

A Fourth Order Accurate Discretization for the Laplace and Heat Equations on Arbitrary Domains, with Applications to the Stefan Problem *

Frédéric Gibou[†]

Ronald Fedkiw[‡]

April 27, 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 time discretization to avoid the stringent time step restrictions associated with 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.

*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, NSF DMS-0106694 and NSF ACI-0323866. In addition, the first author was supported in part by an NSF postdoctoral fellowship (DMS-0102029).

[†]Mathematics Department & Computer Science Department, Stanford University, Stanford, CA 94305.

[‡]Computer Science Department, Stanford University, Stanford, CA 94305.

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. 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. Moreover, the explicit treatment of connectivity makes the method challenging to extend to three spatial dimensions. Implicit representations such as the level set method [24, 30] or the phase-field method [17] embed 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. [17, 7]). 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 [24, 30]. The earliest level set method for solidification type problems was presented in [31] 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. [10] showed that this discretization produces results in accordance with solvability theory. In [35], the authors discretized the heat equation on a Cartesian grid in a manner quite similar to that proposed in [2] resulting in a nonsymmetric linear system when applying implicit time discretization. [35] 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 [25] 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 nonsymmetric discretization matrix. Both multigrid methods and adaptive mesh refinement were used, and in [14] this nonsymmetric discretization was coupled to a volume of fluid front tracking method in order to solve the Stefan problem. In [27] 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 [26]. Recently, [36] 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 to three spatial dimensions. Moreover, the velocity was computed under the assumption that the interface cuts the element in a straight line. The interested reader is referred to [16], [2], [8] and the references therein for an extensive summary of computational results for the Stefan problem.

Standard convergence proofs use stability and consistency analysis to imply convergence, i.e. given stability, a *sufficient* condition for a scheme to be p^{th} order accurate is that the local truncation error is p^{th} order. However, a p^{th} order local truncation error is not a *necessary* condition and one can derive p^{th} order accurate schemes despite the fact that their local truncation error is of lower order. [23] (see also [19]) made this point in the context of second order, scalar boundary value problems on nonuniform 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 accurate 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 accurate central scheme limiting the overall solution to second order accuracy. However, their 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. [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 and heat equations on

irregular domains. Moreover, their discretization had the added benefit of yielding a symmetric linear system as opposed to the nonsymmetric 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 equation 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, $\Omega \subset R^n$, 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

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

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 function 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 Equation

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:

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

For each unknown, T_i , equation (2) is used to fill in one row of a matrix creating a linear system of equations. This discretization is valid if all the nodal 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 figure 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_{i+1}^G and T_{i+2}^G constructed by extrapolating the values of T across the interface. The discretization is then rewritten as

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

More precisely, we first construct an interpolant $\tilde{T}(x)$ of $T(x)$ on the left of the interface with $\tilde{T}(0) = T_i$, and then we define $T_{i+1}^G = \tilde{T}(\Delta x)$ and $T_{i+2}^G = \tilde{T}(2\Delta x)$. Figure 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 extrapolation defined by:

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

- $d = T_I$.

Linear extrapolation: Take $\tilde{T}(x) = cx + d$ with

- $\tilde{T}(0) = T_i$,
- $\tilde{T}(\theta\Delta x) = T_I$.

Quadratic extrapolation: Take $\tilde{T}(x) = bx^2 + cx + d$ with

- $\tilde{T}(-\Delta x) = T_{i-1}$,

- $\tilde{T}(0) = T_i$,
- $\tilde{T}(\theta\Delta x) = T_I$.

Cubic extrapolation: Take $\tilde{T}(x) = ax^3 + bx^2 + cx + d$ with

- $\tilde{T}(-2\Delta x) = T_{i-2}$,
- $\tilde{T}(-\Delta x) = T_{i-1}$,
- $\tilde{T}(0) = T_i$,
- $\tilde{T}(\theta\Delta x) = T_I$.

In these equations $\theta = (x_I - x_i)/\Delta x$.

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 $T_i^G = \tilde{T}(\Delta x)$ and $T_{i-1}^G = \tilde{T}(2\Delta x)$ 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} .

The construction of \tilde{T} is both limited by the number of points in the domain and by how close the interface is to a grid node. The latter restriction stems from the fact that as $\theta \rightarrow 0$, 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 $\tilde{T}(\Delta x + \theta\Delta x) = T_I$ instead of $\tilde{T}(0) = T_i$ and $\tilde{T}(\theta\Delta x) = T_I$. Then the ghost nodes are defined as $T_{i+1}^G = \tilde{T}(2\Delta x)$ and $T_{i+2}^G = \tilde{T}(3\Delta x)$. Higher order extrapolation follows similarly. Finally, we note that one can lower the degree of the interpolant in order to preserve the pentadiagonal structure of the linear system in the case where the stencil has been shifted.

For the constant and the linear extrapolations, the matrix entries to the right of the diagonal for the i -th 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 the preconditioned conjugate gradient (PCG) method (see, e.g. [28]). Moreover the corresponding matrix is strictly diagonally dominant, and therefore nonsingular. For higher order extrapolations, the linear system is nonsymmetric and not necessarily strictly diagonally dominant, but we can still develop high order accurate methods. The overall accuracy for T and the nature of the resulting linear system is determined by the degree of the interpolation which is summarized in table 1. Finally, we note that [21] designed a method that also yields a nonsymmetric linear system, but which is only second order accurate.

Degree of Extrapolation	Order of Accuracy	Linear System
Constant	First	Symmetric
Linear	Second	Symmetric
Quadratic	Third	Nonsymmetric
Cubic	Fourth	Nonsymmetric

Table 1: Order of accuracy and nature of the linear system.

We illustrate our methodology with 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 $\phi = x - .5$ (so that the interface never falls on a grid node). Dirichlet boundary conditions are enforced on the $\partial\Omega$ using the exact solution. Tables 2-5 give the error between the numerical solution and the exact solution in the L^1 and L^∞ -norms. These same results are also presented on a log-log plot in figure 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 extrapolation, respectively. We use a PCG method with an incomplete Cholesky preconditioner for the symmetric linear systems, and the BiCGSTAB method (see e.g. [28]) otherwise.

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}	.98
64	3.057×10^{-2}	1.03	6.018×10^{-2}	.99
128	1.512×10^{-2}	1.02	3.020×10^{-2}	1.00

Table 2: Accuracy results for the one dimensional Laplace equation with ghost cells defined by constant extrapolation.

2.2 2D Laplace Equation

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

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

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

Table 3: Accuracy results for the one dimensional Laplace equation with ghost cells defined by linear 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

Table 4: Accuracy results for the one dimensional Laplace equation with ghost cells defined by quadratic extrapolation.

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. 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 $\widetilde{\phi^x}$ of ϕ in the x -direction and find the interface location x_I by solving $\widetilde{\phi^x}(x_I) = 0$. 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 with the following example. Let $\Omega = [-1, 1] \times [-1, 1]$ with an exact solution of $T = \sin(\pi x) + \sin(\pi y) + \cos(\pi x) +$

Number of Points	L^1 - error	order	L^∞ - 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

Table 5: Accuracy results for the one dimensional Laplace equation with ghost cells defined by cubic extrapolation.

$\cos(\pi y) + x^6 + y^6$ 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 $\alpha \in [0, 2\pi]$. Figure 3 depicts the solution on a 64×64 grid, and figure 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 figure 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.

3 Heat Equation

Consider a Cartesian computational domain, $\Omega \subset R^n$, 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

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

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

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 $0 < \theta = (x_I - x_i) / \Delta x \leq 1$ 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. Thus, we use implicit time discretization. In particular, we choose the Crank-Nicholson scheme and impose a time step restriction of $\Delta t = c\Delta x^2$ with $0 < c < 1$ 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 used instead in order to relax the time step restriction to $\Delta t = c\Delta x$. For more on numerical methods for ordinary differential equations, see [20].

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 $f^n = \left(I + \frac{\Delta t}{2}A^n(\Delta)\right)T^n$ at time t^n using the methodology of section 2 to define the ghost cells in a dimension by dimension fashion. 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 Equation

In one spatial dimension, 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 $A^{n+1}(T_{xx})$ 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_{xx}^n with the fourth order accurate discretization of

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

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 are left to solve

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

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

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

For each unknown, T_i^{n+1} , equations (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 again 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 extrapolation, 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. Figure 5 illustrates the fourth order accuracy in the L^∞ -norm.

3.2 2D Heat Equation

The algorithm described above extends readily to two and three spatial dimensions. The approximations of T_{xx} and T_{yy} are performed independently 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 $\alpha \in [0, 2\pi]$. Figure 6 depicts the solution on a 64×64 grid and figure 7 demonstrates the fourth order accuracy in the L^∞ -norm.

4 Stefan Problem

Consider again a Cartesian computational domain, $\Omega \subset R^n$, 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 possibly 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 . Dirichlet boundary conditions are specified on $\partial\Omega$.

