UCLA

COMPUTATIONAL AND APPLIED MATHEMATICS

Immersed Interface Methods for Moving Interface Problems

Zhilin Li

May 1995

CAM Report 95-25

Department of Mathematics
University of California, Los Angeles
Los Angeles, CA. 90024-1555

IMMERSED INTERFACE METHODS FOR MOVING INTERFACE PROBLEMS *

ZHILIN LI †

Abstract. A second order difference method is developed for the nonlinear moving interface problem of the form

$$u_t + \lambda u u_x = (\beta u_x)_x - f(x,t), \quad x \in [0,\, \alpha\,) \bigcup (\, \alpha,\, 1],$$
$$\frac{d\alpha}{dt} = w\,(t, \alpha;\, u, u_x)\,,$$

where $\alpha(t)$ is the moving interface. The coefficients $\beta(x,t)$ and the source term $f(x,t)$ can be discontinuous across $\alpha(t)$ and moreover, $f(x,t)$ may have a delta function singularity there. As a result, although the equation is parabolic, the solution u and its derivatives may be discontinuous across $\alpha(t)$. Two typical interface conditions are considered. One condition occurs in Stefan-like problems in which the solution is known on the interface. A new stable interpolation strategy is proposed. The other type occurs in a one-dimensional model of Peskin's immersed boundary method in which only jump conditions are given across the interface. The Crank-Nicolson difference scheme with modifications near the interface is used to solve for the solution $u(x,t)$ and the interface $\alpha(t)$ simultaneously. Several numerical examples, including models of ice-melting and glaciation, are presented. Second order accuracy on uniform grids is confirmed both for the solution and the position of the interface.

Key words. immersed interface method, Stefan problem, moving interface, discontinuous coefficients, singular source term, immersed boundary method, Cartesian grid, heat conduction

AMS subject classifications. 65N06, 76T05, 80A22, 86A40

1. Introduction.

In this paper we study the immersed interface method for the one-dimensional moving interface problem

$$u_t + \lambda u u_x = (\beta u_x)_x - f(x,t), \quad x \in [\,0,\, \alpha\,) \bigcup (\, \alpha,\, 1\,], \tag{1}$$

$$\frac{d\alpha}{dt} = w\left(t, \alpha;\, u^-, u^+, u_x^-, u_x^+\right), \quad t > 0, \tag{2}$$

where w is a known function and u^-, u^+, u_x^-, and u_x^+ are the limiting values of $u(x,t)$ and $u_x(x,t)$ from the left and right side of a moving interface $\alpha(t)$. The coefficient $\beta(x,t) > 0$ and the source term $f(x,t)$ may be discontinuous; furthermore, $f(x,t)$ may have a delta function singularity at the interface $\alpha(t)$. This is a parabolic problem and the solution is piecewise smooth. The discontinuities can only occur on the interface $\alpha(t)$. There are also initial and boundary conditions, which are not our main interests here, may also be imposed. We assume these can be handled using conventional techniques.

The interface $\alpha(t)$ divides the solution domain into two parts: $0 \le x < \alpha(t)$ and $\alpha(t) < x \le 1$. The solution in each domain $[0, \alpha(t))$ and $(\alpha(t), 1]$ is smooth, but coupled with the solution on the other side by interface conditions (or internal boundary conditions) which usually take one of the following forms:

* This work was supported by URI grant #N00014092-J-1890 from ARPA, NSF Grants DMS-9204329, DMS-9303404, and DOE Grant DE-FG06-93ER25181.

† Department of Mathematics, University of California at Los Angeles, Los Angeles, CA 90095. (zhilin@math.ucla.edu).

Case 1: The solution on the interface is given. One example is a mathematical model for solidification problems. When tracking an interface of melting ice, for example, the temperature at the melting/freezing interface is given by the melting temperature of the ice. In this case $u(\alpha, t) = u_0$, the melting temperature. When $\lambda \equiv 0$, this is the classical Stefan problem.

Various approaches have been used to solve Stefan or other *linear* free or moving interface/boundary problems numerically [1], [3], [5], [6], [7], [8], [10], [18], [20], [25], and recent work by Chen and Osher [4] using the level set approach. Compared to Case 2 discussed below, the Stefan problem is relatively easier to solve because the value of the solution on the interface is known. However, few numerical methods are second order accurate in the infinity norm for both the solution and the interface. Most methods involve some transformations either for the differential equations or the coordinate system, which complicates the problem in some way. Some of the methods would exhibit some kind of instability near the interface as explained in Section 4. The method proposed in Section 2–6 is simpler, more stable, and is second order accurate both for the solution u and the interface $\alpha(t)$ simultaneously for more general equations. Furthermore, the solution is obtained using a fixed Cartesian grid.

Case 2: Jump conditions of the form

$$[u] \stackrel{\text{def}}{=} u(\alpha^+, t) - u(\alpha^-, t) = q(t), \tag{3}$$

$$[\beta u_x] \stackrel{\text{def}}{=} \beta(\alpha^+, t)\, u_x(\alpha^+, t) - \beta(\alpha^-, t)\, u_x(\alpha^-, t) = v(t) \tag{4}$$

are given. Our interest in this case was motivated by the desire to develop a second order accurate algorithm for solving the incompressible Navier-Stokes equations obtained from the formulation of the *immersed boundary method.* This method was originally developed by Peskin and his co-workers for studying blood flow in a beating heart [21], [22], [23], [24], but has been used in a wide variety of other problems. In Peskin's method the physical domain is immersed in a rectangular region. The boundary condition is treated as a forcing term which is only supported on the boundary, therefore the forcing term is singular. Case 2 is a one-dimensional model for the immersed boundary method formulation with a more general equation for the motion. A linear model in one dimension would be

$$u_t = (\beta u_x)_x - C(t)\, \delta(x - \alpha(t)) - \hat{C}(t)\, \delta'(x - \alpha(t)) - f(x, t),$$

