Available online at www.sciencedirect.com
SCIENCE DIRECT®
Journal of Computational Physics 202 (2005) 577–601
JOURNAL OF COMPUTATIONAL PHYSICS
www.elsevier.com/locate/jcp

A fourth order accurate discretization for the Laplace and heat equations on arbitrary domains, with applications to the Stefan problem ☆

Frédéric Gibou [a,b,*], Ronald Fedkiw [b]

[a] *Mathematics Department, Stanford University, Stanford, CA 94305, USA*
[b] *Computer Science Department, Stanford University, Stanford, CA 94305, USA*

Received 6 October 2003; received in revised form 3 May 2004; accepted 27 July 2004
Available online 22 September 2004

Abstract

In this paper, we first describe a fourth order accurate finite difference discretization for both the Laplace equation and the heat equation with Dirichlet boundary conditions on irregular domains. In the case of the heat equation we use an implicit discretization in time to avoid the stringent time step restrictions associated with requirements for explicit schemes. We then turn our focus to the Stefan problem and construct a third order accurate method that also includes an implicit time discretization. Multidimensional computational results are presented to demonstrate the order accuracy of these numerical methods.

1. Introduction

Various numerical methods have been developed to solve the Stefan problem. These methods need to be able to efficiently solve the heat equation on irregular domains and keep track of a moving interface that may undergo complex topological changes.

☆ Research supported in part by an ONR YIP and PECASE award (N00014-01-1-0620), a Packard Foundation Fellowship, a Sloan Research Fellowship, ONR N00014-03-1-0071, ONR N00014-02-1-0720 and NSF DMS-0106694. In addition, the first author was supported in part by an NSF postdoctoral fellowship (DMS-0102029).
* Corresponding author. Present address: Department of Mechanical and Environmental Engineering, University of California, Barbara Santa, CA 93106-5070, USA.
E-mail address: fgibou@engineering.ucsb.edu (F. Gibou).

0021-9991/$ - see front matter © 2004 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2004.07.018

The interface that separates the two phases can be either explicitly tracked or implicitly captured. The main disadvantage of an explicit approach, e.g. front tracking (see e.g. [16]), is that special care is needed for topological changes such as merging or breaking. The explicit treatment of connectivity makes the method challenging to extend to three spatial dimensions. Implicit representations such as the level set method [23,29] or the phase-field method [17] represent the front as an isocontour of a continuous function. Topological changes are consequently handled in a straightforward fashion, and thus these methods are readily implemented in both two and three spatial dimensions.

Phase-field methods represent the front implicitly and have produced impressive three-dimensional results (see e.g. [7,17]). However, these methods only have an approximate representation of the front location and thus the discretization of the diffusion field is less accurate near the front resembling an enthalpy method [5]. Moreover, it is often challenging to add new physics to the model since new asymptotic analysis is often required. For more details on phase field methods for the Stefan problem, see [17] and the references therein.

In this paper, we employ the sharp interface implicit representation of the level set method [23,29]. The earliest level set method for solidification type problems was presented in [30] where the authors recast the equations of motion into a boundary integral equation and used the level set method to update the location of the interface. In [2] the boundary integral equations were avoided by using a finite difference method to solve the diffusion equation on a Cartesian grid with Dirichlet boundary conditions imposed on the interface. The jump in the first derivatives of the temperature was used to compute an interface velocity that was extended to a band about the interface and used to evolve the level set function in time. Kim et al. [10] showed that such discretization produces results in accordance with solvability theory.

In [34], the authors discretized the heat equation on a Cartesian grid in a manner quite similar to that proposed in [2] resulting in a non-symmetric matrix when applying implicit time discretization. Udaykumar et al. [34] used front tracking to update the location of the interface improving upon the front tracking approach proposed in [16] which used the smeared out immersed boundary method [24] and an explicit time discretization.

In [15], the authors solved a variable coefficient Poisson equation in the presence of an irregular interface where Dirichlet boundary conditions were imposed. They used a finite volume method that results in a non-symmetric discretization matrix. Both multigrid methods and adaptive mesh refinement were used and in [14] this non-symmetric discretization was coupled to a volume of fluid front tracking method in order to solve the Stefan problem.

In [26] the authors used adaptive finite element methods for both the heat equation and for the interface evolution producing stunning three-dimensional results. Other remarkable three-dimensional results can be found in the finite difference diffusion Monte Carlo method of [25].

Recently, Zhao and Heinrich [35] formulated a second order accurate method for the Stefan problem in two spatial dimensions using a Galerkin finite element approach to solve for the energy equation. In this work, the interface was tracked with a set of marker particles making the method potentially hard to extend in three spatial dimensions. Moreover, the velocity is computed under the assumption that the interface cuts the element in a straight line.

The interested reader is referred to [2,8,16] and the references therein for an extensive summary of computational results for the Stefan problem.

Standard proofs of convergence use stability and consistency analysis to imply convergence, i.e, given stability, a *sufficient* condition for a scheme to be pth order accurate is that the local truncation error is pth order. However, a pth order local truncation error is not a *necessary* condition and one can derive pth order accurate schemes despite the fact that their local truncation error is of lower order. Manteuffel and White [22] (see also [18]) have made this point in the context of second order, scalar boundary value problems on non-uniform meshes. In fact, in the process of constructing second order accurate methods for

such problems, many authors had unnecessarily focused on imposing special restrictions on the mesh size in order to obtain a second order local truncation error, see e.g [4,11]. In the case of our present work, authors have also been misled by the limitation of standard convergence analysis proofs and have proposed unnecessarily complex schemes. For example, in [2] (see also [10]) the authors approximate the Laplace operator with the standard second order central scheme, limiting the overall solution to second order accuracy. However, the discretization of the Laplace operator for grid nodes neighboring the interface amounts to differentiating a quadratic interpolant of the temperature twice in each spatial dimension. Gibou et al. [9] reformulated the interface treatment with the use of ghost cells (based on the ghost fluid method [6]) defined by extrapolation of the temperature across the interface and showed that local linear extrapolation is enough to obtain second order spatial accuracy for both the Laplace equation and the heat equation on irregular domains. Moreover, such a discretization has the benefit of yielding a symmetric linear system as opposed to a non-symmetric system in [2]. This scheme served as the basis of a simple method to solve the Stefan problem. It was further used in [8] to show that one could obtain solutions in agreement with solvability theory, and could simulate many of the physical features of crystal growth such as molecular kinetics and surface tension.

In this paper, we exploit the methodology of [9] to derive a fourth order accurate finite difference discretization for the Laplace equation on irregular domains. Then, we apply this framework to derive a fourth order accurate discretization for the heat equations with Dirichlet boundary conditions on arbitrary domains. In this case we use an implicit time discretization to avoid the stringent time step restrictions induced by explicit schemes. We then turn our focus to the Stefan problem with Dirichlet boundary conditions and construct a third order accurate discretization that includes implicit integration in time. Multidimensional computational results are presented to verify the order accuracy of these numerical methods.

2. Laplace equation

Consider a Cartesian computational domain, $\Omega \in R^n$, with exterior boundary, $\partial\Omega$, and a lower dimensional interface, Γ, that divides the computational domain into disjoint pieces, Ω^- and Ω^+. The Laplace equation is given by

$$\Delta T(\vec{x}) = f(\vec{x}), \quad \vec{x} \in \Omega^-, \tag{1}$$

where $\vec{x} = (x, y, z)$ is the vector of spatial coordinates, $\Delta = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2}$ is the Laplace operator, and T is assumed to be smooth on Ω^-. On Γ, Dirichlet boundary conditions are specified.

To separate the different domains, we introduce a level set function ϕ defined as

$$\begin{cases} \phi < 0 & \text{for } \vec{x} \in \Omega^-, \\ \phi > 0 & \text{for } \vec{x} \in \Omega^+, \\ \phi = 0 & \text{for } \vec{x} \in \Gamma. \end{cases}$$

A convenient choice that ensures numerical robustness is to define ϕ as the signed distance function to the interface. The level set is also used to identify the location of the interface to high order accuracy as will be discussed throughout this paper.

The discretization of the Laplace operator, including the special treatments needed at the interface, is performed in a dimension by dimension fashion. Therefore, without loss of generality, we first describe the discretization in one spatial dimension.

2.1. 1D Laplace

Consider the Laplace equation in one spatial dimension, i.e. $T_{xx} = f$. The computational domain is discretized into cells of size Δx with the grid nodes x_i located at their centers. The solution to the Laplace equation is computed at the grid nodes and is written as $T_i = T(x_i)$. We consider the standard fourth order discretization:

$$(T_{xx})_i \approx \frac{-\frac{1}{12}T_{i-2} + \frac{4}{3}T_{i-1} - \frac{5}{2}T_i + \frac{4}{3}T_{i+1} - \frac{1}{12}T_{i+2}}{\Delta x^2}. \tag{2}$$

For each unknown, T_i, Eq. (2) is used to fill in one row of a matrix creating a linear system of equations. This discretization is valid if all the node values used belong to the same domain, but needs to be modified otherwise. For example, suppose the interface location, x_I, is located in between the nodes x_i and x_{i+1} (see Fig. 1) and suppose that we seek to write the equation satisfied by T_i. Since the solution is not defined across the interface, we need valid values for T_{i+1} and T_{i+2} that 'emulate' the behavior of the solution defined to the left of the interface. We achieve this by defining 'ghost values' T^G_{i+1} and T^G_{i+2} constructed by extrapolating the values of T across the interface. The discretization for such points in the neighborhood of the interface is then rewritten as

$$(T_{xx})_i \approx \frac{-\frac{1}{12}T_{i-2} + \frac{4}{3}T_{i-1} - \frac{5}{2}T_i + \frac{4}{3}T^G_{i+1} - \frac{1}{12}T^G_{i+2}}{\Delta x^2}. \tag{3}$$

More precisely, we first construct an interpolant $\tilde{T}(x)$ of $T(x)$ on the left of the interface, such that $\tilde{T}(0) = T_i$, and then we define $T^G_{i+1} = \tilde{T}(\Delta x)$ and $T^G_{i+2} = \tilde{T}(2\Delta x)$. Fig. 1 illustrates the definition of the ghost cells in the case of the linear extrapolation.

In this paper, we consider constant, linear, quadratic and cubic extrapolations defined by:

Constant extrapolation: Take $\tilde{T}(x) = d$ with:
- $d = T_I$.

Linear extrapolation: Take $\tilde{T}(x) = cx + d$ with:
- $\tilde{T}(0) = T_i$,
- $\tilde{T}(\theta\Delta x) = T_I$.