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

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

The velocity components are defined by the x , y and z projections of the jump in the temperature gradient, i.e. $(u, v, w) = ([DT_x]_\Gamma, [DT_y]_\Gamma, [DT_z]_\Gamma)$. The level set advection equation is discretized with a HJ-WENO scheme [12], see also [22, 13]. For more details on the level set method, see e.g. [24, 30].

Consider the Crank-Nicholson framework:

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

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 $f^n = T^n + \frac{\Delta t}{2} A^n(\Delta) T^n$.

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 Extrapolation in the Normal Direction

We follow along the lines of section 3 discretizing 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 moving domain provides added difficulty. Figure 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 in the figure). Since we need to evaluate the right hand side at these new nodes, valid values for T^n must be defined there. Moreover, since the interface evolves in the normal direction, valid values of T^n are needed in more than just the Cartesian directions. Therefore, a high order extrapolant must be defined in the *normal* direction at the interface and such a procedure must be straightforward to implement in one, two and three spatial dimensions.

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 $\phi \leq 0$ to the region where $\phi > 0$. In the case of cubic extrapolation, we first compute $T_{nnn} = \vec{\nabla} \left(\vec{\nabla} \left(\vec{\nabla} T \cdot \vec{n} \right) \cdot \vec{n} \right) \cdot \vec{n}$ in the region $\phi \leq 0$ 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 $\phi \geq -band$. For example, we set $band = 2\sqrt{dx^2 + dy^2}$ 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

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

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

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

defining T in such a way that its normal derivative is equal to T_n . These equations 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.

Figure 9 illustrates 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 $\vec{x} \in \Omega^-$. Figure 9 (top) illustrates contours of T after it has been extrapolated across the interface, and figure 9 (bottom) depicts contours of the exact solution for comparison. For the sake of clarity, we have extrapolated T in the entire region in this example, but 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 figure 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. Consequently, the velocity will only be third order accurate as it involves gradients of the temperature. Not surprisingly, this limits the overall order of accuracy to three. For example, let $\Omega = [0, 1]$ with an exact solution of $T = e^{t-x+.5} - 1$ on Ω^- with $\phi = x - .5$ 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 $\Delta t \approx \Delta x^2$ for fourth order time accuracy, and use cubic extrapolation to define the ghost values. In a sequence of tests, we use the *exact* interface velocity perturbed by an $O(\Delta x^3)$, $O(\Delta x^2)$ and $O(\Delta x)$ amount. Figure 11 illustrates the fact that a third, second and first order accurate velocity produces overall results of the same order.

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 as a derivative of a second order accurate solution. As a consequence, the authors had the leeway to compute the jumps in T_x (respectively T_y and T_z) at the point where the interface crosses the x (respectively y and z) axis. That short cut in the velocity computation produces a simple discretization, but it cannot readily be extended to higher order accuracy. Moreover, since the level set moves in the normal direction, the velocity should be constant in the normal direction. That is, the velocity at a grid node near the interface should be equal to that of the *closest* point on the interface. In addition, in order to construct an overall third order accurate scheme, we require 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 defined as $([\tilde{T}_x]_\Gamma, [\tilde{T}_y]_\Gamma, [\tilde{T}_z]_\Gamma)$. The construction of \tilde{T} is straightforward once the temperature values have been extrapolated across the interface as described in section 4.1, since we then have valid values for the temperature of each domain in both $\phi > 0$ and $\phi \leq 0$.

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 [3] since it is based on bi-cubic interpolation that 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, we seek to find the closest point on the set $S = \{\vec{x} \in \Omega | \tilde{\phi}(\vec{x}) = 0\}$. [3] 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 [3] proposes an iterative scheme starting with $\vec{x}_I^0 = \vec{P}$ and solving simultaneously equations (8) and (9) with

a Newton-type algorithm:

$$\begin{aligned}
\vec{\delta}_1 &= -\tilde{\phi}(\vec{x}_I^k) \frac{\vec{\nabla} \tilde{\phi}(\vec{x}_I^k)}{\vec{\nabla} \tilde{\phi}(\vec{x}_I^k) \cdot \vec{\nabla} \tilde{\phi}(\vec{x}_I^k)}, \\
\vec{x}_I^{k+1/2} &= \vec{x}_I^k + \vec{\delta}_1, \\
\vec{\delta}_2 &= (\vec{P} - \vec{x}_I^k) - \frac{(\vec{P} - \vec{x}_I^k) \cdot \vec{\nabla} \tilde{\phi}(\vec{x}_I^k)}{\vec{\nabla} \tilde{\phi}(\vec{x}_I^k) \cdot \vec{\nabla} \tilde{\phi}(\vec{x}_I^k)} \vec{\nabla} \tilde{\phi}(\vec{x}_I^k), \\
\vec{x}_I^{k+1} &= \vec{x}_I^{k+1/2} + \vec{\delta}_2.
\end{aligned}$$

Convergence is assumed when $\sqrt{\|\delta_1\|^2 + \|\delta_2\|^2} < 10^{-3} \Delta x \Delta y$. Typically, five iterations are sufficient to find the closest point to third order accuracy in the case where the interface is smooth. However, we use 20 iterations just to be safe. 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 Reinitialization 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. $|\nabla \phi| = 1$. Since the Fast Marching Method [34, 29] is only first order accurate, and an $O(\Delta x)$ perturbation of the interface leads to a first order accurate temperature, we cannot use this method. Another way of reinitializing ϕ is to solve the reinitialization equation introduced in [33]

$$\phi_\tau + S(\phi_o) (|\nabla \phi| - 1) = 0 \quad (10)$$

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 this method is only second order accurate as depicted in figure 12. This is due to the fact that the reinitialization equation is a Hamilton-Jacobi type equation with discontinuous characteristics emanating from the interface. Since a $O(\Delta x^2)$ perturbation of the interface leads to a second order accurate solution, we do not use equation (10) to reinitialize 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.

4.2.3 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 cubic extrapolation to define the ghost cells. Figure 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.

4.3 Time Discretization

In the example in section 4.2.3, the interface velocity is constant in time, i.e. $[T_x]_{\Gamma} = 1$. Therefore, this example focused on the spatial discretization and velocity computation, but was not sensitive to the time discretization details.

Consider the Frank spheres solution in one spatial dimension. Let $\Omega = [-1, 1]$ with Dirichlet boundary conditions at the domain boundaries. The Frank spheres 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

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

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 $F(s) = \text{erfc}(s/2)$, with $\text{erfc}(z) = 2 \int_z^{\infty} e^{-t^2} dt/\sqrt{\pi}$. We choose $t_{\text{initial}} = 1$ and $T_{\infty} = -.5$ obtaining an initial radius of $S_0 \approx .86$. Initially, we set $\phi = |x| - S_0$ and compute the solution at time $t_{\text{final}} = 1.5$ with the Crank-Nicholson scheme. Since the velocity is third order accurate (hence we expect a third order accurate method), we choose 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 figure 14.

This lower accuracy stems from the lack of consistency in the definition of V^{n+1} . For example, consider approximating

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

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

1. Use $V^n(\phi^n)$ to evolve ϕ^n to ϕ_{temp}^{n+1} with an Euler step.
2. Use $V^{n+1}(\phi_{\text{temp}}^{n+1})$ to evolve ϕ_{temp}^{n+1} to ϕ^{n+1} 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 than 10^{-8} noting that very few iterations are needed. Figure 15 demonstrates that this time discretization yields a third order accurate solution.

4.4 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 Frank spheres in two spatial dimensions describes a disk of radius $R(t) = S_0\sqrt{t}$ parameterized by S_0 . The exact solution takes the form

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

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 two spatial dimensions, $F(s) = E_1(s^2/4)$ with $E_1(z) = \int_z^\infty e^{-t}/t dt$. We take $t_{initial} = 1$ and $S_0 = .5$ ($\phi = |x| - S_0$). This defines $T_\infty \approx -.15$. We compute the solution at time $t_{final} = 2.89$ with the ghost cells defined via cubic extrapolation. The Frank spheres 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 6 presents the accuracy results.

For the sake of comparison, table 7 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 the present method with 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. Figure 18 illustrates the comparison between the method

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 6: Accuracy results for the algorithm described in section 4.

presented in [9] and the algorithm described in section 4. Figure 17 depicts the interface evolution at several times (top), and the cross section of the temperature at initial (bottom left) and final (bottom right) times.

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

Table 7: Accuracy results for the algorithm described in [9].

5 Conclusions and Future Work

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 accurate 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 stringent 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 6 digits of accuracy (for the Laplace and heat equations, 5 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 nonsymmetric

linear system, the ability of this algorithm to perform well on a very coarse grid makes it exceptionally efficient. Future work on this subject will include the use of adaptive mesh refinement techniques.