From this equation, assuming $f(x,t)$ is continuous, we can derive the following jump conditions

$$[u] = \frac{2\, \hat{C(t)}}{\beta^- + \beta^+}, \qquad [\beta u_x] = C(t),$$

where β^- and β^+ are the limiting value of $\beta(x,t)$ from left and right of the interface $\alpha(t)$. In many other cases the jump condition can be obtained from physical reasoning. For instance, if u stands for temperature, then $[u] = 0$, and $[\beta u_x] = v(t)$ meaning that the net heat flux across the interface is equal to the source strength $v(t)$. One of advantages of using the jump conditions is that we do not need to discretize the delta function, an approach described briefly in the next paragraph.

Peskin and many other people use a discrete delta function approach to spread the singular force term to the nearby grid points and use the discrete delta function again to interpolate the velocity field to move the boundary. While this approach seems to work, it is usually only *first* order accurate and difficult to analyze especially in two or higher dimensions.

Various attempts have been made to analyze and improve the discrete delta function approach or to try to solve the resulting differential equations differently.

Beyer and LeVeque [2] studied various one-dimensional moving interface problems for the heat equation assuming a *priori* knowledge of the interface. A discrete delta function is carefully selected and some correction terms are added if necessary in their approach to get second order accuracy. Wiegmann and Bube [26] recently applied the immersed interface method for certain one-dimensional nonlinear problem with a *fixed* interface. However, for the interface problems discussed here the interface is *unknown* and *moving*, and the discrete difference scheme is a nonlinear system of equations involving the solution and the interface.

In two dimensions, the discrete delta function approach generally is only first order accurate. And it seems impossible to find a generic discrete delta function which makes the algorithm second order accurate. A new approach, *the immersed interface method* [12], [15], is intended to solve general PDEs with discontinuous coefficients and/or singular sources with second order accuracy at all grid points including those which are close to or on the interface. The main idea is to incorporate the known jumps in the solution or its derivatives into the finite difference scheme, obtaining a modified scheme whose solution is second order accurate at all grid points on the uniform grid even for a quite arbitrary interface. This method has been implemented for several different applications in one, two and three dimensions [13], [14] [16], [17]. By implementing this method for one dimensional model here, we not only effectively solve some general one-dimensional moving interface problems, but also hope to get some insight into the method as well as the problem. We are currently working on second order accurate immersed interface methods for the full Navier-Stokes equation with a moving boundary, where we have to deal with the nonlinear term $u \cdot \nabla$. That is one of the reasons why we have the nonlinear term $\lambda u u_x$ in our model equation (1).

Case 2 (with $\lambda = 0$) is also a model of heat conduction with an interface between two different materials. Now u is the temperature and hence is continuous meaning $q(t) \equiv 0$ in (3). The net heat flux across the interface is $v(t)$ in (4), the source strength on the interface. Again in this case we do not know the value of the solution on the interface but only the jump conditions.

For many classical Stefan problem, the motion of the interface is proportional to the flux across the interface

$$\frac{d\alpha}{dt} = \sigma(t)\,[\beta u_x], \qquad u(\alpha, t) = u_0 \tag{5}$$

where u_0 is the known temperature at the interface. This kind of problems fit both Case 1 and Case 2 and will be discussed in §4.3.

2. Computational frame. We use a uniform grid

$$x_i = ih, \qquad i = 0, 1, \cdots, N, \quad x_0 = 0, \quad x_N = 1,$$

where h is the step size in space. We use k as the step size in time and assume the ratio k/h is a constant so that we can write $O(k)$ as $O(h)$ or vice versa. Using the

Crank–Nicolson scheme, the semi-discrete difference scheme for (1) can be written in the following general form,

$$\tag{6} \boxed{\begin{aligned} \frac{u_i^{n+1}-u_i^n}{k} - Q_i^{n+\frac{1}{2}} + \frac{\lambda}{2}\left(u_i^n u_{x,i}^n + u_i^{n+1}u_{x,i}^{n+1}\right) &= \\ \frac{1}{2}\left((\beta u_x)_{x,i}^n + (\beta u_x)_{x,i}^{n+1}\right) &- \frac{1}{2}\left(f_i^n + f_i^{n+1}\right), \end{aligned}}$$

where $u_{x,i}^n$ and $(\beta u_x)_{x,i}^n$ are discrete analogues of u_x and $(\beta u_x)_x$ at (x_i, t^n), and $Q_i^{n+\frac{1}{2}}$ is a correction term needed in the case when α crosses a grid point during the time step, as discussed in the next section. Numerical schemes will be displayed in a box, as illustrated by equation (6). For simplicity, we will drop the superscript n when there is no confusion. At a grid point x_i, which is away from the interface, (*i.e.* $\alpha \notin [x_{i-1}, x_{i+1}]$), the classic central differences will be used

$$\tag{7} u_{x,i} = \frac{u_{i+1}-u_{i-1}}{2h},$$

$$\tag{8} (\beta u_x)_{x,i} = \frac{\beta_{i-\frac{1}{2}}u_{i-1} - (\beta_{i-\frac{1}{2}}+\beta_{i+\frac{1}{2}})u_i + \beta_{i+\frac{1}{2}}u_{i+1}}{h^2},$$

where $\beta_{i+\frac{1}{2}} = \beta(x_i + h/2, t)$. We will discuss how to discretize u_x and $(\beta u_x)_x$ when $\alpha \in [x_{j-1}, x_{j+1}]$ for Case 1 and Case 2 in Section 4.

