichungen, Teubner Studienbücher, Teubner, Stuttgart, 2003.