The Gibbs-Thompson interface condition can be used to account for the deviation of the interface temperature from equilibrium with $T_I = -\epsilon_c \kappa - \epsilon_v V_n$, 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. Anisotropic surface tension effects can be included in the coefficient ϵ_c . For example, in two spatial dimensions, one can take $\epsilon_c = (1 - 15\epsilon \cos(4\alpha))$ 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-Thompson relation was computed at every grid point neighboring the interface and then linearly interpolated to the front. In that case, the computation of the interface curvature was 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 discretizations must first be performed before considering the more general Gibbs-Thompson case. See, for example, the work of [32].

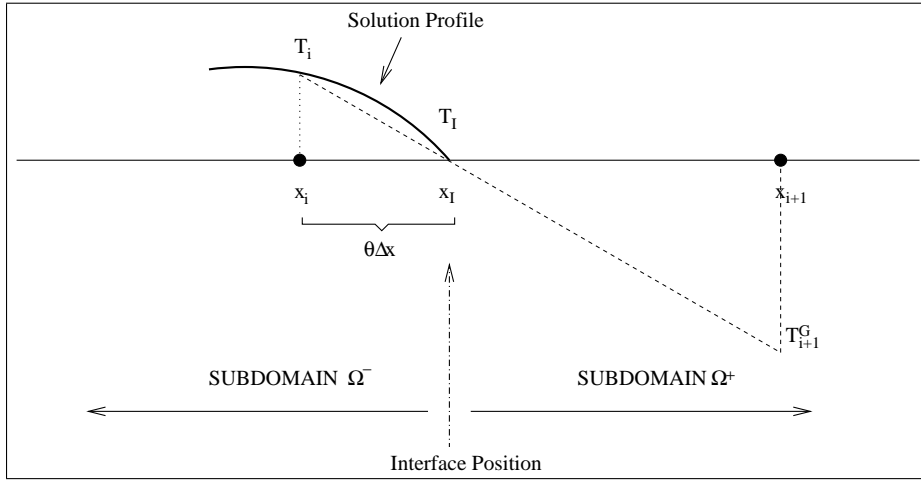


Figure 1: Definition of the ghost cells with linear extrapolation. First, we construct a linear interpolant $\tilde{T}(x) = ax + b$ of T such that $\tilde{T}(0) = T_i$ and $\tilde{T}(\theta\Delta x) = T_I$. Then we define $T_{i+1}^G = \tilde{T}(\Delta x)$ (likewise, $T_{i+2}^G = \tilde{T}(2\Delta x)$).

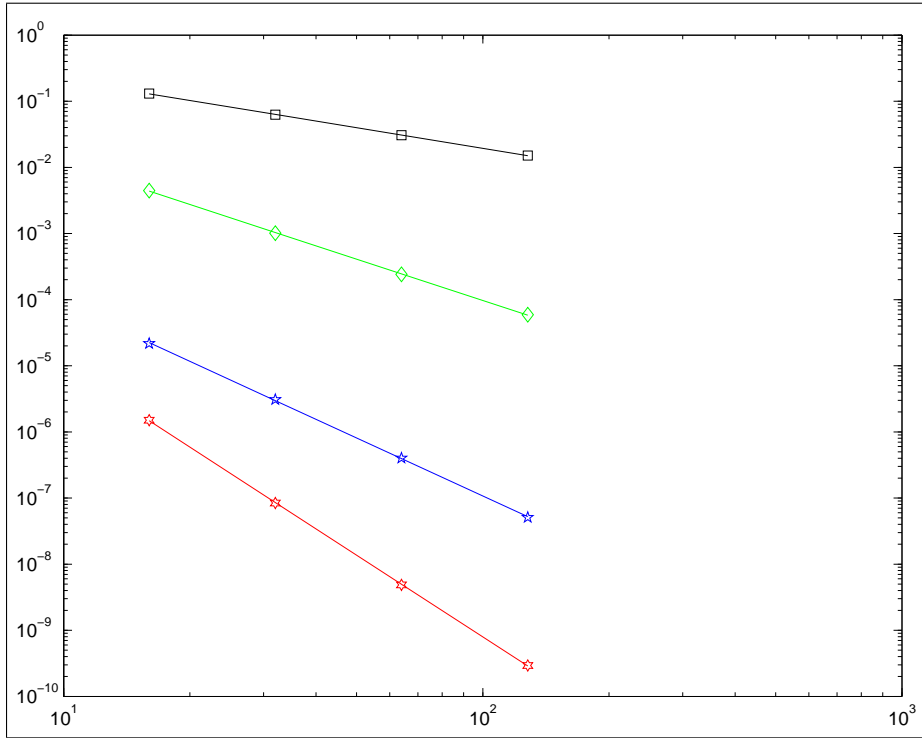


Figure 2: Error analysis in the L^∞ -norm for the one dimensional Laplace equation. The open symbols represent the errors versus the number of grid nodes on a loglog scale, and the solid lines are the least square fits with slopes -1.03 , -2.07 , -2.91 and -4.10 in the case where the ghost cells are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

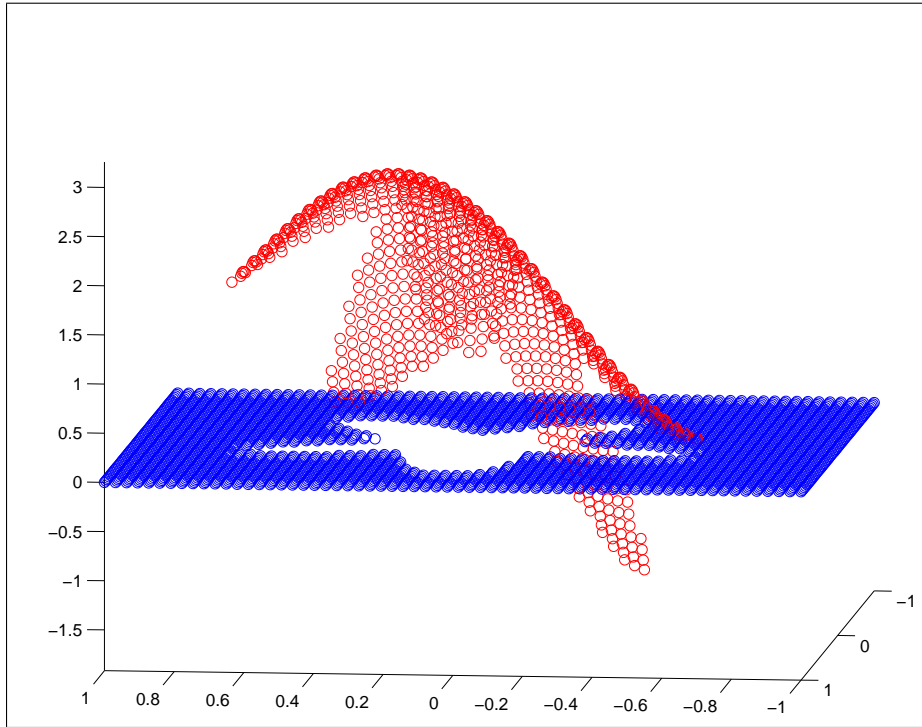


Figure 3: Solution of the Laplace equation on an irregular domain in two spatial dimensions. The exact solution is $T = \sin(\pi x) + \sin(\pi y) + \cos(\pi x) + \cos(\pi y) + x^6 + y^6$ in Ω^- . The grid size is 64×64 , and the ghost cells are defined using cubic extrapolation.

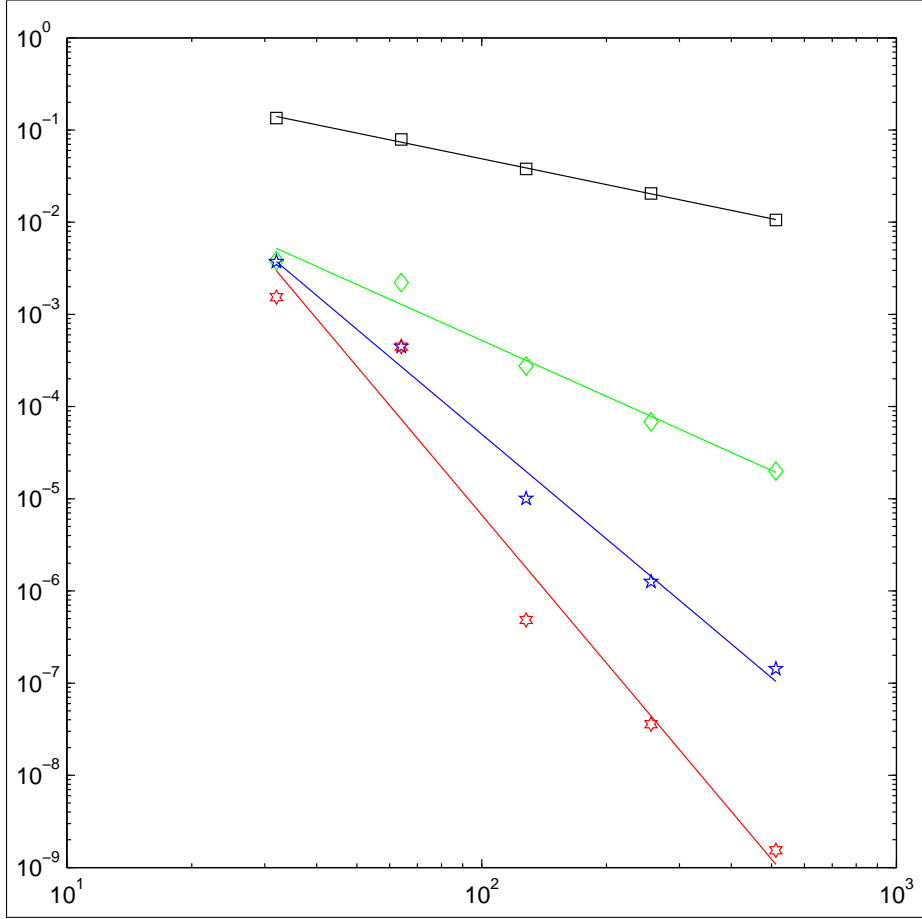


Figure 4: Error analysis in the L^∞ -norm for the two dimensional Laplace equation. The open symbols represent the errors versus the number of grid nodes on a loglog scale, and the solid lines are the least square fits with slopes -0.85 , -1.94 , -2.94 and -3.96 in the case where the ghost cells are defined by constant, linear, quadratic and cubic extrapolation, 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, this is exacerbated by our use of the L^∞ -norm. The L^1 error is more forgiving, since it is based on averaging.