The interface location is also determined by the trapezoidal method applied to (2)

$$\tag{9} \boxed{\frac{\alpha^{n+1}-\alpha^n}{k} = \frac{1}{2}\left(w^n + w^{n+1}\right),}$$

where $w^l = w\left(t^l, \alpha^l; u^{-,l}, u^{+,l}, u_x^{-,l}, u_x^{+,l}\right)$, and α^l, $u^{\pm,l}$, $u_x^{\pm,l}$ are the approximation of $\alpha(t^l)$, $u(\alpha^\pm, t^l)$, and $u_x(\alpha^\pm, t^l)$ respectively. These quantities are only defined on the interface. We use the following notation to express the jump in a function $g(x,t)$ across the interface

$$[g] \overset{\text{def}}{=} g(\alpha^+, t) - g(\alpha^-, t); \qquad [g]_{;t} \overset{\text{def}}{=} g(\alpha, t^+) - g(\alpha, t^-).$$

It is easy to see that $[g] = \pm[g]_{;t}$ and the sign depends on the motion of the interface. For example we have minus sign for the case in Fig.1 (*a*), and plus sign for the one in Fig.1 (*b*).

The kernel of the algorithm at a time level t^n consists of the following:

- Determine $Q_j^{n+\frac{1}{2}}$ if the interface crosses the grid line $x = x_j$ from time t^n to t^{n+1}.
- Derive the difference formula for u_x and $(\beta u_x)_x$ at the two grid points closest to the interface.
- Compute the quantities $u^\pm$, $u_x^\pm$, $[u_t]$ *etc.* on the interface.
- Solve the nonlinear system of equations for $\{u_i^{n+1}\}$ and the location of the inteface α^{n+1}.

Each of these steps will be described in detail in the remainder of the paper. Away from the interface, the local truncation errors for the difference scheme are $O(h^2)$. But at few grid points near the interface, we allow the local truncation errors to be $O(h)$ based on the fact that the local truncation error of a difference scheme on a boundary can be one order lower than those of interior points without affecting global second order accuracy.

3. Grid crossing. If there is no grid crossing at a grid point x_i from time t^n to time t^{n+1}, meaning that (x_i, t^n) and (x_i, t^{n+1}) are on the same side of the interface $\alpha(t)$, i.e., $x_i \notin (\alpha^n, \alpha^{n+1})$, then we can take $Q_i^{n+\frac{1}{2}} = 0$ and we have

$$\frac{u(x_i, t^{n+1}) - u(x_i, t^n)}{k} = \frac{1}{2}\left[u_t(x_i, t^{n+1}) + u_t(x_i, t^n)\right] + O(k^2). \tag{10}$$

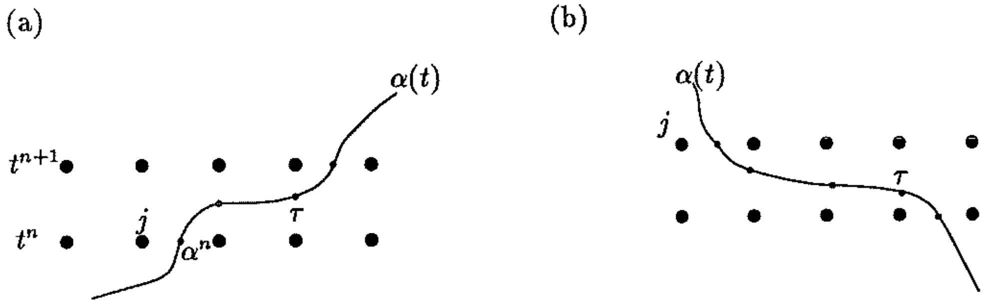

FIG. 1. *Interface crossing the grid. (a) $\alpha(t)$ increases with time. (b) $\alpha(t)$ decreases with time.*

However, if the interface crosses the grid line $x = x_j$ at some time[1] τ, $t^n < \tau < t^{n+1}$, such that $x_j = \alpha(\tau)$, see Fig.1, then the time-derivative of u is not smooth. In this case even though we can approximate the x-derivatives well at each time level (see Section 4), the standard Crank-Nicolson scheme needs to be corrected to guarantee second order accuracy. This is done by choosing a correction term $Q_j^{n+\frac{1}{2}}$ based on the following theorem:

THEOREM 3.1. *Suppose the equation $\alpha(t) = x_j$ has a unique solution τ in the interval $t^n < t < t^{n+1}$. If we choose*

$$\boxed{Q_j^{n+\frac{1}{2}} = \frac{[u]_{;\tau}}{k} + \frac{1}{k}\left(t^{n+\frac{1}{2}} - \tau\right)[u_t]_{;\tau}} \tag{11}$$

then

$$\frac{u(x_j, t^{n+1}) - u(x_j, t^n)}{k} - Q_j^{n+\frac{1}{2}} = \frac{1}{2}\left(u_t(x_j, t^{n+1}) + u_t(x_j, t^n)\right) + O(k). \tag{12}$$

Proof: We expand $u(x_j, t^n)$ and $u(x_j, t^{n+1})$ in Taylor series about time τ from each side of the interface to get

$$\begin{aligned} u(x_j, t^n) &= u(x_j, \tau^-) + (t^n - \tau)\, u_t(x_j, \tau^-) + O(k^2), \\ u(x_j, t^{n+1}) &= u(x_j, \tau^+) + (t^{n+1} - \tau)\, u_t(x_j, \tau^+) + O(k^2) \\ &= u(x_j, \tau^-) + [u]_{;\tau} + (t^{n+1} - \tau)\, u_t(x_j, \tau^-) + (t^{n+1} - \tau)\,[u_t]_{;\tau} + O(k^2). \end{aligned}$$

Combining the two expressions above gives

$$u(x_j, t^{n+1}) - u(x_j, t^n) = k\, u_t(x_j, \tau^-) + [u_t]_{;\tau} + (t^{n+1} - \tau)\,[u_t]_{;\tau} + O(k^2). \tag{13}$$

[1] The cross time really depends on the grid index j as well as the time index n, see Fig.1, we use the notation τ in this paper to simplify the notation.