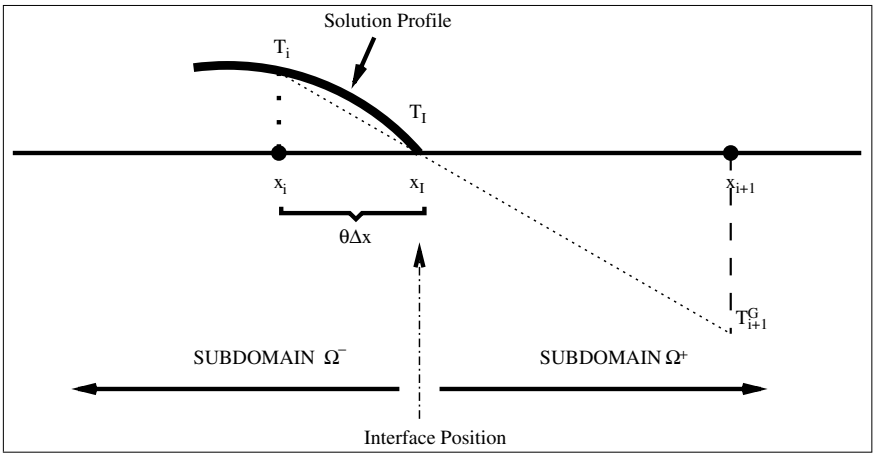


Fig. 1. Definition of the ghost cells with linear extrapolation. First, we construct a linear interpolant $\tilde{T}(x) = ax + b$ of T such that $\tilde{T}(0) = T_i$ and $\tilde{T}(\theta\Delta x) = T_I$. Then, we define $T^G_{i+1} = \tilde{T}(\Delta x)$ (likewise, $T^G_{i+2} = \tilde{T}(2\Delta x)$).

Quadratic extrapolation: Take $\tilde{T}(x) = bx^2 + cx + d$ with:
- $\tilde{T}(-\Delta x) = T_{i-1}$,
- $\tilde{T}(0) = T_i$,
- $\tilde{T}(\theta\Delta x) = T_{\mathrm{I}}$.

Cubic extrapolation: Take $\tilde{T}(x) = ax^3 + bx^2cx + d$ with:
- $\tilde{T}(-2\Delta x) = T_i - 2$,
- $\tilde{T}(-\Delta x) = T_i - 1$,
- $\tilde{T}(0) = T_i$,
- $\tilde{T}(\theta\Delta x) = T_{\mathrm{I}}$.

In these equations $\theta = (x_{\mathrm{I}} - x_i)/\Delta x$ refers to the cell fraction occupied by the subdomain Ω^-.

Similarly, if we were solving for the domain Ω^+, the equation satisfied by T_{i+1} requires the definition of the ghost cells T_i^{G} and T_{i-1}^{G}. In this case, we write $T_i^{\mathrm{G}} = \tilde{T}(\Delta x)$ and $T_{i-1}^{\mathrm{G}} = \tilde{T}(2\Delta x)$ with the definition for $\tilde{T}$ modified as follows: θ is replaced by $1 - \theta$, T_i is replaced by T_{i+1}, T_{i-1} is replaced by T_{i+2} and T_{i-2} is replaced by T_{i+3}.

We note that the construction of $\tilde{T}$ cannot be arbitrary. It is obviously limited by the number of points within the domain, but also by how close the interface is from a grid node. The latter restriction comes from the fact that, as $\theta \to 0$, the behavior of the interpolant deteriorates. We found that a good rule of thumb is to shift the interpolation to be centered one grid point to the left when $\theta < \Delta x$, e.g. in the case of a linear extrapolation, we use the conditions $\tilde{T}(0) = T_{i-1}$ and $\tilde{T}((1 + \theta)\Delta x) = T_{\mathrm{I}}$ instead of $\tilde{T}(0) = T_i$ and $\tilde{T}(\theta\Delta x) = T_I$. Then, the ghost nodes are defined as $T_{i+1}^{\mathrm{G}} = \tilde{T}(2\Delta x)$ and $T_{i+1}^{\mathrm{G}} = \tilde{T}(3\Delta x)$. Finally, we lower the degree of the interpolant in order to preserve the higher order extrapolation follows similarly. The resulting linear system is pentadiagonal (in the case where the stencil has to be shifted, we lower the degree of the interpolant in order to preserve that structure).

In the case of the constant and the linear extrapolations, the matrix entries to the right of the diagonal for the ith row and to the left of the diagonal for the $(i + 1)$th row are equal to zero, yielding a symmetric linear system. This allows for the use of fast iterative solvers such as preconditioned conjugate gradient (see, e.g., [27]). Moreover the corresponding matrix is strictly diagonally dominant and therefore non-singular. In the case of extrapolations of higher degrees, the linear system is non-symmetric and not necessarily strictly diagonally dominant, but we can still develop high order accurate methods. We note that [20] designed a method that also yields a non-symmetric linear system but which is only second order accurate.

The overall accuracy of the method is also determined by the order of the extrapolation. We illustrate in Sections 2.1 and 2.2 that such discretizations yield first, second, third and fourth order accuracy in the case of the constant, linear, quadratic and cubic extrapolations, respectively.

The overall accuracy for T and the nature of the resulting linear system is determined by the degree of the interpolation function $\tilde{T}$, which is summarized in Table 1.

We test our methodology on the following example. Let $\Omega = [0,1]$ with an exact solution of $T = x^5 - x^3 + 12x^2 - 2.5x + 2$ on Ω^-. We define $\phi = x - .5$, hence the interface never falls on a grid node. Dirichlet boundary conditions are enforced on the $\partial\Omega$ using the exact solution. The following tables give the error between the numerical solution and the exact solution in the L^1- and L^∞-norms. These same results

Table 1
Order of accuracy and nature of the linear system corresponding to the constant, linear, quadratic and cubic case

Degree of extrapolation	Order of accuracy	Linear system
Constant	First	Symmetric
Linear	Second	Symmetric
Quadratic	Third	Non-symmetric
Cubic	Fourth	Non-symmetric

are also presented on a log–log plot in Fig. 2, where the open symbols represent the error in the L^∞-norm and the solid lines represent the least square fit. These results illustrate the first, second, third and fourth order accuracy in the case of constant, linear, quadratic and cubic extrapolations, respectively. In the case where the linear system is symmetric, we use a preconditioned conjugate gradient method with an incomplete Cholesky preconditioner. In the case where the linear system is non-symmetric, we use the BiCGSTAB method (see e.g. [27]).

Constant extrapolation

Number of points	L^1-error	Order	L^∞-error	Order
16	1.307×10^{-1}	–	2.369×10^{-1}	–
32	6.248×10^{-2}	1.06	1.196×10^{-1}	0.98
64	3.057×10^{-2}	1.03	6.018×10^{-2}	0.99
128	1.512×10^{-2}	1.02	3.020×10^{-2}	1.00

Linear extrapolation

Number of points	L^1-error	Order	L^∞-error	Order
16	4.456×10^{-3}	–	8.463×10^{-3}	–
32	1.013×10^{-3}	2.13	2.045×10^{-3}	2.05
64	2.417×10^{-4}	2.06	5.031×10^{-4}	2.02
128	5.901×10^{-5}	2.03	1.247×10^{-4}	2.01

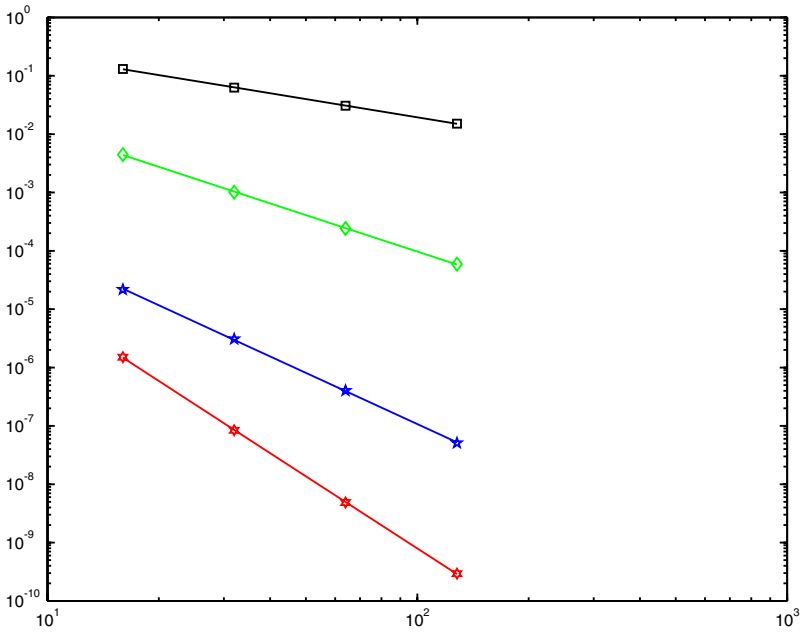


Fig. 2. Error analysis in the L^∞-norm for the one-dimensional Laplace equation. The open symbols represent the errors versus the number of grid nodes on a log–log scale and the solid lines are the least square fits with slopes −1.03, −2.07, −2.91 and −4.10 in the case where the ghost cells are defined by constant, linear, quadratic and cubic extrapolations, respectively.

Quadratic extrapolation

Number of points	L^1-error	Order	L^∞-error	Order
16	2.168×10^{-5}	–	5.197×10^{-5}	–
32	3.084×10^{-6}	2.81	7.532×10^{-6}	2.78
64	4.013×10^{-7}	2.94	9.971×10^{-7}	2.91
128	5.095×10^{-8}	2.98	1.278×10^{-7}	2.96

Cubic extrapolation

Number of points	L^1-error	Order	$L^\infty -$ error	Order
16	1.502×10^{-6}	–	8.519×10^{-6}	–
32	8.416×10^{-8}	4.15	5.401×10^{-7}	3.97
64	4.867×10^{-9}	4.11	3.378×10^{-8}	3.99
128	2.936×10^{-10}	4.05	2.109×10^{-9}	4.00

2.2. 2D Laplace

The methodology discussed in Section 2.1 extends naturally to two and three spatial dimensions. For example, in the case of two spatial dimensions, we solve

$$T_{xx} + T_{yy} = f.$$

The spatial derivatives T_{xx} and T_{yy} are approximated as

$$(T_{xx})_{i,j} \approx \frac{-\frac{1}{12}T_{i-2,j} + \frac{4}{3}T_{i-1,j} - \frac{5}{2}T_{i,j} + \frac{4}{3}T_{i+1,j} - \frac{1}{12}T_{i+2,j}}{\Delta x^2},$$

$$(T_{yy})_{i,j} \approx \frac{-\frac{1}{12}T_{i,j-2} + \frac{4}{3}T_{i,j-1} - \frac{5}{2}T_{i,j} + \frac{4}{3}T_{i,j+1} - \frac{1}{12}T_{i,j+2}}{\Delta y^2},$$

and for cells cut by the interface, ghost values are defined by extrapolating the value of T across the interface as described in Section 2.1. Note that in two spatial dimensions, the definition of the ghost cells involves θ^x and θ^y, i.e. the cell fractions in the x and y direction, respectively. These quantities are evaluated as follows. Consider a grid node (x_i,y_j) in the neighborhood of the interface. We first construct a cubic interpolant $\tilde{\phi}^x$ of ϕ in the x-direction and find the interface location x_I by solving $\tilde{\phi}^x(x_I) = 0$. Then, we define $\theta^x = |x_i - x_I|/\Delta x$. The procedure to find θ^y is similar.

We emphasize that the numerical discretization of T_{xx} is independent from that of T_{yy}, making the procedure trivial to extend to two and three spatial dimensions.