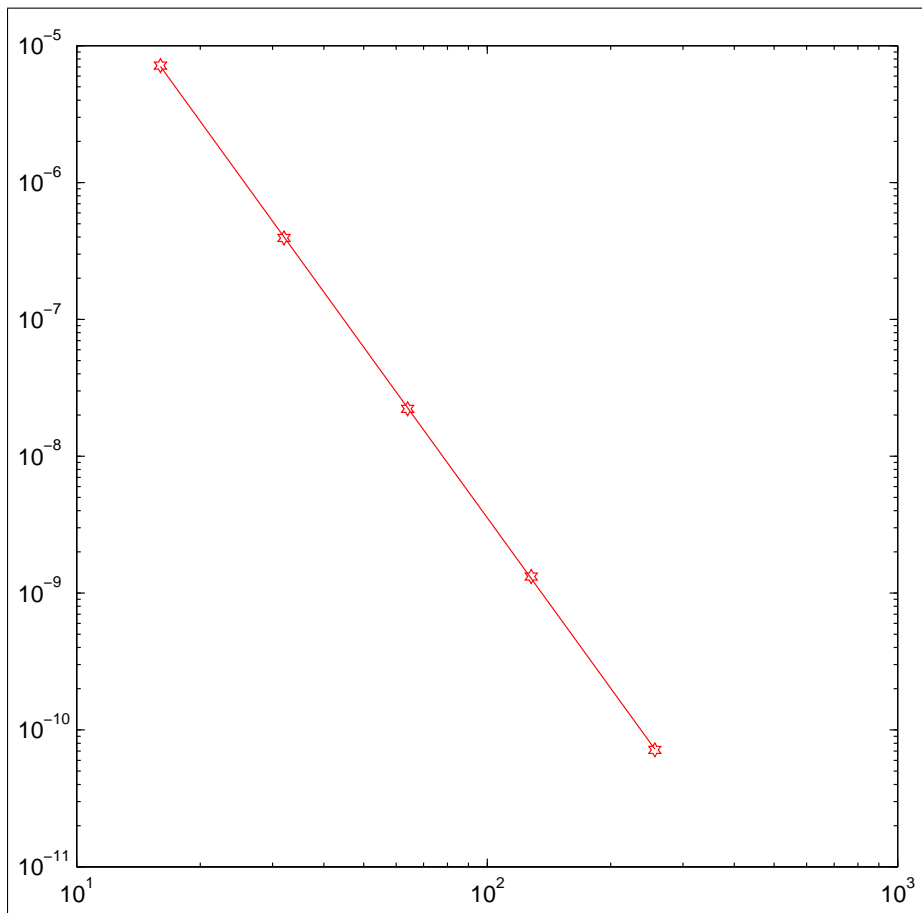


Figure 5: Error analysis in the L^∞ -norm for the one dimensional heat equation. The open symbols represent the error versus the number of grid nodes on a loglog scale, and the solid line depicts the least square fit with slope -4.14 in the case where the ghost cells are defined by cubic extrapolation.

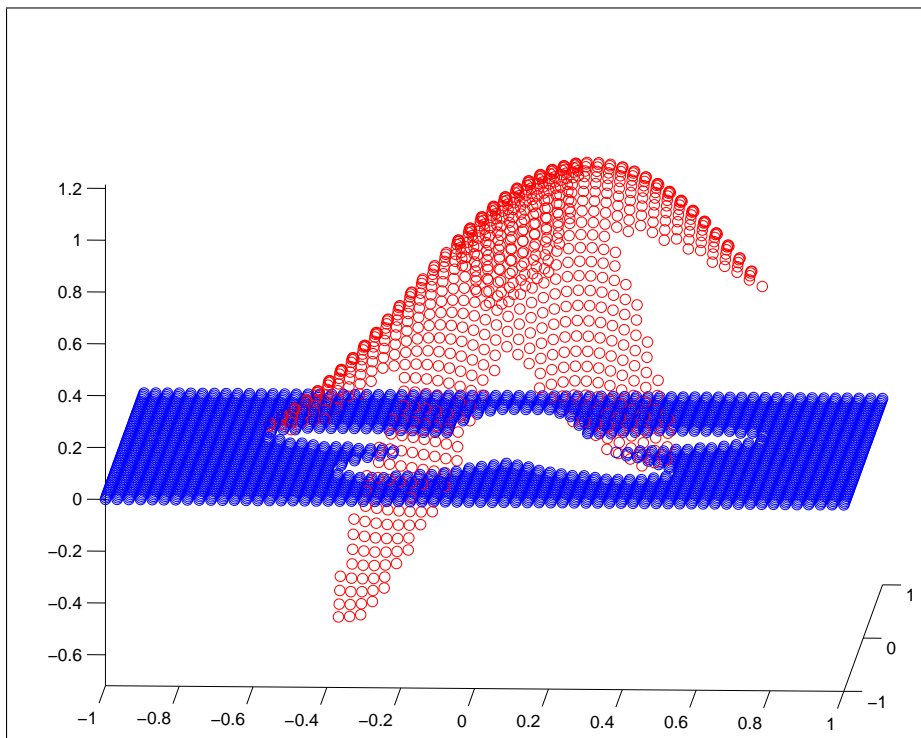


Figure 6: Solution of the heat equation on an irregular domain in two spatial dimensions. The exact solution is $T = e^{-2t} \sin(x) \sin(y)$ in Ω^- . The grid size is 64×64 , and the ghost cells are defined by cubic extrapolation.

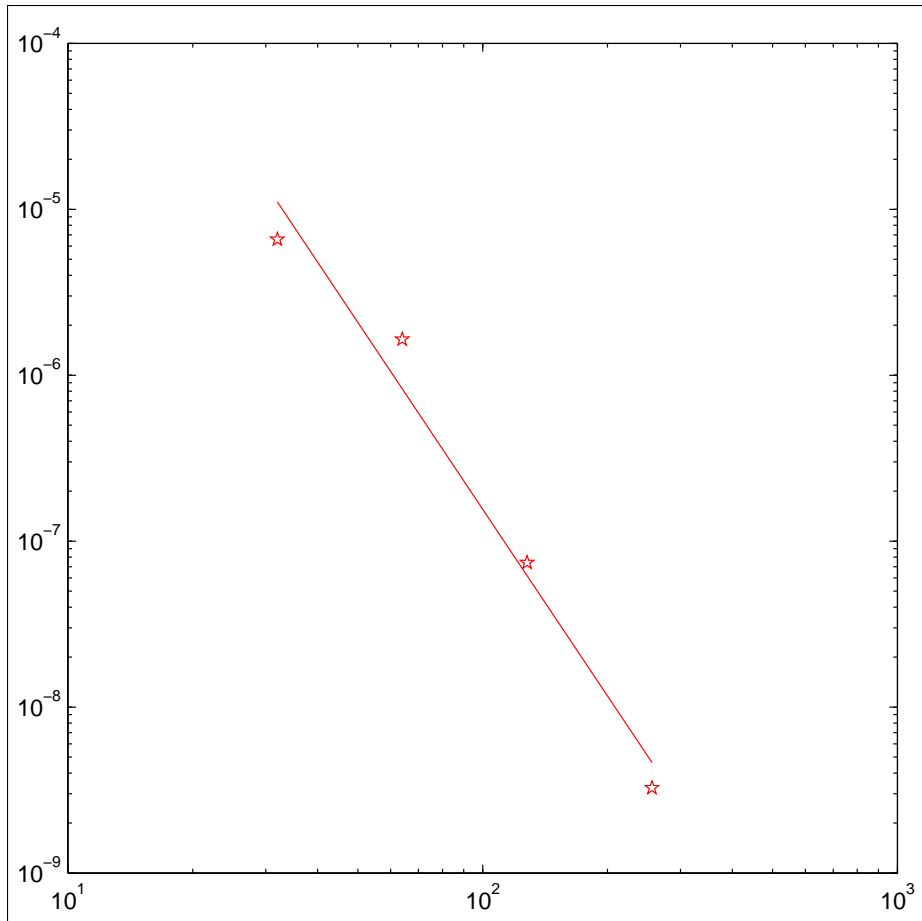


Figure 7: Error analysis in the L^∞ -norm for the two dimensional heat equation. The open symbols represent the error versus the number of grid nodes on a loglog scale, and the solid line depicts the least square fit with slope -3.94 in the case where the ghost cells are defined by cubic extrapolation.