On the other hand, from [2] (p. 363) we also have

$$ku_t(x_j,\tau^-) = \frac{k}{2}\left(u_t(x_j,t^n) + u_t(x_j,t^{n+1})\right) - \frac{k}{2}[u_t]_{;\tau} + O(k^2). \tag{14}$$

Substituting (14) into (13) gives

$$\begin{aligned}\frac{u(x_j,t^{n+1}) - u(x_j,t^n)}{k} &= \frac{1}{2}\left(u_t(x_j,t^n) + u_t(x_j,t^{n+1})\right) + \frac{[u]_{;\tau}}{k} \\ &\quad + \frac{t^{n+\frac{1}{2}} - \tau}{k}[u_t]_{;\tau} + O(k).\end{aligned} \tag{15}$$

This is equivalent to (12). □

We know $[u]_{;\tau}$ from jump conditions. However, to compute $Q_j^{n+\frac{1}{2}}$, we also need to find the location τ and the jump $[u_t]_{;\tau}$. Let us first discuss how to find τ if it exists. Using the Crank–Nicolson formula twice we get:

$$\begin{aligned}\frac{\alpha^\tau - \alpha^n}{\tau - t^n} &= \frac{1}{2}\left(w^n + w^\tau\right), \\ \frac{\alpha^{n+1} - \alpha^\tau}{t^{n+1} - \tau} &= \frac{1}{2}\left(w^\tau + w^{n+1}\right).\end{aligned}$$

Eliminating w^τ term we get

$$\boxed{\frac{x_j - \alpha^n}{\tau - t^n} - \frac{\alpha^{n+1} - x_j}{t^{n+1} - \tau} = \frac{1}{2}\left(w^n - w^{n+1}\right).} \tag{16}$$

From this quadratic equation, we can solve for the crossing time τ.

The estimation of $[u_t]_{;\tau}$ depends on interface conditions and will be discussed in Section 5.

4. Discretization of u_x and $(\beta u_x)_x$ near the interface. As we mentioned earlier, at grid points which are away from the interface ($|x_i - \alpha| > h$) we use the central difference scheme. So only the closest grid points from the left and the right of the interface, need special treatment at each time level t^n or t^{n+1}. The discretization apparently depends on interface conditions and will be discussed separately in this section.

4.1. Case 1: the solution on the interface is known. Let the solution on the interface be

$$u(\alpha, t) = r(t). \tag{17}$$

We begin our discussion with very general motion equation (2). For those Stefan problems in which the velocity is propotional to the flux, the discussion is given in §4.3. Since we know the value of solution on the interface, we could discretize u_x and u_{xx} using one sided interpolation as usual. For example, if $x_j \le \alpha^n < x_{j+1}$ then

$$u^n_{x,j} = u^n_x(j-1, j, \alpha^n, x_j),$$

where

$$\begin{aligned}u_x(j-1, j, \alpha, x) &= \frac{x_j + \alpha - 2x}{h(x_{j-1} - \alpha)}u_{j-1} + \frac{x_{j-1} + \alpha - 2x}{h(\alpha - x_j)}u_j \\ &\quad + \frac{x_{j-1} + x_j - 2x}{(x_j - \alpha)(\alpha - x_{j-1})}r(t).\end{aligned} \tag{18}$$

This is a second order approximation to $u_x(x,t)$. However, notice that $\alpha(t)$ changes with time, so does $x_j - \alpha$. If $|x_j - \alpha|$ gets too small, the magnitudes of the coefficients in the interpolation (18) become very large, sometimes even blow up. Such instability is caused from the formulation of the derivatives which is very sensitive to the location of the interface. An intuitive fix would be

$$u^n_{x,j} = u^n_x(j-2,\, j-1,\, \alpha^n, x_j),$$

when $|x_j - \alpha|$ is small. A more robust way which we have been using successfully is the linear combination of those two above:

$$\boxed{\begin{aligned} u^n_{x,j} &= \frac{\alpha^n - x_j}{h}\, u^n_x(j-1,\, j,\, \alpha^n, x_j) \\ &\quad + \frac{x_{j+1} - \alpha^n}{h}\, u^n_x(j-2,\, j-1,\, \alpha^n, x_j). \end{aligned}} \tag{19}$$

There are several advantages of this robust approach. First of all, the interpolation is still second order accurate. Secondly, if we rewrite (19) as

$$u^n_{x,j} = \sum_{j=j-2}^{j} \gamma^n_j\, u^n_j + \gamma^n_\alpha\, r(t^n),$$

then the magnitudes of the coefficients γ^n_j and γ^n_α will always be order $O(1/h)$. Furthermore the truncation error in such interpolation will smoothly vary as time increases. This is very significant in two or higher dimensional moving interface problems where we want to avoid non-physical oscillations.