We illustrate the order of accuracy on the following example. Let $\Omega = [-1,1] \times [-1,1]$ with an exact solution of $T = \sin(\pi x) + \sin(\pi y) + \cos(\pi x) + \cos(\pi y) + x^6 + y^6$ in Ω^-. The interface is parameterized by $(x(\alpha),y(\alpha))$, where

$$\begin{cases} x(\alpha) = .02\sqrt{5} + (.5 + .2\sin(5\alpha))\cos(\alpha), \\ y(\alpha) = .02\sqrt{5} + (.5 + .2\sin(5\alpha))\sin(\alpha), \end{cases}$$

with $\alpha \in [0,2\pi]$.

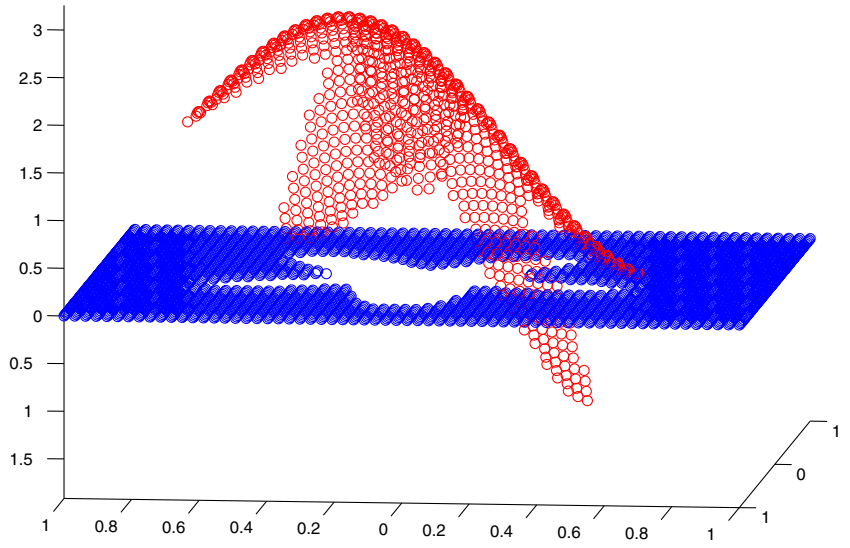


Fig. 3. Solution of the Laplace equation on an irregular domain in two spatial dimensions. The exact solution is $T = \sin(\pi x) + \sin(\pi y) + \cos(\pi x) + \cos(\pi y) + x^6 + y^6$ in Ω^-. The grid size is 64×64 and the ghost cells are defined by cubic extrapolation.

Fig. 3 depicts the solution on a 64×64 grid and Fig. 4 illustrates the accuracy in the L^∞-norm. Note that on irregular domains, the number of available grid nodes within the domain might limit the extrapolation to a lower degree for some grid resolutions. This partially explains the 'oscillatory' nature of the accuracy results in the graph of Fig. 3. However, the slopes of the least square fits are still in accordance with first, second, third and fourth order accuracy for the constant, linear, quadratic and cubic extrapolation definitions of the ghost nodes.

3. Heat equation

Consider a Cartesian computational domain, $\Omega \in R^n$, with exterior boundary, $\partial\Omega$, and a lower dimensional interface, Γ, that divides the computational domain into disjoint pieces, Ω^- and Ω^+. The heat equation is written as

$$T_t(\vec{x}) = \Delta T(\vec{x}), \quad \vec{x} \in \Omega^-, \tag{4}$$

where $T(\vec{x})$ is assumed to be smooth on Ω^-. On Γ, Dirichlet boundary conditions are specified.

Explicit time discretization schemes are impractical in the case of arbitrary domains because they suffer from stringent time step restrictions. For example in one spatial dimension, we must impose a time step restriction of $O(\theta^2 \Delta x^2)$ with $0 < \theta = (x_I - x_i)/\Delta x \leqslant 1$ for cells cut by the interface. Since θ can be arbitrarily small, explicit schemes are prohibitively computationally expensive. Although one could remesh the domain to keep θ reasonable, in the case of a moving interface this would require remeshing every time the value of θ gets below an acceptable threshold. On the other hand, an implicit time discretization can obtain an unconditionally stable if its region of absolute stability spans the entire left half plane (see [19] for more details). Thus, we choose the Crank–Nicholson scheme and impose a time step restriction of $\Delta t = c\Delta x^2$, $0 < c < 1$ in order to obtain a fourth order accurate discretization in time. However, we note that backward differentiation formulae or implicit Runge–Kutta schemes could be use instead in order to relax

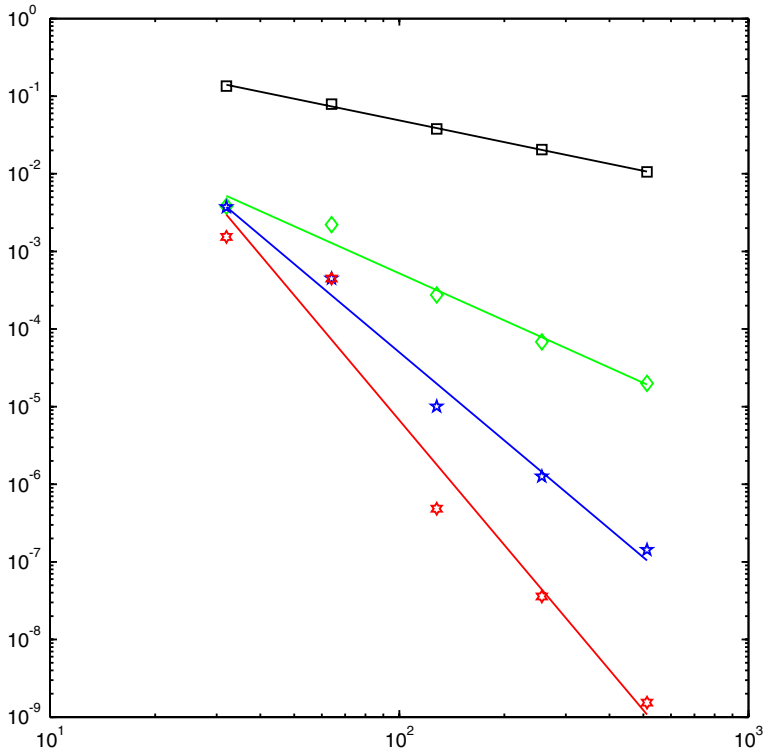


Fig. 4. Error analysis in the L^∞-norm for the two-dimensional Laplace equation. The open symbols represent the errors versus the number of grid nodes on a log–log scale and the solid lines are the least square fits with slopes $-.85$, -1.94, -2.94 and -3.96 in the case where the ghost cells are defined by constant, linear, quadratic and cubic extrapolations, respectively. Note that on a 32×32 mesh, the error for the quadratic extrapolation is the same as that for the cubic extrapolation. This is an example where there was not enough points within the domain to construct a cubic interpolant and the algorithm is temporarily forced to use a quadratic interpolant. Moreover, since we use the L^∞-norm, the results are strict. Although not presented here, the L^1-error is much less affected by artifacts since it is an average quantity.

the time step restriction to $\Delta t = c\Delta x$, $0 < c < 1$. The reader is referred to [19] for additional details about numerical methods for ordinary differential equations.

The Crank–Nicholson scheme can be written as

$$\left(I - \frac{\Delta t}{2} A^{n+1}(\Delta)\right) T^{n+1} = \left(I + \frac{\Delta t}{2} A^{n}(\Delta)\right) T^{n},$$

where $A^n(\Delta)$ and $A^{n+1}(\Delta)$ represent the spatial approximation of the Laplace operator at time t^n and t^{n+1}, respectively. The spatial discretization is performed in a dimension by dimension fashion and resembles that of Section 2. More precisely, we first evaluate the right-hand side $f^n = (I + \frac{\Delta t}{2} A^n(\Delta)) T^n$ at time t^n using the methodology of Section 2 to define the ghost cells dimension by dimension. We emphasize that we only need to define the ghost cells in the Cartesian directions. Then, we solve

$$\left(I - \frac{\Delta t}{2} A^{n+1}(\Delta)\right) T^{n+1} = f^{n}.$$

The Laplace operator at time t^{n+1} is discretized along the lines of Section 2 as well. Each equation is used to fill one row of a linear system that is then solved with an iterative solver.

Since the discretization is performed in a dimension by dimension fashion, we first present the one-dimensional case.

3.1. 1D Heat

In the one spatial dimension case, we discretize the heat equation as

$$T^{n+1} - \frac{\Delta t}{2} A^{n+1}(T_{xx}) = T^n + \frac{\Delta t}{2} A^n(T_{xx}),$$

where $A^n(T_{xx})$ and $A^n(T_{xx})$ are the fourth order approximations of T_{xx} at time t^n and t^{n+1}, respectively. The discretization of the heat equation is performed in two steps and depends heavily on that of the Laplace operator described in Section 2.1. First, we approximate T^n_{xx} with the fourth order accurate discretization of

$$(T^n_{xx})_i \approx \frac{-\frac{1}{12}T^n_{i-2} + \frac{4}{3}T^n_{i-1} - \frac{5}{2}T^n_i + \frac{4}{3}T^n_{i+1} - \frac{1}{12}T^n_{i+2}}{\Delta x^2}.$$

The special treatment needed for grid nodes neighboring the interface is performed as described in Section 2.1, using the values at the interface at time t^n. We then evaluate $f^n = T^n + \frac{\Delta t}{2} A^n(T_{xx})$ and we are left to solve

$$T^{n+1} - \frac{\Delta t}{2} A^{n+1}(T_{xx}) = f^n. \tag{5}$$

We consider again the standard fourth order discretization to approximate T_{xx} at time t^{n+1}:

$$\left(T^{n+1}_{xx}\right)_i \approx \frac{-\frac{1}{12}T^{n+1}_{i-2} + \frac{4}{3}T^{n+1}_{i-1} - \frac{5}{2}T^{n+1}_i + \frac{4}{3}T^{n+1}_{i+1} - \frac{1}{12}T^{n+1}_{i+2}}{\Delta x^2}. \tag{6}$$

For each unknown, T^{n+1}_i, Eqs. (5) and (6) are used to fill in one row of a matrix creating a linear system of equations. The treatment of the grid nodes in the neighborhood of the interface is also based on defining ghost node values and uses the values of the temperature at the interface at time t^{n+1}.

For the remainder of this paper, we focus on designing a fourth order accurate method for the heat equation and a third order accurate method for the Stefan problem. Therefore, we present only the results for these accuracy tests, although we have checked that one obtains first, second, third and fourth order accuracy for the constant, linear, quadratic and cubic extrapolations, respectively. The nature of the linear system and the order of accuracy is the same as that of the Laplace operator (see Table 1).

Consider the following example. Let $\Omega = [-1,1]$ with an exact solution of $T = e^{-\pi^2 t}\cos(\pi x)$ on Ω^-. We take $\phi = x - .313$ (thus $0 < \theta < 1$). Dirichlet boundary conditions are enforced on the $\partial\Omega$ using the exact solution and the final time is $t = 1/\pi^2$. The ghost values are defined with a cubic extrapolation of T across the interface. Fig. 5 illustrates the fourth order accuracy in the L^∞-norm.