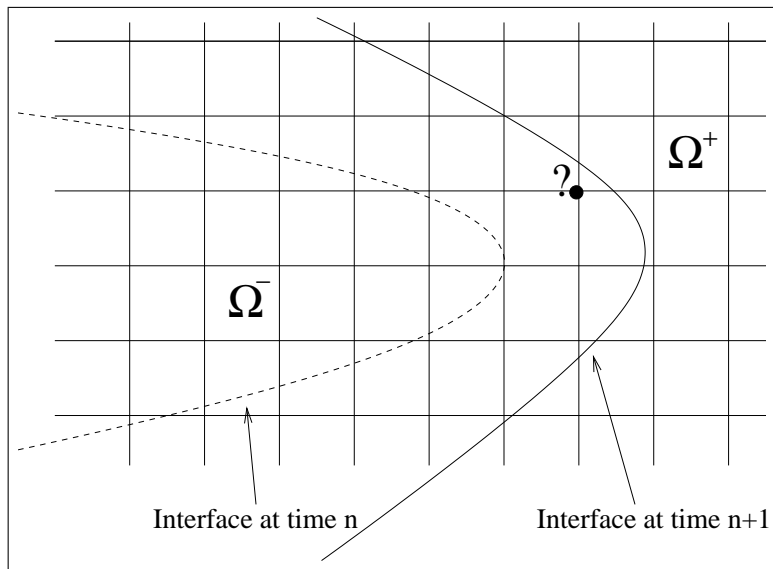


Figure 8: Interface at time t^n (dashed line) and t^{n+1} (solid line). The dark point with a question mark represents a grid node that is swept over by the interface between two consecutive time steps, and where a valid value of T^n needs to be extrapolated in a non-Cartesian normal direction in order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

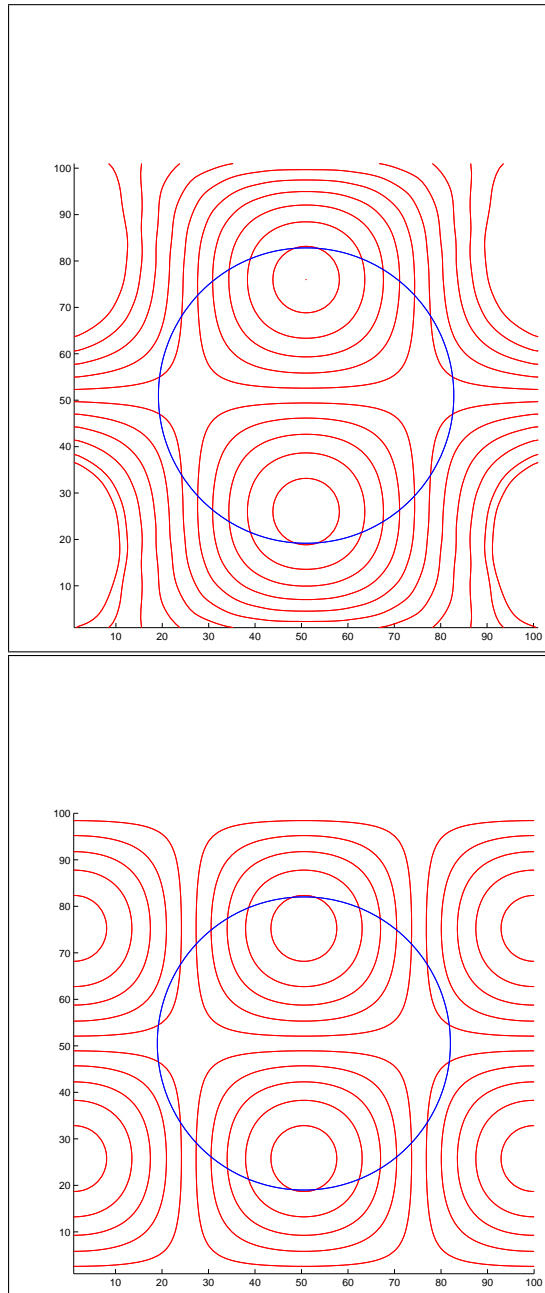


Figure 9: Top: Plots of several isocontours of the solution extrapolated in the normal direction from inside the disk to the outside (cubic extrapolation). Bottom: Exact solution for the sake of comparison.

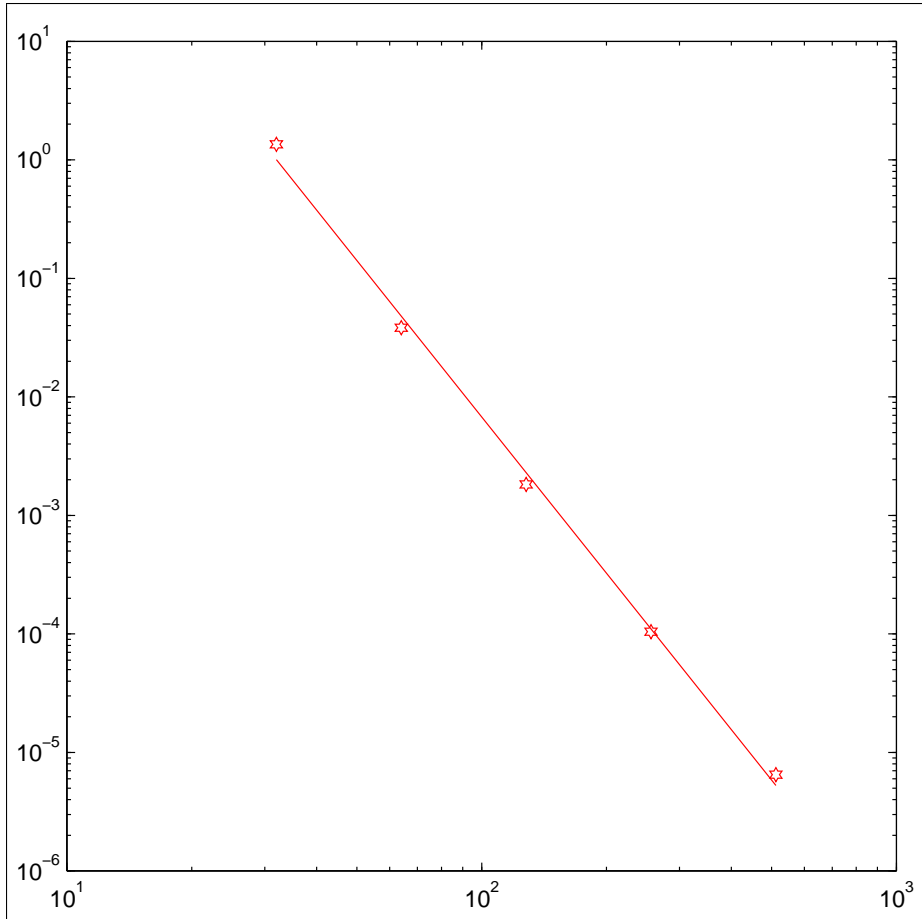


Figure 10: Error analysis in the L^∞ -norm for the two dimensional definition of the ghost cells defined by cubic extrapolation in the normal direction as proposed in [1]. The open symbols are the errors in the L^∞ -norm, and the solid line is a least square fit with slope -4.38 .