In the same manner, we use the following interpolation to discretize u^n_{x,j_0+1}:

$$\boxed{\begin{aligned} u^n_{x,j+1} &= \frac{\alpha^n - x_j}{h}\, u^n_x(j+2,\, j+3,\, \alpha^n, x_{j+1}) \\ &\quad + \frac{x_{j+1} - \alpha^n}{h}\, u^n_x(j+1,\, j+2,\, \alpha^n, x_{j+1}). \end{aligned}} \tag{20}$$

Similarly, using the following second order discretization for u_{xx},

$$\begin{aligned} u_{xx}(j-1,\, j,\, \alpha, x) &= \frac{2}{h(\alpha - x_{j-1})}\, u_{j-1} + \frac{2}{h(\alpha - x_j)}\, u_j \\ &\quad + \frac{2}{(x_{j-1} - \alpha)(x_j - \alpha)}\, r(t). \end{aligned} \tag{21}$$

we can compute $u^n_{xx,j}$ and $u^n_{xx,j+1}$ as follows,

$$\boxed{\begin{aligned} u^n_{xx,j} &= \frac{\alpha^n - x_j}{h}\, u^n_{xx}(j-1,\, j,\, \alpha^n, x_j) \\ &\quad + \frac{x_{j+1} - \alpha^n}{h}\, u^n_{xx}(j-2,\, j-1,\, \alpha^n, x_j). \end{aligned}} \tag{22}$$

$$\boxed{\begin{aligned} u^n_{xx,j+1} &= \frac{\alpha^n - x_j}{h}\, u^n_{xx}(j+2,\, j+3,\, \alpha^n, x_{j+1}) \\ &\quad + \frac{x_{j+1} - \alpha^n}{h}\, u^n_{xx}(j+1,\, j+2,\, \alpha^n, x_{j+1}). \end{aligned}} \tag{23}$$

At time level $l = n+1$, we still can use the same trick when x_j is too close to α^{n+1}. But the resulting linear system (if we freeze the nonlinear term uu_x) is no longer tridiagonal, since an additional point $j-2$ is involved at grid point x_j. However in this case we can simply set

$$\boxed{u_j^{n+1} = r(t^{n+1}), \quad \text{if} \quad |x_j - \alpha^{n+1}| \le h^2} \tag{24}$$

without affecting second order accuracy. By doing so we will still have a tridiagonal system for the linearized equations.

4.2. Case 2: The jump conditions are known. Suppose we know the jump conditions (3), (4) across the interface, $[u] = q(t)$ and $[\beta u_x] = v(t)$. We also should have knowledge of $[\beta(t)]$, $[f(t)]$ across the interface. As in [12], [15], the immersed interface method for the space discretization involves the following steps:

- Use the jump conditions and the differential equation to get the interface relations between the quantities on each side of the interface.
- Use the interface relations to derive a modified difference scheme.
- Derive the correction term based on the difference scheme and the interface relations.

With this process, we have the following theorem:

THEOREM 4.1. *If* $x_j \le \alpha(t) < x_{j+1}$, *then*

$$\begin{bmatrix} u(\alpha^-,t) \\ u_x(\alpha^-,t) \\ u_{xx}(\alpha^-,t) \end{bmatrix} = S \begin{bmatrix} u(x_{j-1},t) \\ u(x_j,t) \\ u(x_{j+1},t) \end{bmatrix} - \begin{bmatrix} C_{j,1} \\ C_{j,2} \\ C_{j,3} \end{bmatrix} + \begin{bmatrix} O(h^3) \\ O(h^2) \\ O(h) \end{bmatrix}, \tag{25}$$

where for $k = 1,2,3$,

$$\begin{aligned} C_{j,k} &= s_{k3}\left\{ q + \frac{v}{\beta^+}(x_{j+1}-\alpha) \right. \\ &\quad \left. + \frac{(x_{j+1}-\alpha)^2}{2\beta^+}\left(q' + [f] - \frac{v}{\beta^+}\left(w - \lambda u^+ + \beta_x^+\right)\right)\right\}, \end{aligned} \tag{26}$$

q and v are defined as in (3) and (4), and $S = \{s_{kj}\} = A^{-1}$, the inverse of the following matrix

$$A^T = \{a_{kj}\}^T = \begin{bmatrix} 1 & 1 & 1 \\ x_{j-1}-\alpha & x_j - \alpha & a_{23} \\ \dfrac{(x_{j-1}-\alpha)^2}{2} & \dfrac{(x_j-\alpha)^2}{2} & \dfrac{\rho\,(x_{j+}-\alpha)^2}{2} \end{bmatrix}$$

with

$$a_{23} = \rho(x_{j+1}-\alpha) + \frac{(x_{j+1}-\alpha)^2}{2\beta^+}\left(w(1-\rho) + \lambda(u^+\rho - u^-) + \beta_x^- - \rho\beta_x^+\right).$$

The proof can be found in the Appendix.

Using this theorem we get a discretized form of $(\beta u_x)_x$ at the grid point x_j, $x_j \le \alpha < x_{j+1}$,

$$\boxed{\begin{aligned} &(\beta u_x)_{x,j} - \lambda u_j u_{x,j} = (\beta u_x)_x^- - \lambda u^- u_x^- + O(h) \\ &\approx \left(\beta^- s_{31} + s_{21}(\beta_x^- - \lambda u^-)\right) u_{j-1} + \left(\beta^- s_{32} + s_{22}(\beta_x^- - \lambda u^-)\right) u_j \\ &\quad + \left(\beta^- s_{33} + s_{23}(\beta_x^- - \lambda u^-)\right) u_{j+1} - \left(\beta^- C_{j,3} + C_{j,2}(\beta_x^- - \lambda u^-)\right). \end{aligned}} \tag{27}$$

The attractive aspect of this approach is that we can still use a three-point stencil and the discretization is valid for any location α. The theorem above also gives an interpolation formula which can be used to compute u^- and u_x^- values which are needed for the computation of w and the frozen term $\lambda u u_x$ (see §5).