3.2. 2D heat

The algorithm described above extends readily to two and three spatial dimensions. The approximations of T_{xx} and T_{yy} are performed independently of each other making the procedure trivial to implement.

Consider the following example. Let $\Omega = [-1,1]\times[-1,1]$ with an exact solution of $T = e^{-2t}\sin(x)\sin(y)$ in Ω^-. The interface is parameterized by $(x(\alpha),y(\alpha))$, where

$$\begin{cases} x(\alpha) = .02\sqrt{5} + (.5 + .2\sin(5\alpha))\cos(\alpha), \\ y(\alpha) = .02\sqrt{5} + (.5 + .2\sin(5\alpha))\sin(\alpha), \end{cases}$$

with $\alpha \in [0,2\pi]$.

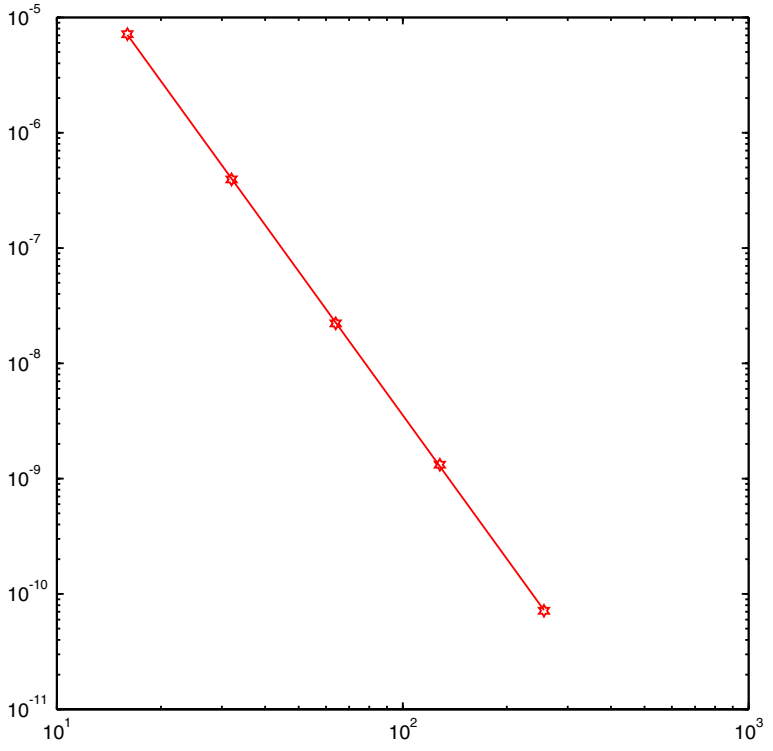


Fig. 5. Error analysis in the L^∞-norm for the one-dimensional heat equation. The open symbols represent the error versus the number of grid nodes on a log–log scale and the solid line depicts the least square fit with slope -4.14 in the case where the ghost cells are defined by cubic extrapolation.

Fig. 6 depicts the solution on a 64×64 grid and Fig. 7 demonstrates the fourth order accuracy in the L^∞-norm.

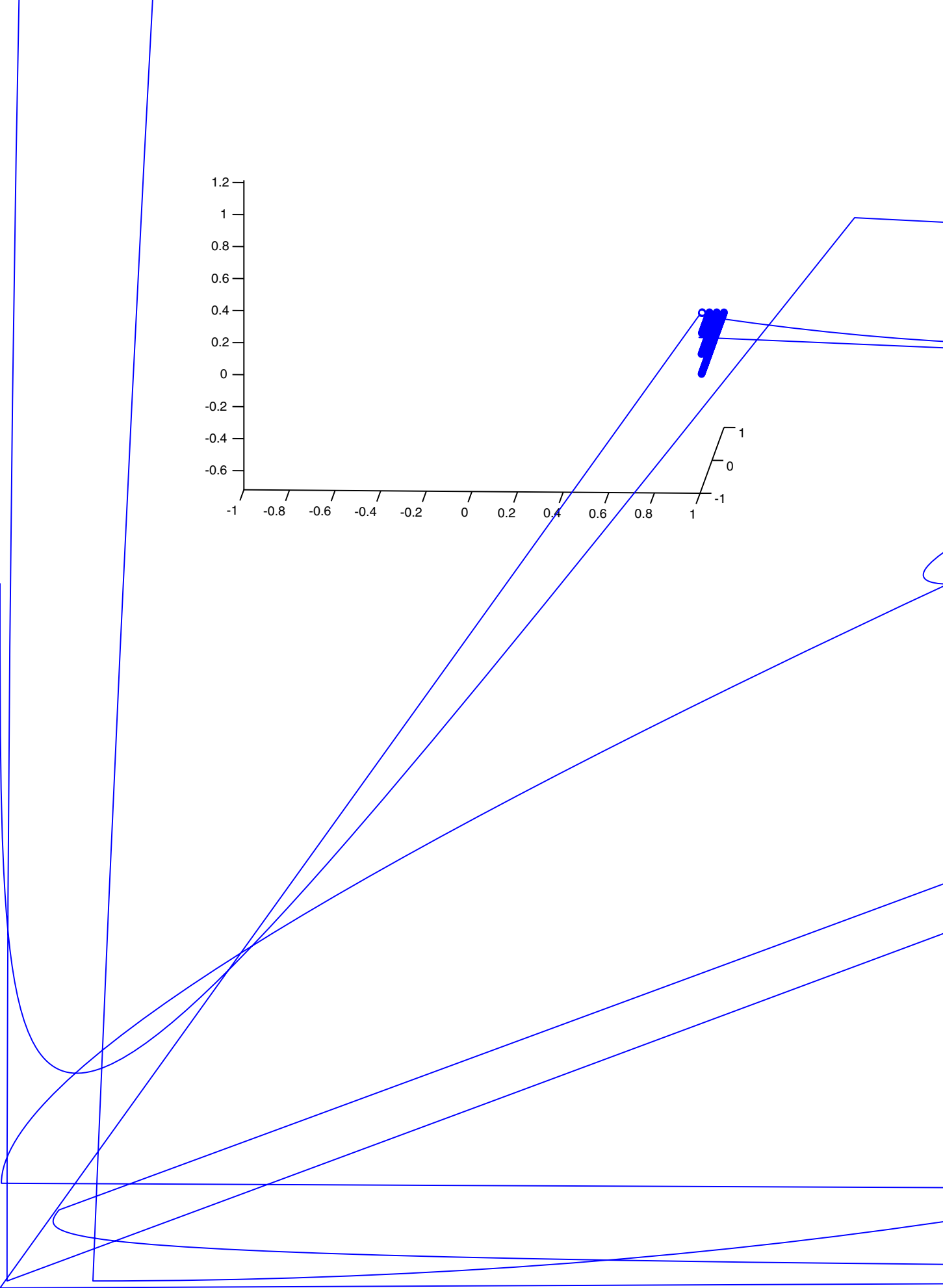
4. Stefan problem

Consider again a Cartesian computational domain, $\Omega \in R^n$, with exterior boundary, $\partial\Omega$, and a lower dimensional interface, Γ, that divides the computational domain into disjoint pieces, Ω^- and Ω^+. The Stefan problem is written as

$$\begin{cases} T_t(\vec{x}) = D\Delta T(\vec{x}), & \vec{x} \in \Omega, \\ T(\vec{x}) = 0, & \vec{x} \in \Gamma, \\ V_n = -D[\nabla T]_{|\Gamma} \cdot \vec{n}, \end{cases}$$

where D is the diffusion coefficient, assumed in this work to be constant on each subdomain, but may be discontinuous across the interface. $T(\vec{x})$ is assumed to be smooth on each disjoint subdomain, Ω^- and Ω^+, but may have a kink at the interface Γ. On $\partial\Omega$, Dirichlet boundary conditions are specified.

We need to both discretize the heat equation and evaluate the velocity at the interface. The added complexity for the Stefan problem stems from the fact that the interface is evolving in time. We keep track of the interface evolving under the velocity field $\vec{V} = (u, v, w)$ by solving the advection equation:


1.2
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1
1
0
-1

$$\phi_t + \vec{V} \cdot \nabla \phi = 0.$$

The velocity components are defined by the x, y and z projections of the jump in the temperature gradient, i.e. $(u,v,w) = ([T_x]_{|\Gamma},[T_y]_{|\Gamma},[T_z]_{|\Gamma})$. The level set advection equation is discretized with a HJ-WENO scheme [12], see also [13,21]. For more details on the level set method, see e.g. [23,29].

4.1. Discretization of the Stefan problem

Consider the Crank–Nicholson framework:

$$T^{n+1} - \frac{\Delta t}{2} A^{n+1}(T_{xx}) = T^n + \frac{\Delta t}{2} A^n(T_{xx}), \tag{7}$$

and let the temperature be defined at time t^n. The algorithm to solve the Stefan problem is:

1. Extrapolate T^n in the *normal* direction.
2. Discretize $f^n = T^n + \frac{\Delta t}{2} A^n(\Delta) T^n$ in Eq. (7).
3. Evolve the level set function for one time step Δt.
4. Assemble and solve the linear system for T^{n+1}.
5. Repeat 1–4 until done.

4.1.1. Why extrapolate in the normal direction?

We follow along the lines of Section 3 to discretize the heat equation with the Crank–Nicholson scheme by writing the discretization of the Laplace operator at time t^n and evaluating the right-hand side $f^n = T^n + \frac{\Delta t}{2} A^n(\Delta) T^n$. However, the added difficulty is due to the moving domain. Fig. 8 depicts a two-dimensional example. As the interface moves from its position at time t^n to its new location at time t^{n+1}, new grid nodes are added to Ω^- (depicted for example by the dark node). Since we need to evaluate the right-hand side at these new nodes, valid values for T^n must exist there. Moreover, since the interface is evolved in the normal direction, valid values of T^n must exist not only in the Cartesian directions, as it was the case for the heat equation, but in every direction. Therefore, a high order extrapolant must be defined in the *normal* direction at the interface and such a procedure must be easy to implement in one, two and three spatial dimensions.

4.1.2. High order extrapolation

High order extrapolation in the normal direction is performed in a series of steps, as proposed in [1]. For example, suppose that we seek to extrapolate T from the region where $\phi \leqslant 0$ to the region where $\phi > 0$. In the case of a cubic extrapolation, we first compute $T_{nnn} = \vec{\nabla}(\vec{\nabla}(\vec{\nabla} T \cdot \vec{n}) \cdot \vec{n}) \cdot \vec{n}$ in the region $\phi \leqslant 0$ and extrapolate it across the interface in a constant fashion by solving the following partial differential equation:

$$\frac{\partial T_{nnn}}{\partial \tau} + H(\phi + band) \vec{\nabla} T_{nnn} \cdot \vec{n} = 0,$$

where H is the Heaviside function and *band* accounts for the fact that T_{nnn} is not defined in the region where $\phi \geqslant band$. Typically, we take $band = 2\ \mathrm{d}x^2 + \mathrm{d}y^2$ in the case where T_{nnn} is computed by central differencing.