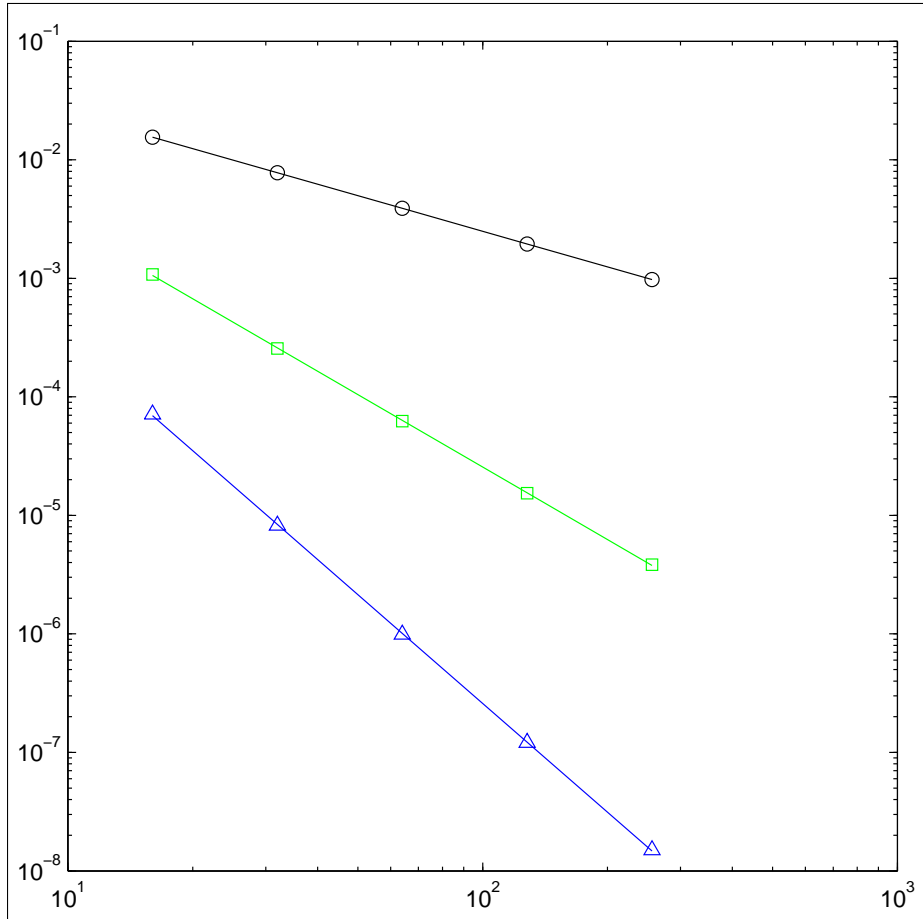


Figure 11: Error analysis on the effect of perturbing the velocity by an $O(\Delta x)$ (circles), $O(\Delta x^2)$ (squares), and $O(\Delta x^3)$ (triangles) amount in the case of the one dimensional Stefan problem. The symbols represent the numerical solution on a loglog scale, and the solid lines are the least square fits with slopes -0.99 , -2.03 and -3.05 .

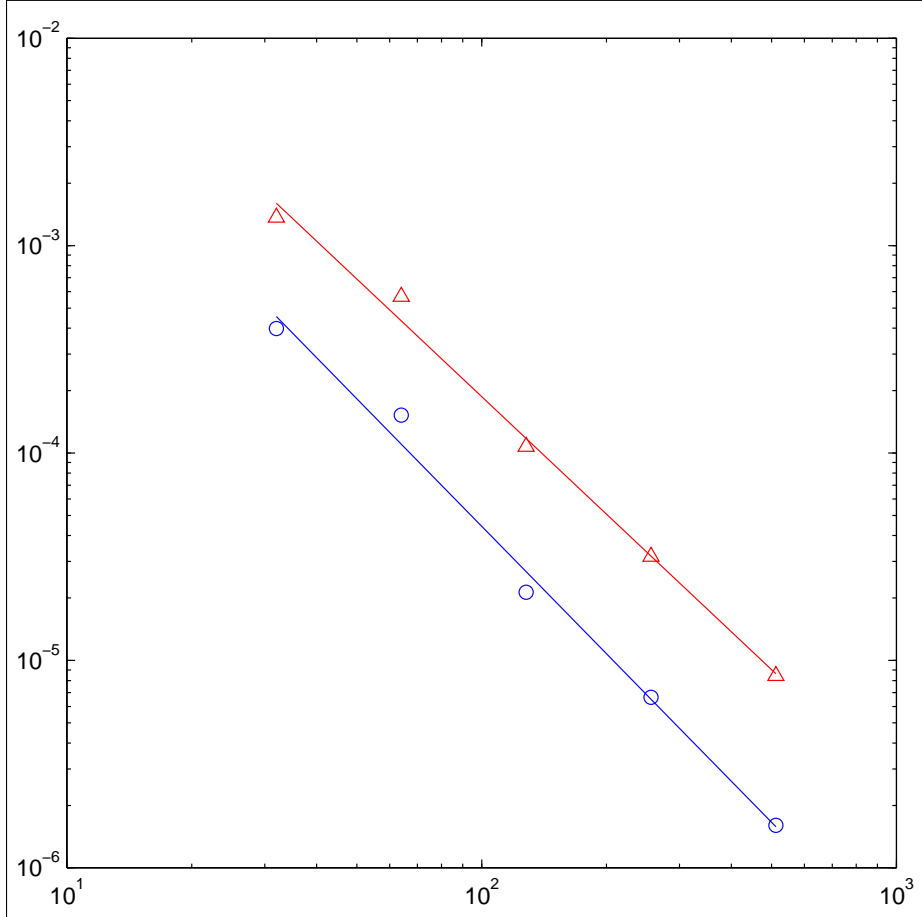


Figure 12: Error analysis in the L^1 (circles) and L^∞ (triangles) norms for the two dimensional reinitialization equation (10). We use a fifth order accurate WENO discretization in space and a third order accurate TVD Runge-Kutta discretization in time. The initial data is $\phi = e^{\rho-2.313} - 1$, with $\rho = \sqrt{x^2 + y^2}$. The symbols represent the errors on a loglog scale, and the solid lines are the least square fits with slopes -2.04, -1.88, respectively.

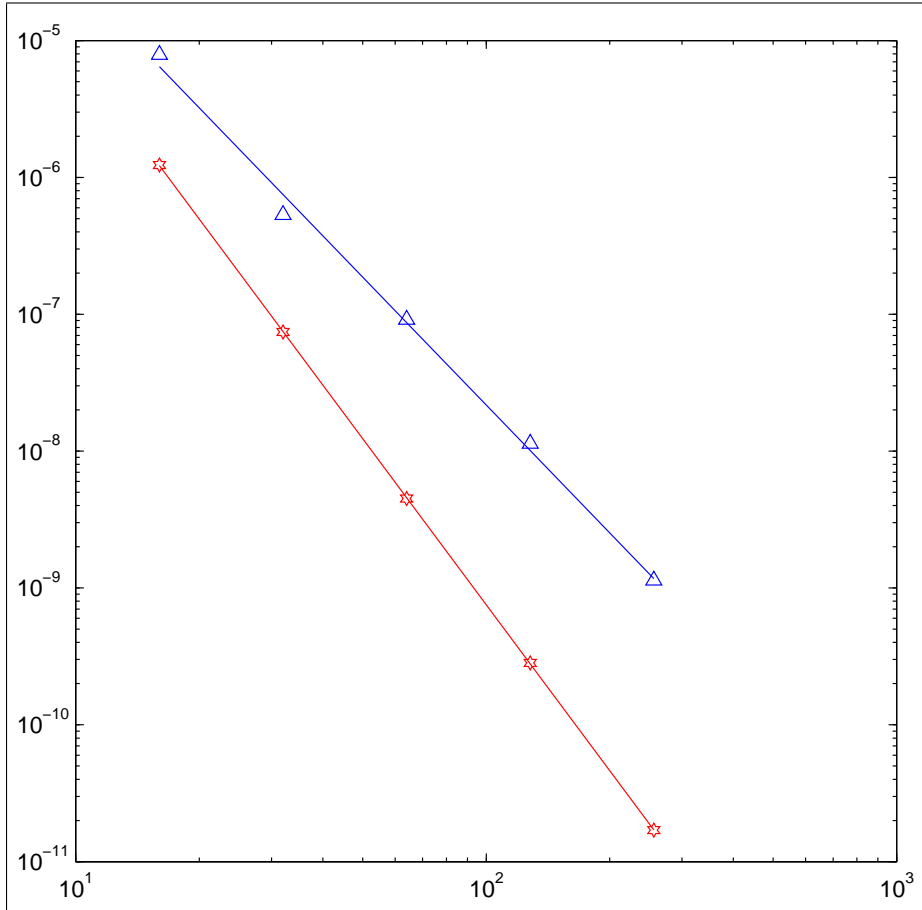


Figure 13: Error analysis in the L^∞ -norm for the one dimensional Stefan problem on a loglog scale. The ghost cells are defined via cubic extrapolation, and we use the Crank-Nicholson scheme with $\Delta t \approx \Delta x^2$. The stars depict the errors when the exact velocity is given, and the triangles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