For the grid point x_{j+1}, $x_j \le \alpha < x_{j+1}$, there is a similar formula which we state as follows:

THEOREM 4.2. *Let $u(x,t)$ be the solution of (1) and (2) with jump condition (3) and (4). If $x_j \le \alpha(t) < x_{j+1}$, then*

$$\begin{bmatrix} u(\alpha^+,t) \\ u_x(\alpha^+,t) \\ u_{xx}(\alpha^+,t) \end{bmatrix} = \tilde{S} \begin{bmatrix} u(x_j,t) \\ u(x_{j+1},t) \\ u(x_{j+2},t) \end{bmatrix} - \begin{bmatrix} C_{j+1,1} \\ C_{j+1,2} \\ C_{j+1,3} \end{bmatrix} + \begin{bmatrix} O(h^3) \\ O(h^2) \\ O(h) \end{bmatrix}, \tag{28}$$

where for $k = 1,2,3$

$$\begin{aligned} C_{j+1,k} &= \tilde{s}_{k1} \left\{ -[u] - \frac{[\beta u_x]}{\beta^-}(x_j - \alpha) \right. \\ &\left. -\frac{(x_j-\alpha)^2}{2\beta^-}\left(q' + [f] - \frac{v}{\beta^-}\left(w - \lambda u^- + \beta_x^-\right)\right)\right\}, \end{aligned} \tag{29}$$

and $\tilde{S} = \{\tilde{s}_{kj}\} = \tilde{A}^{-1}$, the inverse of the following matrix

$$\tilde{A}^T = \{\tilde{a}_{kj}\}^T = \begin{bmatrix} 1 & 1 & 1 \\ \tilde{a}_{21} & x_{j+1}-\alpha & x_{j+2}-\alpha \\ \dfrac{(x_j-\alpha)^2}{2\rho} & \dfrac{(x_{j+1}-\alpha)^2}{2} & \dfrac{(x_{j+2}-\alpha)^2}{2} \end{bmatrix},$$

with

$$\tilde{a}_{21} = \frac{(x_j-\alpha)}{\rho} - \frac{(x_j-\alpha)^2}{2\beta^-}\left(w(1-\frac{1}{\rho}) + \lambda(\frac{u^-}{\rho} - u^+) - \frac{\beta_x^-}{\rho} + \beta_x^+\right).$$

4.3. Stefan problems. For a number of Stefan problems, the governing equations have the form

$$\begin{aligned} u_t &= (\beta u_x)_x \\ \frac{d\alpha}{dt} &= \sigma(t)[\beta u_x] \\ u(\alpha,t) &= u_0. \end{aligned} \tag{30}$$

So we know the solution on the interface as well as the jump relations $[u] = 0$ and $[\beta u_x] = \dot{\alpha}/\sigma$.

It is easy to see that the equation (30) can be discretized using either of the approaches described in §4.1 or §4.2. We have found out that the approach using the jump relations described in §4.2 is better. However, we need one more equation to make the discrete system closed because now $w = \sigma[\beta u_x]$ is unknown. This equation is the restriction of the solution on the interface

$$\int_0^1 u(x,t)\,\delta(x-\alpha)dx = u_0.$$

A second order accurate discretization can be easily derived from Theorem 4.1

$$s_{11}u_{j-1} + s_{12}u_j + s_{13}u_{j+1} - C_{j,1} = u_0$$

5. Computation of the quantities on the interface. As we mentioned in Section 2, we need to know some quantities defined only on the interface such as $u^{\pm}$, $u_x^{\pm}$, $[u_t]_\tau$ in order to compute (9), (11), (16) and (27). Again we distinguish the two different cases.

5.1. Case 1: the solution on the interface $u(\alpha, t) = r(t)$ is known. In this case, the solution is continuous, which means $u^- = u^+ = r(t)$.

With the knowledge of the computed solution u_i^n, an estimation of u_i^{n+1}, and the solution on the interface $r(t^n)$, we use the one sided difference (19) exchanging the position between x_j and α^n, $x_j \le \alpha^n < x_{j+1}$, to compute $u_x^{-,n}$, and (20) exchanging the position between x_{j+1} and α^n to compute $u_x^{+,n}$. The same approach is used for the next time level t^{n+1}.

If the interface crosses a grid line $x = x_j$ at some time τ, we need to compute $[u_t]_{;\tau}$ in order to get the correction term $Q_j^{n+\frac{1}{2}}$. In this case we simply use

$$[u_t]_{;\tau} = \frac{u_j^{n+1} - r(\tau)}{t^{n+1} - \tau} - \frac{r(\tau) - u_j^n}{\tau - t^n}. \tag{31}$$

5.2. Case 2: the jump conditions are known. In this case, $u^{\pm}$ and $u_x^{\pm}$ are computed using (25)–(26) and (28)–(29). In order to compute the jump $[u_t]_\tau$, we differentiate the first jump condition

$$u(\alpha^+, t) - u(\alpha^-, t) = q(t)$$

with respect to t to get

$$(u_x(\alpha^+, t) - u_x(\alpha^-, t)) \frac{d\alpha}{dt} + u_t(\alpha^+, t) - u_t(\alpha^-, t) = q'(t),$$

$$i.e., \qquad [u_t] = q'(t) - [u_x] w. \tag{32}$$

We need to express (32) in terms of the quantities at time level either t^n or t^{n+1}. If $\alpha^n \ge \alpha^{n+1}$, (see Fig. 1 (b)), we have, at time $t = \tau$,

$$u^- = u_j^n + O(h), \qquad u^+ = u_j^{n+1} + O(h),$$
$$u_x^- = u_{x,j}^n + O(h), \qquad u_x^+ = u_{x,j}^{n+1} + O(h).$$