Then, the value of T across the interface is found by solving the following three partial differential equations: First solve

$$\frac{\partial T_{nn}}{\partial \tau} + H(\phi)\left(\vec{\nabla} T_{nn} - T_{nnn}\right) = 0,$$

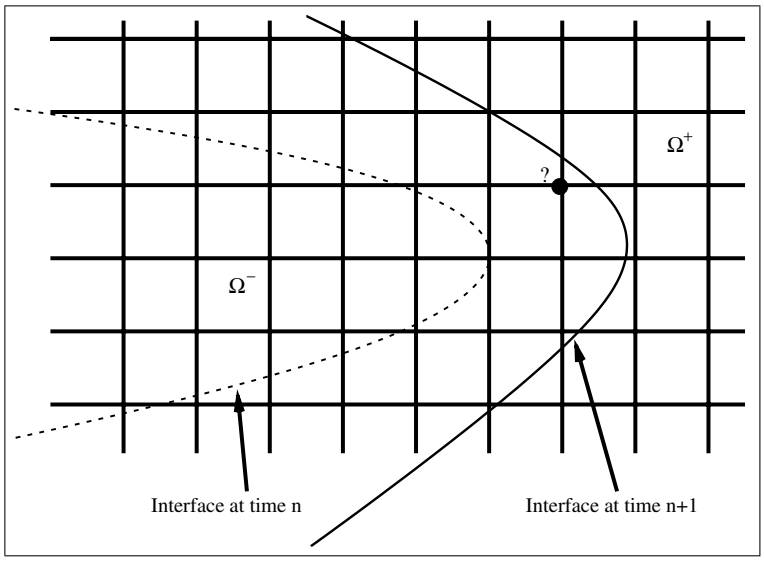


Fig. 8. Interface at time t^n (dashed line) and t^{n+1} (solid line). The dark point with a question mark represents a grid node that is swept over by the interface between two consecutive time steps and where valid value of T^n needs to be extrapolated in a non-Cartesian normal direction in order to evaluate $f^n = T^n + \frac{\Delta t}{2} A^n(\Delta) T^n$ in Eq. (7).

defining T_{nn} in such a way that its normal derivative is equal to T_{nnn}. Then, solve

$$\frac{\partial T_n}{\partial \tau} + H(\phi)\left(\vec{\nabla} T_n - T_{nn}\right) = 0,$$

defining T_n in such a way that its normal derivative is equal to T_{nn}. Finally, solve

$$\frac{\partial T}{\partial \tau} + H(\phi)\left(\vec{\nabla} T - T_n\right) = 0,$$

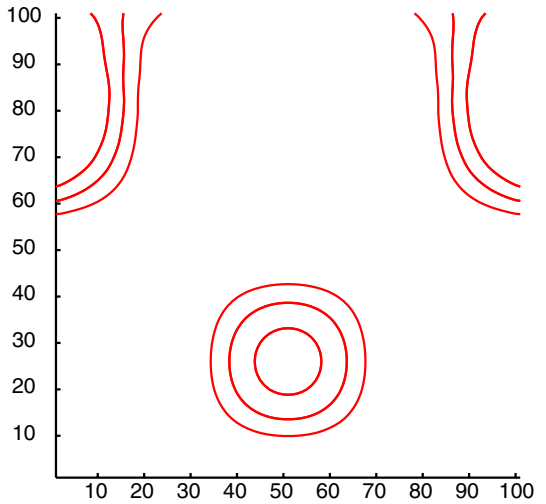
defining T in such a way that its normal derivative is equal to T_n. These PDEs are solved in fictitious time τ for a few iterations (typically 15) since we only seek to extrapolate the values of T in a narrow band of a few grid cells around the interface.

Fig. 9 illustrates the cubic extrapolation. This example is taken from [1]. Consider a computational domain $\Omega = (-\pi,\pi) \times (-\pi,\pi)$ separated into two regions: Ω^- defined as the interior of a disk with center at the origin and radius two, and its complementary Ω^+. The function T to be extrapolated from Ω^- to Ω^+ is defined as $T = \cos(x)\sin(y)$ for $\vec{x} \in \Omega^-$. Fig. 9 (top) illustrates the contours of T after it has been extrapolated across the interface and Fig. 9 (bottom) depicts the contour plots for the exact solution for comparison. For the sake of clarity we have extrapolated T in the entire region in this example but we iterate that in practice the extrapolation is performed only in a neighborhood of the interface. The extrapolation is fourth order accurate in the L^∞-norm as demonstrated in Fig. 10.

At the end of the high order extrapolation procedure we have T^n at all grid nodes near the interface, so that if the interface moves across new grid nodes, we can still evaluate $T^n + \frac{\Delta t}{2} A^n(\Delta) T^n$ and proceed as in Section 3 to assemble the linear system.

4.2. Discretization of the velocity

The method described above can be used to obtain a fourth order accurate temperature, at best. As a consequence, the velocity involving gradients of the temperature will only be third order accurate. Not sur-



prisingly, the order of accuracy in the computation of the velocity is a determining factor towards the overall accuracy of the method.

For example let $\Omega = [0,1]$ with an exact solution of $T = e^{t - x + .5} - 1$ on Ω^- and take $\phi = x - .5$ at $t = 0$. Dirichlet boundary conditions are enforced on $\partial\Omega$ using the exact solution. We compute the solution at time $t_{\text{final}} = .25$. We use the the Crank–Nicholson scheme with $\Delta t \approx \Delta x^2$ to allow for a fourth order accurate scheme in time and we use the cubic extrapolation to define the ghost values. In these tests, we use the *exact* interface velocity perturbed by an $O(\Delta x^3)$, $O(\Delta x^2)$ and $O(\Delta x)$ amount. Fig. 11 illustrates the fact that a third, second and first order accurate velocity will produce an overall third, second and first order accuracy, respectively.

The examples above demonstrate that a $O(\Delta x)$ perturbation in the velocity forces the solution to be first order accurate in the case of the Stefan problem, regardless of the order of accuracy of the discretization of the heat equation operator. In [9], the velocity could be at most first order accurate, since it was computed

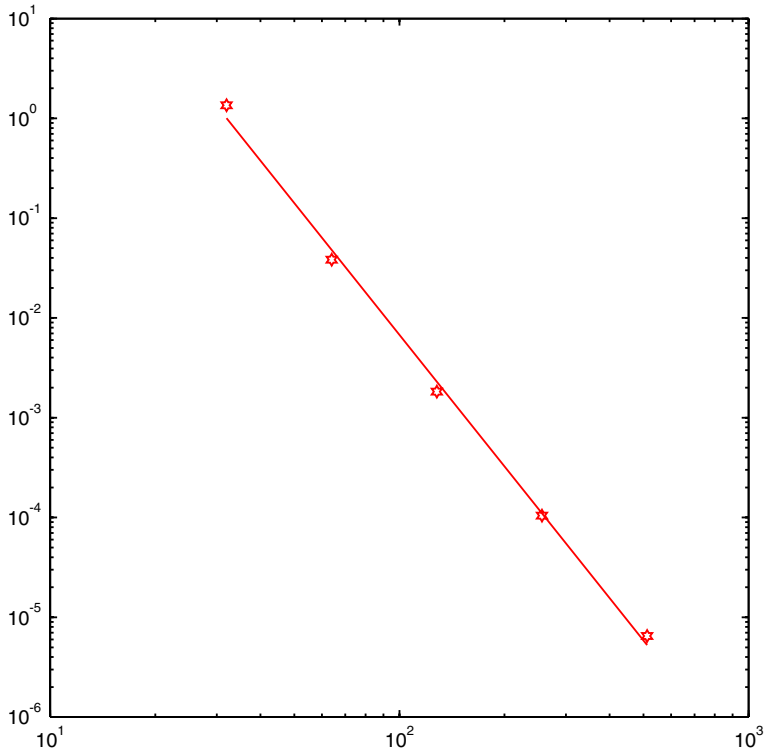


Fig. 10. Error analysis in the L^∞-norm for the two-dimensional definition of the ghost cells defined by cubic extrapolation in the normal direction as proposed in [1]. The open symbols are the errors in the maximum norm and the solid line is a least square fit with slope −4.38.

as a derivative of a second order accurate solution. As a consequence, the authors had the leeway to compute the jumps in T_x (resp. T_y, T_z) at the point where the interface crosses the x (resp. y, z) axis. That short cut in the velocity computation produces a simple discretization but is not possible in the design of a high order discretization. Moreover, since the level set moves in the normal direction, the velocity must also be constant in the normal direction. Consequently, the velocity at a grid nodes near the interface must be that of the *closest* point to the interface. In addition, in order to construct an overall third order accurate scheme, we must be able to construct a third order accurate velocity.

Suppose that we construct a cubic interpolation $\tilde{T}$ of the temperature around the interface and that the interface position x_I is known to third order accuracy, then the velocity is simply defined as $([\tilde{T}_x], [\tilde{T}_y], [\tilde{T}_z])_\Gamma$. Note that the construction of $\tilde{T}$ is straightforward once the temperature values have been extrapolated across the interface as described in Section 4.1.2, since we then have valid values for the temperature of each domain in both $\phi > 0$ and $\phi \leqslant 0$.

4.2.1. Finding the closest point

There are many ways of finding the closest point to an implicitly defined interface. Here, we follow the work of Chopp [3] since it is based on bi-cubic interpolation and fits well into our framework. Given the level set function, one can identify the cells crossed by the interface by simply checking the sign change of ϕ. For each such cell C with vertices (x_i,y_j), (x_{i+1},y_j), (x_i,y_{j+1}) and (x_{i+1},y_{j+1}), we construct a cubic interpolation $\tilde{\phi}$ of ϕ using the grid nodes of the 3×3 cells centered at C. For each grid node $\vec{P} = (x_i, y_j)$ near the interface,