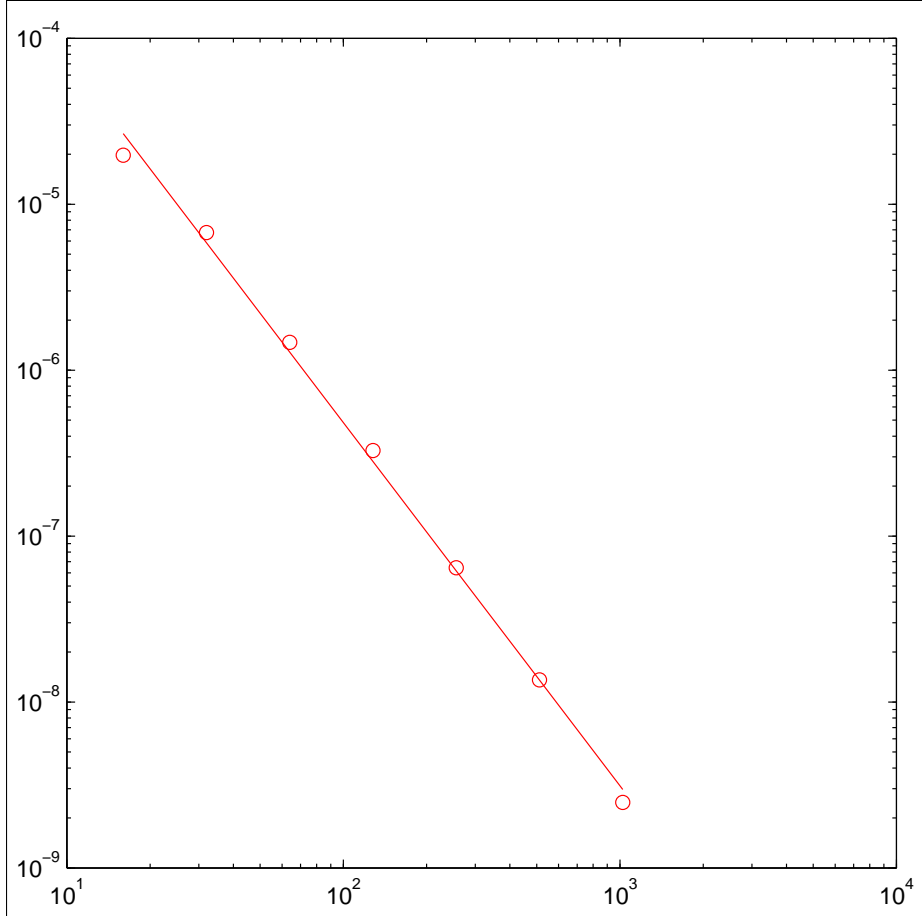


Figure 14: Error analysis in the L^∞ -norm for the one dimensional Frank spheres solution (the interface velocity is not constant in time). The ghost cells are defined by cubic extrapolation, and we use the Crank-Nicholson scheme with $\Delta t \approx \Delta x^{3/2}$. The symbols represent the numerical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

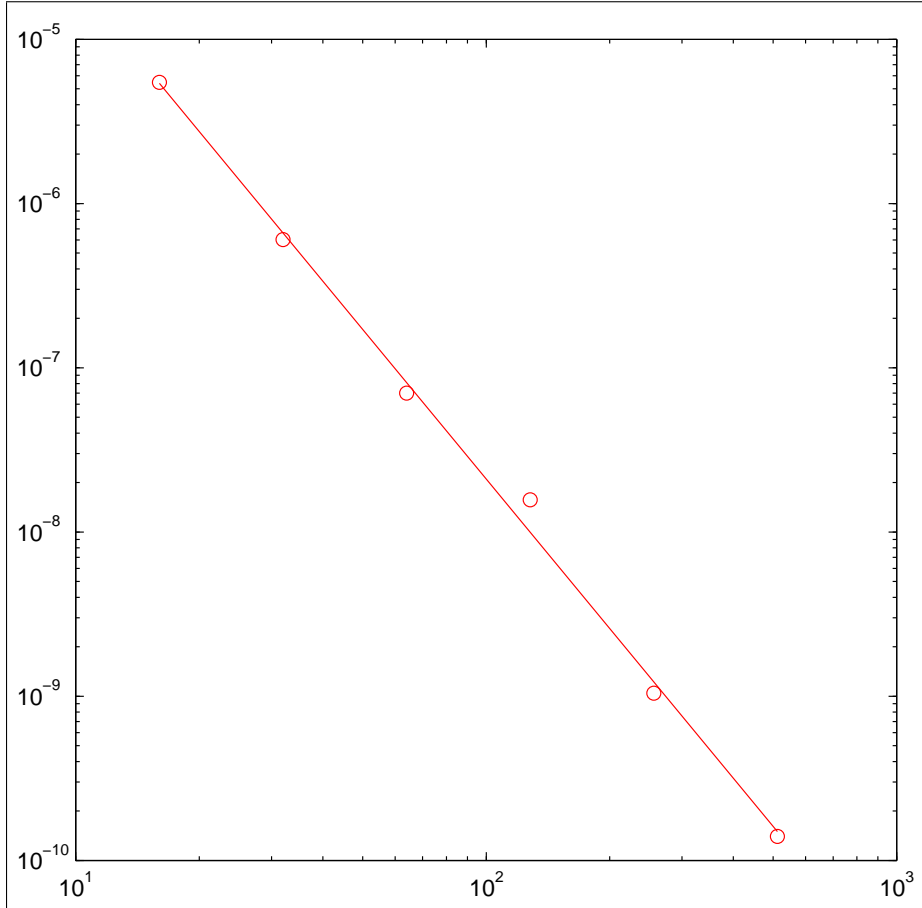


Figure 15: Error analysis in the L^∞ -norm for the one dimensional Frank spheres solution (the interface velocity is not constant in time). The ghost cells are defined via cubic extrapolation, and we use the Crank-Nicholson scheme with $\Delta t \approx \Delta x^{3/2}$. The time discretization involves iterating on the velocity as described in section 4.3. The symbols represent the numerical solution on a loglog scale, and the solid line is the least square fit with slope -3.02.

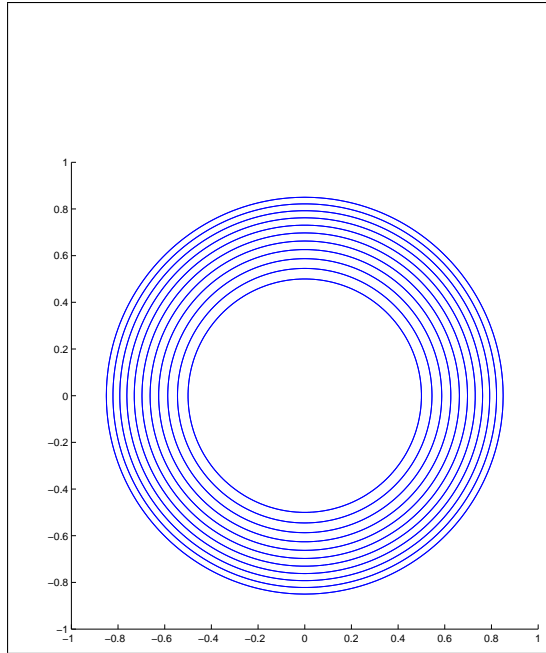


Figure 16: Interface evolution in the case of the two dimensional Frank spheres solution.

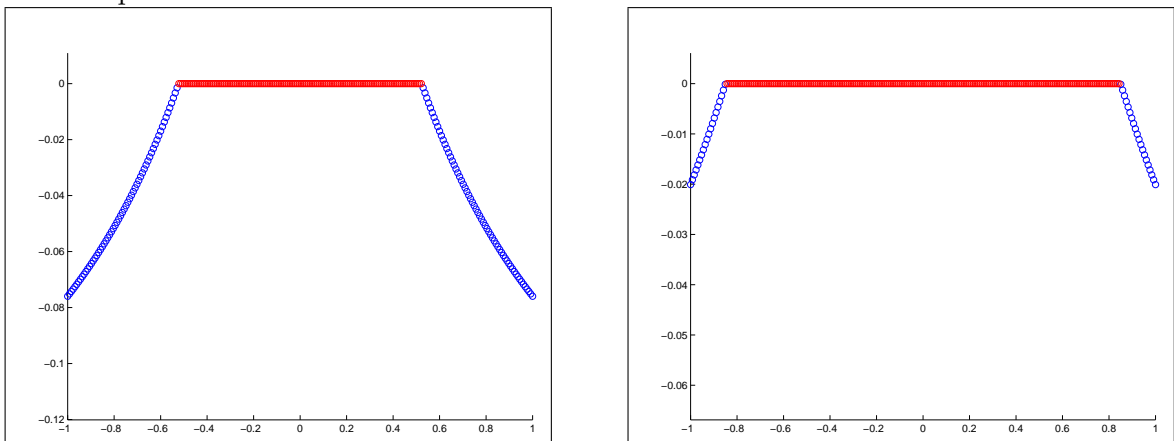


Figure 17: Cross section of the two dimensional Frank spheres solution at $t_{\text{initial}} = 1$ (left) and $t_{\text{final}} = 2.89$ (right).