Otherwise we have

$$u^- = u_j^{n+1} + O(h), \qquad u^+ = u_j^n + O(h),$$
$$u_x^- = u_{x,j}^{n+1} + O(h), \qquad u_x^+ = u_{x,j}^{n+1} + O(h),$$

for $\alpha^n < \alpha^{n+1}$. Thus we use following scheme to compute $[u_t]_\tau$

$$\boxed{[u_t]_{;\tau} = \begin{cases} q'(\tau) - w(\tau, u_j^n, u_j^{n+1}, u_{x,j}^n, u_{x,j}^{n+1})(u_{x,j}^{n+1} - u_{x,j}^n) & \text{if } \alpha^n \ge \alpha^{n+1} \\ q'(\tau) - w(\tau, u_j^{n+1}, u_j^n, u_{x,j}^{n+1}, u_{x,j}^n)(u_{x,j}^{n+1} - u_{x,j}^n) & \text{if } \alpha^n < \alpha^{n+1}. \end{cases}} \tag{33}$$

6. Solving the resulting nonlinear system of equations. From the discussion above we know that in order to get the solution $u(x,t)$ at time t^{n+1}, generally we need to solve the following nonlinear system:

$$\frac{u_i^{n+1}-u_i^n}{k} - Q_i^{n+\frac{1}{2}} + \frac{\lambda}{2}\left(u_i^n u_{x,i}^n + u_i^{n+1}u_{x,i}^{n+1}\right) = \frac{1}{2}\left((\beta u_x)_{x,i}^n + (\beta u_x)_{x,i}^{n+1}\right) - \frac{1}{2}\left(f_i^n + f_i^{n+1}\right),$$

$$\frac{\alpha^{n+1}-\alpha^n}{k} = \frac{1}{2}\left(w^n + w^{n+1}\right),$$

where the quantities of $u_{x,i}^l$ and $(\beta u_x)_{x,i}^l$ for $l=n$ or $n+1$, can be expressed as some linear combination of u_i^l. The coefficients of such combination near the interface, and the correction term $Q_i^{n+\frac{1}{2}}$ to $(u_i^{n+1}-u_i^n)/k$, depend on the interface position and the interface conditions. We have shown how to get these quantities in previous sections for different interface conditions.

Since we use a full implicit discretization, the numerical scheme is stable. The local truncation errors are $O(h^2)$ at most grid points, but $O(h)$ at two grid points which are closest to the interface from the left and the right. So the global error in the solution is second order accurate at all grid points.

So we have a quite complicated nonlinear system to solve. The difficulty is that some quantities such as $Q_j^{n+\frac{1}{2}}$, $C_{j,k}^{n+1}$ are not known until we know the solution for u_i^{n+1} and α^{n+1}. This leads to an implicit system, which must be solved iteratively. One concern about iterative methods is whether there is a time step restriction. In our approach we use an implicit discretization for the diffusion term $(\beta u_x)_x$ and an explicit discretization for the motion of $\alpha(t)$ in which the CFL condition is $k \sim h$. Therefore we do not need to worry about the time step restriction unless the interface changes rapidly. In that case, the stiffness of the motion will require that in order to achieve desired accuracy, we must take a small time step anyway. An adaptive time step is chosen for equation (2) based on the classic stability theory

$$k \le min\left\{h,\ \left|\frac{1}{\partial w/\partial \alpha}\right|\right\}.$$

The constrain $k \le h$ is imposed to maintain second order accuracy both in space and time. From equation (2) we can get

$$\frac{\partial \alpha}{\partial w}\frac{\partial w}{\partial t} = w, \qquad i.e. \qquad \frac{\partial w}{\partial \alpha} = \frac{\partial w}{\partial t}/w.$$

That implies

$$k \le min\left\{h,\ \left|\frac{w}{\partial w/\partial t}\right|\right\},$$

or in discrete form

$$\boxed{k_{new} = min\left\{h,\ \left|\frac{w^n\, k_{old}}{w^{n+1}-w^n}\right|\right\}}. \tag{34}$$

Below we give an outline of our iterative process.

Suppose we have obtained all necessary quantities at the time t^n, and the current time step is k (*i.e.* $t^{n+1}=t^n+k$). We want to get all corresponding quantities at time level t^{n+1}. Unfortunately we have to introduce another subscript m for each iteration.

- Determine j_0 such that $x_{j_0} \le \alpha^n < x_{j_0+1}$. Compute u_x^n, $(\beta u_x)_x^n$ at x_{j_0} according to the scheme discussed in Section 4.
- Set
$$\alpha_1^{n+1} = \alpha^n + k\, w(t^n, \alpha^n;\, u^{-,n},\, u^{+,n},\, u_x^{-,n},\, u_x^{+,n}).$$
- Determine an initial guess of the solution $u_{i,1}^{n+1}$ at time level t^{n+1}.