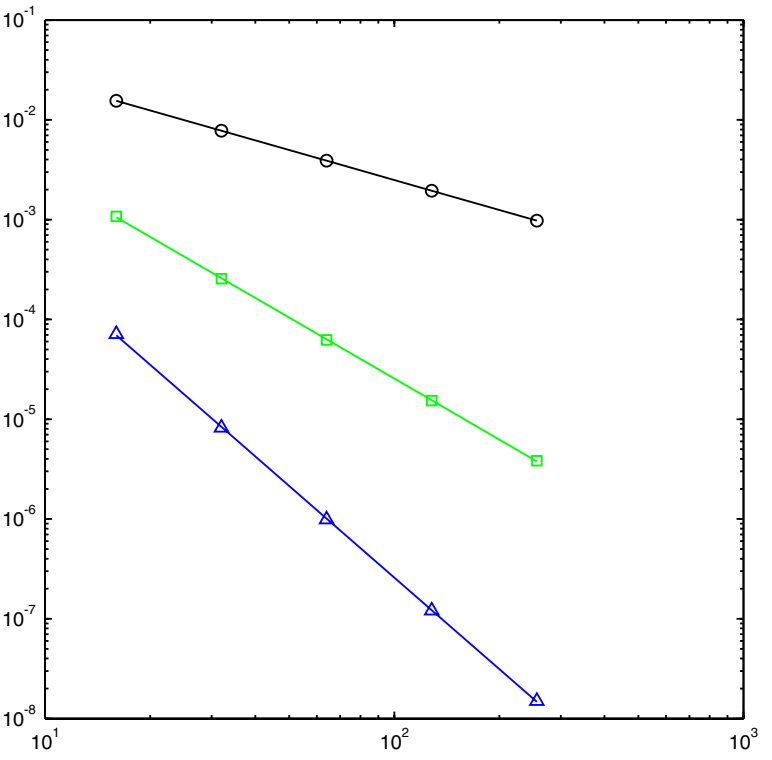


Fig. 11. Error analysis on the effect of perturbing the velocity by a $O(\Delta x)$ (circles), $O(\Delta x^2)$ (squares), and $O(\Delta x^3)$ (triangles) amount in the case of the one-dimensional Stefan problem. The symbols represent the numerical solution on a log–log scale and the solid lines are the least square fits with slopes $-.99$, -2.03 and -3.05.

we seek to find the closest point on the set $S = \{\vec{x} \in \Omega | \tilde{\phi}(\vec{x}) = 0\}$. Chopp notes that such a point $\vec{x}_I$ must satisfy the following two equations:

$$\tilde{\phi}(\vec{x}_I) = 0, \tag{8}$$

$$\vec{\nabla}\tilde{\phi} \times (\vec{P} - \vec{x}_I) = 0, \tag{9}$$

accounting for the fact that $\vec{x}_I$ must be on S and that the normal at this point must be aligned with $\vec{x}_I - \vec{P}$. Then, he proposes an iterative scheme starting with $\vec{x}_I^0 = \vec{P}$ and solving simultaneously (8) and (9) with a Newton-type algorithm:

$$\vec{\delta}_1 = -\tilde{\phi}(\vec{x}_I^k) \frac{\vec{\nabla}\tilde{\phi}(\vec{x}_I^k)}{\vec{\nabla}\tilde{\phi}(\vec{x}_I^k) \cdot \vec{\nabla}\tilde{\phi}(\vec{x}_I^k)},$$

$$\vec{x}_I^{k+1/2} = \vec{x}_I^k + \vec{\delta}_1,$$

$$\vec{\delta}_2 = (\vec{P} - \vec{x}_I^k) - \frac{(\vec{P} - \vec{x}_I^k) \cdot \vec{\nabla}\tilde{\phi}(\vec{x}_I^k)}{\vec{\nabla}\tilde{\phi}(\vec{x}_I^k) \cdot \vec{\nabla}\tilde{\phi}(\vec{x}_I^k)} \vec{\nabla}\tilde{\phi}(\vec{x}_I^k),$$

$$\vec{x}_I^{k+1} = \vec{x}_I^{k+1/2} + \vec{\delta}_2.$$

Convergence is assumed when $\|\delta_1\|^2 + \|\delta_2\|^2 < 10^{-3}\Delta x \Delta y$. Typically five iterations are sufficient to find the closest point to third order accuracy in the case where the interface is smooth. However, since this

algorithm is not guaranteed to converge, the maximum number of iterations is set to 20. Although convergence is not guaranteed, we did not encounter any problems in our computations. The reader is referred to [3] for more details.

4.2.2. A note on the re-initialization equation

For the sake of robustness in the numerics, it is important to keep the values of ϕ close to those of a signed distance function, i.e. $|\nabla\phi| = 1$. The fast marching method [33,28] is a O(nlog(n)) algorithm, where n is the number of grid nodes, but is only first order accurate. In our work, since an O(Δx) perturbation of the interface leads to a first order accurate temperature, we cannot use such a method. Another way of re-initializing ϕ is to solve the re-initialization equation introduced in [32]

$$\phi_\tau + S(\phi_0)(|\nabla\phi| - 1) = 0 \tag{10}$$

for a few steps in fictitious time, τ. This equation is traditionally discretized with the HJ-WENO scheme of [12] in space because it yields less noisy results when computing quantities such as the curvature. However, we note that the order of convergence is only second order accurate as depicted in Fig. 12. This is due to the fact that the re-initialization equation is a Hamilton–Jacobi type equation with discontinuous characteristics near the interface.

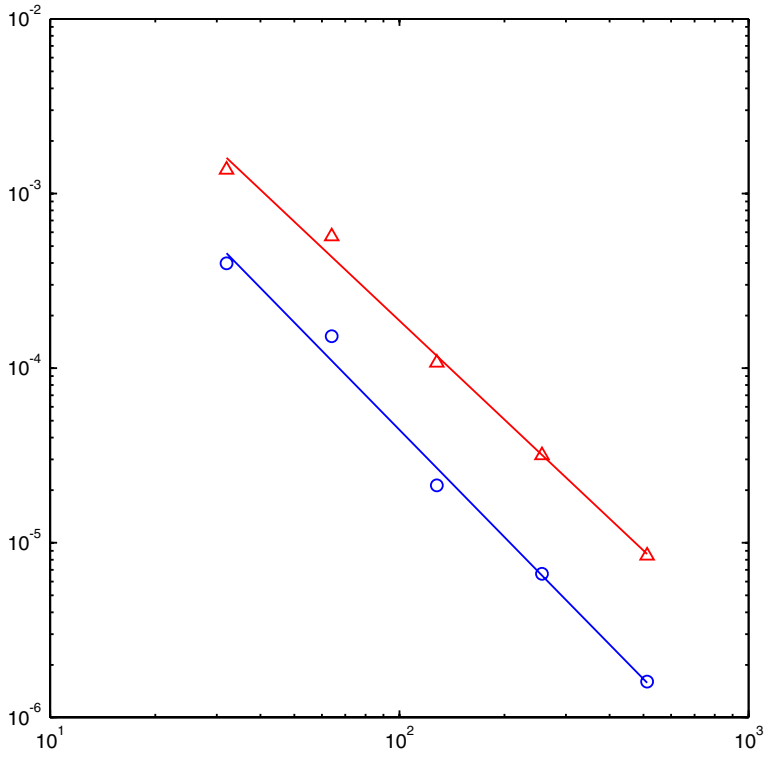


Fig. 12. Error analysis in the L^1 (circles) and L^∞ (triangles) norms for the two-dimensional reinitialization Eq. (10). We use a fifth order WENO discretization in space and a third order TVD Runge–Kutta discretization in time. We take initially the function $e^{\rho - 2.313} - 1$, where $\rho = \sqrt{x^2 + y^2}$. The symbols represent the errors on a log–log scale and the solid lines are the least square fits with slopes -2.04, -1.88, respectively.

Since a $O(\Delta x^2)$ perturbation of the interface can lead to a second order accurate solution for T at best, we do not use Eq. (10) to re-initialize ϕ. Instead, the procedure detailed in Section 4.2.1 gives the distance to the interface to third order accuracy for grid nodes in the neighborhood of the interface.

4.3. A simple example in one spatial dimension

Let $\Omega = [0,1]$ and $\phi = x - .5$ at $t = 0$. We consider an exact solution of $T = e^{t-x+.5} - 1$ on Ω^-. Dirichlet boundary conditions are enforced on the $\partial\Omega$ using the exact solution and we compute the solution at time $t_{\text{final}} = .25$. We use the Crank–Nicholson scheme with $\Delta t \approx \Delta x^2$ and the cubic extrapolation to define the ghost cells. Fig. 13 demonstrates the fourth order accuracy obtained when using the *exact* interface velocity and the third order accuracy obtained when using the *computed* interface velocity.

4.4. Time discretization

In the example of Section 4.3, the interface velocity is constant in time ($[T_{xx}]_{|\Gamma} = 1$). Therefore, this example focused on the spatial discretization and the computation of the velocity, but ignored the details in the time discretization. In fact, consider the Frank sphere solution in one spatial dimension: Let $\Omega = [-1,1]$ with Dirichlet boundary conditions at the domain boundaries. The Frank sphere solution in one spatial dimension describes a slab of radius $R(t) = S_0\sqrt{t}$ parameterized by S_0. The exact solution takes the form

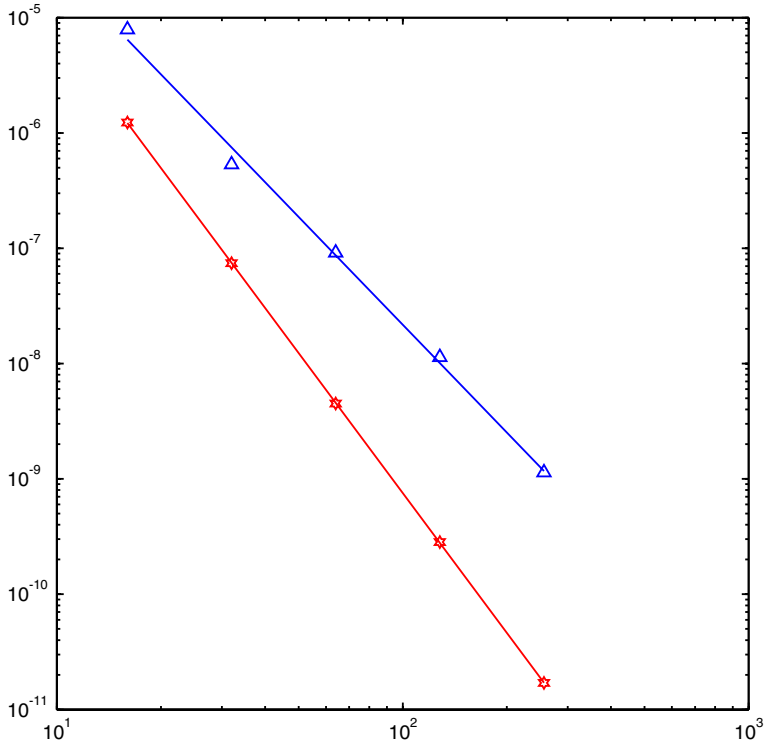


Fig. 13. Error analysis in the L^∞-norm for the one-dimensional Stefan problem on a log–log scale. The ghost cells are defined by cubic extrapolation and we use the Crank–Nicholson scheme with $\Delta t \approx \Delta x^2$. The stars depict the errors when the exact velocity is given and the triangles illustrate the errors when the velocity is computed. The solid lines are the least square fits with slopes -4.03 and -3.10.

$$T = \begin{cases} 0, & s \leqslant S_0, \\ T_\infty \left(1 - \frac{F(s)}{F(S_0)}\right), & s > S_0, \end{cases}$$

where $s = |x|/\sqrt{t}$. T_∞ and S_0 are related by the jump condition $V_n = -D[\nabla T]_{|\Gamma} \cdot \vec{n}$. In one spatial dimension $F(s) = \mathrm{erfc}(s/2)$, with $\mathrm{erfc}(z) = 2\int_z^\infty \mathrm{e}^{-t^2}\,\mathrm{d}t/\sqrt{\pi}$.