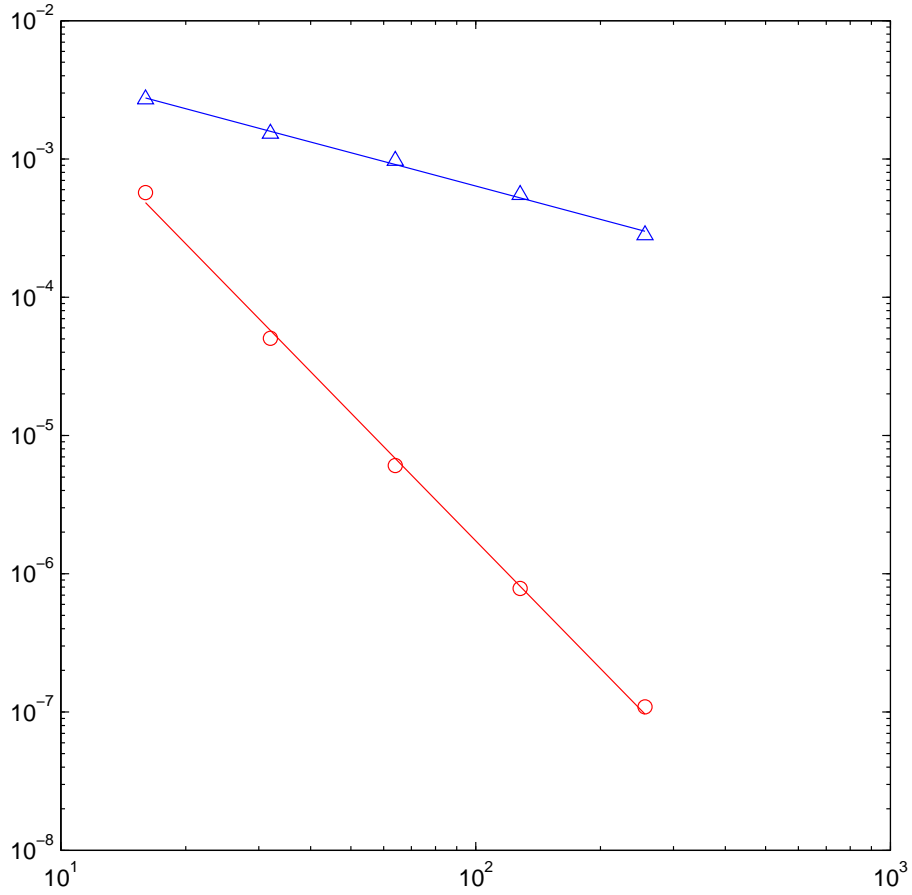


Figure 18: Error analysis in the L^∞ -norm for the two dimensional Frank spheres solution. The triangles illustrate the accuracy obtained with the scheme presented in [9], and the circles represent the accuracy obtained with the algorithm presented in section 4. The open symbols are the errors in the maximum norm, and the solid lines are least square fits with slopes -0.80 and -3.07 , respectively.

References

- [1] Aslam, T., *A Partial Differential Equation Approach to Multidimensional Extrapolation*, J. Comput. Phys. **193**, 349-355 (2003).
- [2] Chen, S., Merriman, B., Osher, S. and Smereka, P., *A Simple Level Set Method for Solving Stefan Problems*, J. Comput. Phys. **135**, 8-29 (1997).
- [3] Chopp, D., *Some Improvement of the Fast Marching Method*, SIAM J. Sci. Comput. **23**, 230-244 (2001).
- [4] Chong, T., *A Variable Mesh Finite difference Method for Solving a Class of Parabolic Differential Equations in one Space Variable*, SIAM J. Num. Anal. **15**, 835-857 (1978).
- [5] Chorin, A., *Curvature and Solidification*, J. Comput. Phys. **58**, 472-490 (1985).
- [6] Fedkiw, R., Aslam, T., Merriman, B., and Osher, S., *A Non-Oscillatory Eulerian Approach to Interfaces in Multimaterial Flows (The Ghost Fluid Method)*, J. Comput. Phys. **152**, 457-492 (1999).
- [7] George, W. and Warren, J., *A Parallel 3D Dendritic Growth Simulator Using the Phase-Field Method*, J. Comput. Phys. **177**, 264-283 (2002).
- [8] Gibou, F., Fedkiw, R., Caflich, R. and Osher, S., *A Level Set Approach for the Simulation of Dendritic Growth*, J. Sci. Comput. **19**, 183-199 (2003).
- [9] Gibou, F., Fedkiw, R., Cheng, L-T. and Kang, M., *A Second Order Accurate Symmetric Discretization of the Poisson Equation on Irregular Domains*, J. Comput. Phys. **176**, 1-23 (2002).
- [10] Kim, Y.-T., Goldenfeld, N. and Dantzig, J., *Computation of Dendritic Microstructures using a Level Set Method*, Phys. Rev. E **62**, 2471-2474 (2000).
- [11] Hoffman, J., *Relationship Between the Truncation Errors of Centered Finite-Difference Approximations on Uniform and Nonuniform Meshes*, J. Comput. Phys. **46**, 469-474 (1982).
- [12] Jiang, G.-S. and Peng, D., *Weighted ENO Schemes for Hamilton Jacobi Equations*, SIAM J. Sci. Comput. **21**, 2126-2143 (2000).

- [13] Jiang, G.-S. and Shu, C.-W., *Efficient Implementation of Weighted ENO Schemes*, J. Comput. Phys. **126**, 202-228 (1996).
- [14] Johansen, H., *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] Johansen, H. and Colella, P., *A Cartesian Grid Embedded Boundary Method for Poisson's Equation on Irregular Domains*, J. Comput. Phys. **147**, 60-85 (1998).
- [16] Juric, D. and Tryggvason, G., *A Front Tracking Method for Dendritic Solidification*, J. Comput. Phys. **123**, 127-148 (1996).
- [17] Karma, A. and Rappel, W.-J., *Quantitative Phase-Field Modeling of Dendritic Growth in Two and Three Dimensions*, Phys. Rev. E **57**, 4323-4349 (1997).
- [18] Kim, Y.-T., Goldenfeld, N. and Dantzig, J., *Computation of Dendritic Microstructures using a Level Set Method*, Phys. Rev. E **62**, 2471-2474 (2000).
- [19] Kreiss, H.-O., Manteuffel, T., Swartz, B., Wendroff, B. and White, A., *Supra-Convergent Schemes on Irregular Grids* Math. Comp. **47**, 176, 537-554 (1986).
- [20] Lambert, J., *Numerical Methods for Ordinary Differential Systems*, Wiley & Sons, New York, 1993.
- [21] LeVeque, R. and Li, Z., *The Immersed Interface Method for Elliptic Equations with Discontinuous Coefficients and Singular Sources*, SIAM J. Num. Anal. **31**, 1019-1044 (1994).
- [22] Liu, X.-D., Osher, S., and Chan, T., *Weighted Essentially Non-Oscillatory Schemes*, J. Comput. Phys. **126**, 200-212 (1996).
- [23] Manteuffel, T. and White, A., *The Numerical Solution of Second-Order Boundary Value Problems on Nonuniform Meshes*, Math. Comp. **47**, 176, 511-535 (1986).
- [24] Osher, S. and Fedkiw, R., *Level Set Methods and Dynamic Implicit Surfaces*, Springer-Verlag, New York (2002).
- [25] Peskin, C., *Flow Patterns Around Heart Valves: A Numerical Method*, J. Comput. Phys. **10**, 252-271 (1972).

- [26] Plapp, M. and Karma, A., *Multiscale Finite-Difference-Diffusion-Monte-Carlo Method for Simulating Dendritic Solidification*, J. Comput. Phys. **165**, 592-619 (2000).
- [27] Schmidt, A., *Computation of Three Dimensional Dendrites with Finite Elements*, J. Comput. Phys. **125**, 293-312 (1996).
- [28] Saad, Y., *Iterative Methods for Sparse Linear Systems*, PWS publishing company, Boston (1996).
- [29] Sethian, J., *A Fast Marching Level Set Method for Monotonically Advancing Fronts*, Proc. Nat. Acad. Sci. **93**, 1591-1595 (1996).
- [30] Sethian, J., *Level Set Methods and Fast Marching Methods*, Cambridge University Press, Cambridge, (1999).
- [31] Sethian, J. and Strain, J., *Crystal Growth and Dendritic Solidification*, J. Comput. Phys. **98**, 231-253 (1992).
- [32] Sussman, M. and Hussaini, M.Y., *A Discontinuous Spectral Element Method for the Level Set Equation*, J. Sci. Comput. **19**, 479-500 (2003).
- [33] Sussman, M., Smereka, P. and Osher, S., *A Level Set Approach for Computing Solutions to Incompressible Two-Phase Flow*, J. Comput. Phys. **114**, 146-159 (1994).
- [34] Tsitsiklis, J., *Efficient algorithms for Globally Optimal Trajectories*, IEEE Trans. on Automatic Control **40**, 1528-1538 (1995).
- [35] Udaykumar, H., Mittal, R. and Shyy, W., *Computation of Solid-Liquid Phase Fronts in the Sharp Interface Limit on Fixed Grids*, J. Comput. Phys. **153**, 535-574 (1999).
- [36] Zhao, P. and Heinrich, J., *Front Tracking Finite Element Method for Dendritic Solidification*, J. Comput. Phys. **173**, 765-796 (2001).