We choose the initial time to be $t_{\text{initial}} = 1$ and $T_\infty = -.5$. This defines the initial radius to be $S_0 \approx .86$. We define $\phi = |x| - S_0$ initially and compute the solution at time $t_{\text{final}} = 1.5$. We employ the Crank–Nicholson scheme in time. Since the velocity is third order (hence we are looking at a third order accurate overall solution), we take a time step restriction of $\Delta t \approx \Delta x^{3/2}$ to obtain a third order accurate scheme in time as well. However, the time discretization presented so far yields results that are only second order accurate for time varying velocities, as demonstrated in Fig. 14.

This lower accuracy stems from the lack of consistency in the definition of V^{n+1}. For example approximate the one-dimensional equation

$$\frac{\mathrm{d}\phi}{\mathrm{d}t} = V(\phi),$$

with the Crank–Nicholson scheme. To evolve ϕ from time t^n to time t^{n+1}, we perform the following three steps:

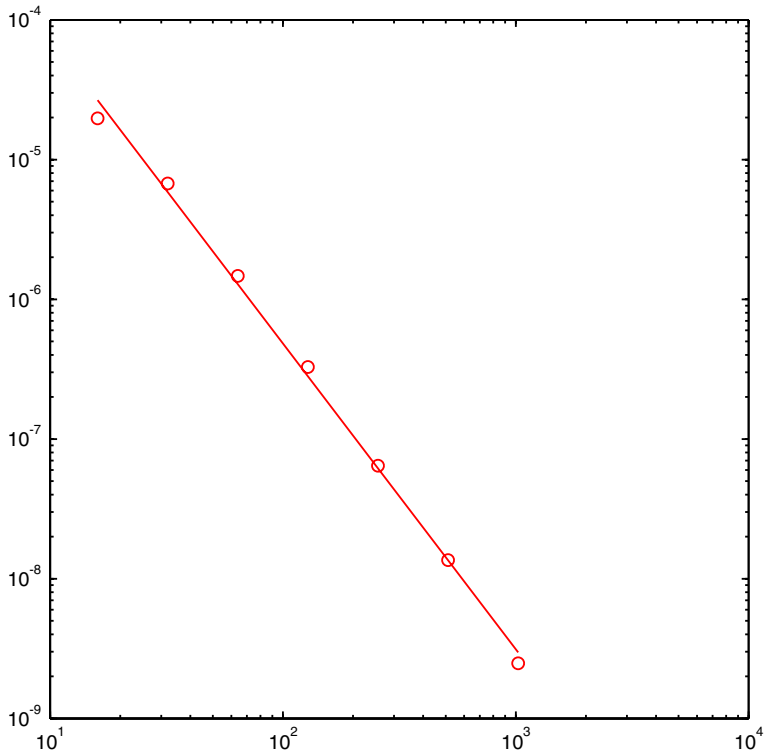


Fig. 14. Error analysis in the L^∞-norm for the one-dimensional Frank sphere solution (the interface velocity is not constant in time). The ghost cells are defined by cubic extrapolation and we use the Crank–Nicholson scheme with $\Delta t \approx \Delta x^{3/2}$. The symbols represent the numerical solution on a log–log scale and the solid line is the least square fit with slope −2.18.

1. Use $V^n(\phi^n)$ to evolve ϕ^n to ϕ^{n+1}_{temp} with an Euler step.
2. Use $V^{n+1}(\phi^{n+1}_{\text{temp}})$ to evolve ϕ^{n+1}_{temp} to ϕ^{n+2} with an Euler step.
3. Define $\phi^{n+1} = (\phi^n + \phi^{n+2})/2$.

In the case of the Stefan problem, the velocity at time t^{n+1} is defined from the jump in the temperature gradient at the interface at time t^{n+1} and thus needs to be consistent. That is, if $V^{n+1} = V^{n+1}(\phi^{n+1})$, then the V^{n+1} from step 2 above needs to be consistent with the ϕ^{n+1} computed in step 3. To solve this problem we iterate steps 2 and 3 in order to guarantee that the velocity at time t^{n+1} is consistent, i.e $V^{n+1} = V^{n+1}(\phi^{n+1}) = V((\phi^n + \phi^{n+2})/2)$. We iterate until the magnitude of the error in this last equation is less that 10^{-8}. We note that very few iterations are needed. We use the Crank–Nicholson scheme with a time step restriction of $\Delta t \approx \Delta x^{3/2}$ and we perform the iteration described above. Fig. 15 demonstrates that such a time discretization yields a third order accurate solution.

4.5. Example in two spatial dimensions

Consider the Stefan problem in a domain $[-1,1]\times[-1,1]$ with Dirichlet boundary conditions at the domain boundaries. The two spatial dimensions Frank sphere solution describes a disk of radius $R(t) = S_0\sqrt{t}$ parameterized by S_0. The exact solution takes the form

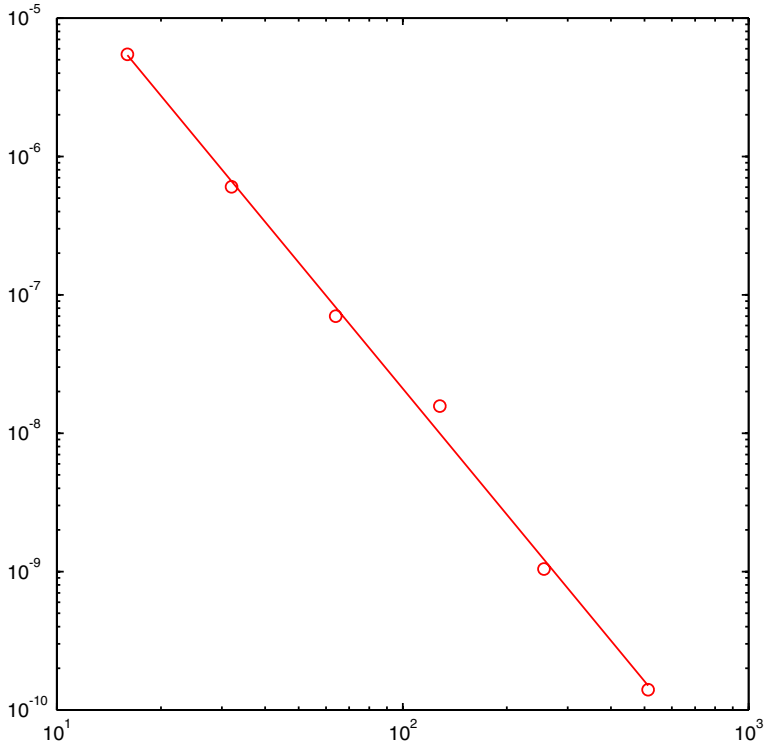


Fig. 15. Error analysis in the L^∞-norm for the one-dimensional Frank sphere solution (i.e. the interface velocity is not constant in time). The ghost cells are defined by cubic extrapolation and we use the Crank–Nicholson scheme with $\Delta t \approx \Delta x^{3/2}$. The time discretization involves iterating on the velocity as described in Section 4.4. The symbols represent the numerical solution on a log–log scale and the solid line is the least square fit with slope −3.02.

$$T = \begin{cases} 0, & s \leqslant S_0, \\ T_\infty \left(1 - \frac{F(s)}{F(S_0)}\right), & s > S_0, \end{cases}$$

where $s = |x|/\sqrt{t}$, and with T_∞ and S_0 related by the jump condition $V_n = -D[\nabla T]_{|\Gamma} \cdot \vec{n}$. In two spatial dimensions $F(s) = E_1(s^2/4)$, with $E_1(z) = \int_z^\infty e^{-t}/t \, dt$.

Choose the initial time to be $t_{\text{initial}} = 1$ and the initial radius to be $S_0 = .5$ ($\phi = |x| - S_0$). This defines $T_\infty \approx -.15$. Compute the solution at time $t_{\text{final}} = 2.89$ with the ghost cells defined by cubic extrapolation. The Frank sphere problem being ill-posed, we choose the final time large enough to allow the interface to cross a large amount of grid cells (about 50), demonstrating the built-in regularization inherent to level set methods. Table 2 presents the accuracy results.

For the sake of comparison, Table 3 offers the convergence results obtained with the symmetric discretization from [9]. We note that the present method and 16 grid nodes yields the same accuracy as in [9] with 128 grid nodes. Likewise, [9] would require 1080 points to obtain the same accuracy as in the present method and 32 points. This may have a significant impact especially in three spatial dimensions. In fact, even when utilizing adaptive grid refinement, this newly proposed method can drastically lower the number of grid nodes needed to represent thin dendrites while retaining the desired accuracy. Fig. 18 illustrates the comparison between the method presented in [9] and the algorithm described in Section 4. Fig. 16 depicts the interface evolution at several times and the cross-section of the temperature at initial (left) and final (right) times is shown in Fig. 17.

4.6. Modified Stefan problem

The Gibbs–Thomson interface condition can be used to account for the deviation of the interface temperature from equilibrium. Such boundary condition reads $T_I = -\epsilon_c \kappa - \epsilon_v V_n$, where κ is the curvature of the front, ϵ_c the surface tension coefficient, ϵ_v the molecular kinetic coefficient and V_n the interface velocity.

Table 2
Accuracy results for the algorithm described in Section 4

Number of points	L^1-error	Order	L^∞-error	Order
16×16	3.204×10^{-4}	–	1.032×10^{-3}	–
32×32	1.092×10^{-5}	4.87	6.954×10^{-5}	3.89
64×64	7.369×10^{-7}	3.89	3.482×10^{-6}	4.32
128×128	8.836×10^{-8}	3.06	3.149×10^{-7}	3.47
256×256	1.168×10^{-8}	2.92	4.424×10^{-8}	2.83

Table 3
Accuracy results for the algorithm described in [9]

Number of points	L^1-error	Order	L^∞-error	Order
16×16	8.047×10^{-4}	–	2.709×10^{-3}	–
32×32	4.384×10^{-4}	0.876	1.528×10^{-3}	0.826
64×64	1.874×10^{-4}	1.23	9.724×10^{-4}	0.652
128×128	9.606×10^{-5}	0.964	5.500×10^{-4}	0.822
256×256	4.723×10^{-5}	1.02	2.822×10^{-4}	0.963

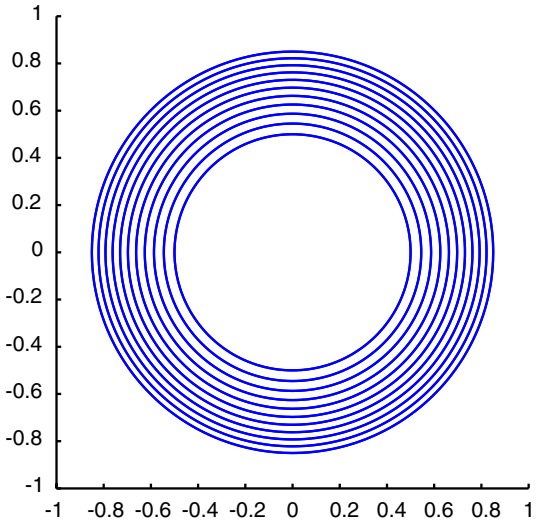


Fig. 16. Interface evolution in the case of the two-dimensional Frank sphere solution.

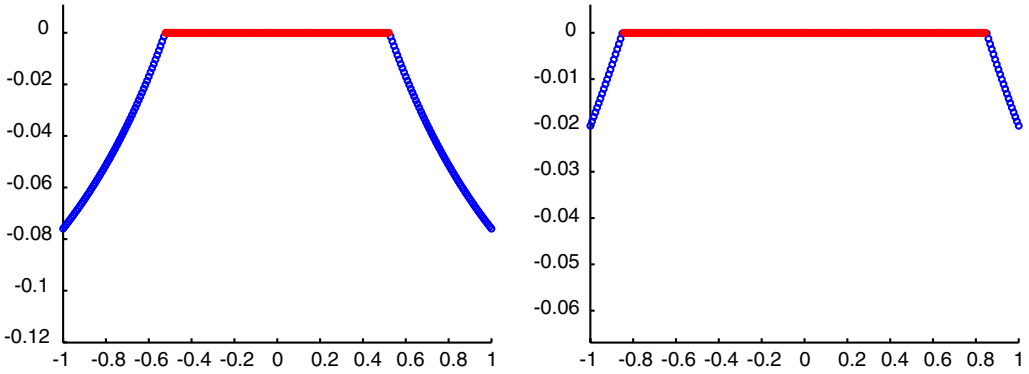


Fig. 17. Cross-section of the two-dimensional Frank sphere solution at $t_{\text{initial}} = 1$ (left) and $t_{\text{final}} = 2.89$ (right).

Anisotropic surface tension effects can be included in the coefficient ϵ_c. For example one can take in two spatial dimensions $\epsilon_c = (1 - 15\epsilon\cos(4\alpha))$ where ϵ is the anisotropy strength and α is the angle between the normal at the interface and the x-axis.

In [9] the Gibbs–Thomson relation is computed at every grid point neighboring the interface and then linearly interpolated to the front. In that case, the computation of the interface curvature is performed with standard second order accurate central differencing. Such computations for the curvature, which do not hinder the first order accuracy of the method presented in [9], cannot be used in this present work without lowering the third order accuracy of our method. As a consequence, more research on high order accurate curvature must first be performed before considering the more general Gibbs–Thomson case. We note that ideas based on [31] could be used and we leave this research as future work.

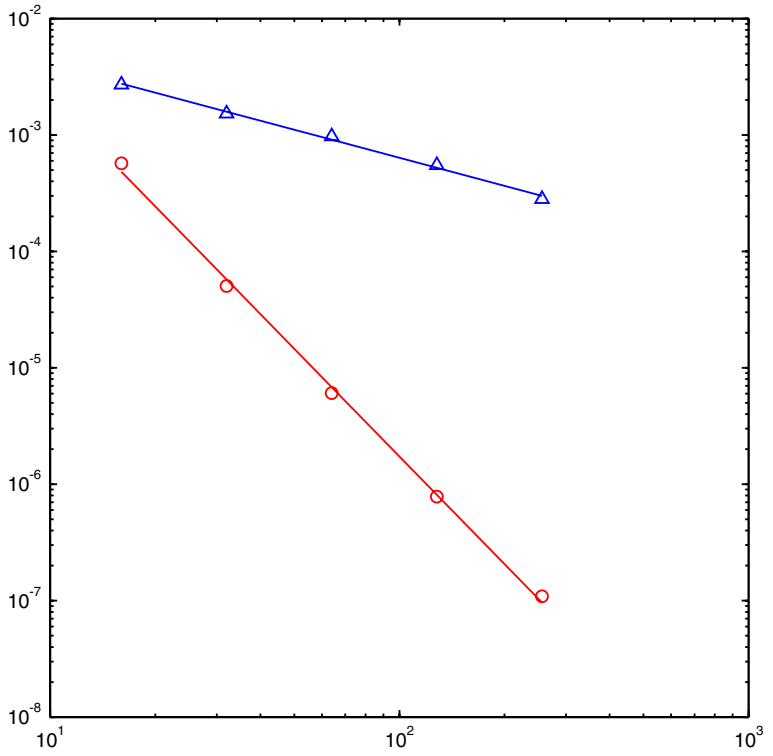


Fig. 18. Error analysis in the L^∞-norm for the two-dimensional Frank sphere solution. The triangles illustrate the accuracy obtained with the scheme presented in [9] and the circles represent the accuracy obtained with the algorithm presented in Section 4. The open symbols are the errors in the maximum norm and the solid line is a least square fit with slope $-.80$ and -3.07, respectively.

5. Conclusions

We have proposed a simple finite difference algorithm for obtaining fourth order accurate solutions for the Laplace equation on arbitrary domains. We also designed a fourth order scheme for the heat equation with Dirichlet boundary conditions on an irregular domain. In the case of the heat equation, we utilize an implicit time discretization to overcome the drastic time step restrictions associated with explicit schemes. We then constructed a third order accurate method for the Stefan problem. We presented multidimensional results to demonstrate the accuracy in the L^∞-norm. Notably, we remark that in two spatial dimensions, one can obtain six digits of accuracy (for the Laplace and heat equations, five digits of accuracy for a Stefan problem) on very coarse grids of 32 grid nodes in each spatial dimension. Therefore, even though the discretization yields a non-symmetric linear system, the ability of this algorithm to perform well on very coarse grid makes it exceptionally efficient. Future work on this subject will include the use of adaptive mesh refinement techniques and will focus on the modified Stefan problem where molecular kinetics and surface tension are included.

References

[1] T. Aslam, A partial differrential equation approach to multidimensional extrapolation, J. Comput. Phys. 193 (2004) 349–355.
[2] S. Chen, B. Merriman, S. Osher, P. Smereka, A simple level set method for solving Stefan problems, J. Comput. Phys. 135 (1997) 8–29.

[3] D. Chopp, Some improvement of the Fast Marching Method, SIAM J. Sci. Comput 23 (2001) 230–244.
[4] T. Chong, A variable mesh finite difference method for solving a class of parabolic differential equations in one space variable, SIAM J. Numer. Anal 15 (1978) 835–857.
[5] A. Chorin, Curvature and solidification, J. Comput. Phys. 58 (1985) 472–490.
[6] R. Fedkiw, T. Aslam, B. Merriman, S. Osher, A non-oscillatory Eulerian approach to interfaces in multimaterial flows (The Ghost Fluid Method), J. Comput. Phys. 152 (1999) 457–492.
[7] W. George, J. Warren, A parallel 3D dendritic growth simulator using the phase-field method, J. Comput. Phys. 177 (2002) 264–283.
[8] F. Gibou, R. Fedkiw, R. Caflisch, S. Osher, A level set approach for the simulation of dendritic growth, J. Sci. Comput. 19 (2003) 183–199.
[9] F. Gibou, R. Fedkiw, L.-T. Cheng, M. Kang, A second order accurate symmetric discretization of the Poisson equation on irregular domains, J. Comput. Phys. 176 (2002) 1–23.
[10] Y.-T. Kim, N. Goldenfeld, J. Dantzig, Computation of dendritic microstructures using a level set method, Phys. Rev. E 62 (2000) 2471–2474.
[11] J. Hoffman, Relationship between the truncation errors of centered finite-difference approximations on uniform and nonuniform meshes, J. Comput. Phys. 46 (1982) 469–474.
[12] G.-S. Jiang, D. Peng, Weighted ENO schemes for Hamilton Jacobi equations, SIAM J. Sci. Comput. 21 (2000) 2126–2143.
[13] G.-S. Jiang, C.-W. Shu, Efficient implementation of weighted ENO schemes, J. Comput. Phys. 126 (1996) 202–228.
[14] H. Johansen, Cartesian grid embedded boundary finite difference methods for elliptic and parabolic differential equations on irregular domains, Ph.D. Thesis, University of California, Berkeley, CA, 1997.
[15] H. Johansen, P. Colella, A Cartesian grid embedded boundary method for Poisson's equation on irregular domains, J. Comput. Phys. 147 (1998) 60–85.
[16] D. Juric, G. Tryggvason, A front tracking method for dendritic solidification, J. Comput. Phys. 123 (1996) 127–148.
[17] A. Karma, W.-J. Rappel, Quantitative phase-field modeling of dendritic growth in two and three dimensions, Phys. Rev. E 57 (1997) 4323–4349.
[18] H.-O. Kreiss, T. Manteuffel, B. Swartz, B. Wendroff, A. White, Supra-convergent schemes on irregular grids, Math. Comput. 47 (176) (1986) 537–554.
[19] J. Lambert, Numerical Methods for Ordinary Differential Systems, Wiley, New York, 1993.
[20] R. LeVeque, Z. Li, The immersed interface method for elliptic equations with discontinuous coefficients and singular sources, SIAM J. Numer. Anal. 31 (1994) 1019–1044.
[21] X.-D. Liu, S. Osher, T. Chan, Weighted essentially non-oscillatory schemes, J. Comput. Phys. 126 (1996) 200–212.
[22] T. Manteuffel, A. White, The numerical solution of second-order boundary value problems on nonuniform meshes, Math. Comput. 47 (176) (1986) 511–535.
[23] S. Osher, R. Fedkiw, Level Set Methods and Dynamic Implicit Surfaces, Springer, New York, 2002.
[24] C. Peskin, Numerical analysis of blood flow in the heart, J. Comput. Phys. 25 (1977) 220–252.
[25] M. Plapp, A. Karma, Multiscale finite-difference-diffusion-Monte-Carlo method for simulating dendritic solidification, J. Comput. Phys. 165 (2000) 592–619.
[26] A. Schmidt, Computation of three dimensional dendrites with finite elements, J. Comput. Phys. 125 (1996) 293–312.
[27] Y. Saad, Iterative Methods for Sparse Linear Systems, PWS Publishing Company, Boston, 1996.
[28] J. Sethian, A fast marching level set method for monotonically advancing fronts, Proc. Natl. Acad. Sci. USA 93 (1996) 1591–1595.
[29] J. Sethian, Level Set Methods and Fast Marching Methods, Cambridge University Press, Cambridge, 1999.
[30] J. Sethian, J. Strain, Crystal growth and dendritic solidification, J. Comput. Phys. 98 (1992) 231–253.
[31] M. Sussman, M.Y. Hussaini, A discontinuous spectral element method for the level set equation, J. Sci. Comput. 19 (2003) 479–500.
[32] M. Sussman, P. Smereka, S. Osher, A level set approach for computing solutions to incompressible two-phase flow, J. Comput. Phys. 114 (1994) 146–159.
[33] J. Tsitsiklis, Efficient algorithms for globally optimal trajectories, IEEE Trans. Autom. Contr. 40 (1995) 1528–1538.
[34] H. Udaykumar, R. Mittal, W. Shyy, Computation of solid–liquid phase fronts in the sharp interface limit on fixed grids, J. Comput. Phys. 153 (1999) 535–574.
[35] P. Zhao, J. Heinrich, Front tracking finite element method for dendritic solidification, J. Comput. Phys. 173 (2001) 765–796.