For $m = 1, 2, \cdots$,

- (**) Determine j_m such that $x_{j_m} \le \alpha_1^{n+1} < x_{j_m+1}$. Determine the coefficients and the correction terms for u_x^{n+1} and u_{xx}^{n+1} at x_{j_m}. Substitute $u_{i,m}^{n+1}$ for u_i^{n+1} in the nonlinear term $u_i^{n+1} u_{x,i}^{n+1}$.
- If $j_0 \ne j_m$, then for $l = j_0 + 1, \cdots, j_m$, when $j_0 < j_m$, or for $l = j_m, j_m + 1, \cdots, j_0$, when $j_0 > j_m$, first get τ_m using (16), then determine the correction $Q_{l,m}^{n+\frac{1}{2}}$ to $(u_{l,m}^{n+1} - u_l^n)/k$ using the technique described in Section 3.
- Solve the tridiagonal system for $u_{i,m+1}^{n+1}$.
- Interpolate $\{u_{i,m+1}^{n+1}\}$ to get $u_{m+1}^{\pm,n+1}$, $u_{x,m+1}^{\pm,n+1}$, if necessary.
- Determine
$$\alpha_{m+1}^{n+1} = \alpha^n + \frac{k}{2}\left(w^n + w_{m+1}^{n+1}\right).$$
where
$$w_{m+1}^{n+1} = w\left(t^{n+1}, \alpha_m^{n+1}; u_{m+1}^{-,n+1}, u_{m+1}^{+,n+1}, u_{x,m+1}^{-,n+1}, u_{x,m+1}^{+,n+1}\right).$$
- If $|\alpha_m^{n+1} - \alpha_{m+1}^{n+1}| > \epsilon$, a given tolerance, then $m = m + 1$. Go to (**).
- If $|\alpha_m^{n+1} - \alpha_{m+1}^{n+1}| < \epsilon$, then set all quantities $\{\ \}_m^{n+1}$ to $\{\ \}^{n+1}$, in other words we drop the $\{m\}$ notation and accept these values at time t^{n+1}. Determine next time step as
$$k = min\left\{ h,\ \left| \frac{w^n\, k}{w^{n+1} - w^n} \right| \right\}. \tag{35}$$
Go to the next time step.

7. Numerical examples. Here we present three different examples. The first two are from real applications. The third one is a constructed example for the nonlinear moving interface problem.

Example 1. Two phase Stefan problem. This is a classical example in tracking a freezing front of ice in water. The description of the problem is excerpted from [8], where Furzeland used this example to compare different methods. The thermal properties are k_i = conductivity, $C_i = c_i\,\rho$ with c_i = specific heat and ρ = density (assume the same in each phase), $\lambda = L\rho$ with L = latent heat. Subscript $i = 1$ denotes phase 1 (ice), $0 < x < \alpha(t)$; $i = 2$ denotes phase 2 (water), $\alpha(t) < x < 1$ (truncated from infinity). The equations are

$$C_i \frac{\partial u_i}{\partial t} = k_i \frac{\partial^2 u_i}{\partial x^2}, \qquad i = 1, \text{ and } 2, \quad t_0 < t < t^*,$$
$$u_1 = u^* < 0, \quad x = 0,\ t > 0,$$

$$\left.\begin{aligned} & u_1 = u_2 = 0 \\ & \lambda \frac{\partial \alpha}{\partial t} = k_1 \frac{\partial u_1}{\partial x} - k_2 \frac{\partial u_2}{\partial x} \end{aligned}\right\} \quad \text{on} \quad x = \alpha(t), \;\; t_0 < t.$$

This problem has an exact solution

$$\alpha(t) = 2\,\phi\,\sqrt{\kappa_1 t}\,,$$

$$u_1 = u^* \left\{ 1 - \frac{\operatorname{erf}(x/2\sqrt{\kappa_1 t}\,)}{\operatorname{erf}\phi} \right\},$$

$$u_2 = u_0 \left\{ 1 - \frac{\operatorname{erfc}(x/2\sqrt{\kappa_2 t}\,)}{\operatorname{erfc}(\phi\sqrt{\kappa_1/\kappa_2})} \right\},$$

where $\kappa_i = k_i/C_i$, erf is the error function, and ϕ is the root of the transcendental equation

$$\frac{e^{-\phi^2}}{\operatorname{erf}\phi} + \frac{k_2}{k_1}\sqrt{\frac{\kappa_1}{\kappa_2}}\,\frac{u_0 e^{-\kappa_1\phi^2/\kappa_2}}{u^*\operatorname{erfc}(\phi\sqrt{\kappa_1/\kappa_2})} + \frac{\phi\lambda\sqrt{\pi}}{C_1 u^*} = 0,$$

which can be easily computed, say using the bisection method. The exact solution is used as the boundary condition at both ends, $x = 0$, and $x = 1$. The following thermal properties are used

$$k_1 = 2.22, \quad k_2 = 0.556, \quad C_1 = 1.762, \quad C_2 = 4.226, \quad \lambda = 338,$$

with $t^* = 0.288$, $u^* = -20$ and $u_0 = 10$ which gives $\phi = 0.2054269\cdots$.

TABLE 1
Grid refinement analysis for Example 1 at $t = 1.0$.

N	$\parallel E_N \parallel_\infty$	ratio	$\lvert E_\alpha \rvert$	ratio
20	4.3067×10^{-3}		1.0941×10^{-4}	
40	9.7147×10^{-4}	4.4333	2.4947×10^{-5}	4.3857
80	2.3713×10^{-4}	4.0967	5.7298×10^{-6}	4.3539
160	5.8160×10^{-5}	4.0772	1.3828×10^{-6}	4.1434
320	1.4213×10^{-5}	4.0920	3.3721×10^{-7}	4.1009

Table 1 shows the results of a grid refinement analysis, where $\parallel E_N \parallel_\infty$ is defined as the infinity norm of the error at the fixed time t, i.e.,

$$\parallel E_N \parallel_\infty = \max_i \left\{ \left| u(x_i, t) - u_i^N \right| \right\},$$

where N is the number of grid points as defined in Section 2. We use u_i^N as the computed solution at the uniform grid points x_i, $i = 1, 2, \ldots,$ at some time t. E_α is the error between $\alpha(t)$ and the computed interface at the time t. We see that doubling the number of grid points give a reduction in both errors by a factor of roughly 4, indicating second order accuracy. Figure 2 (a) shows the true solution and the computed solution for $N = 40$. Figure 2 (b) shows the corresponding error plot.

(a) (b)

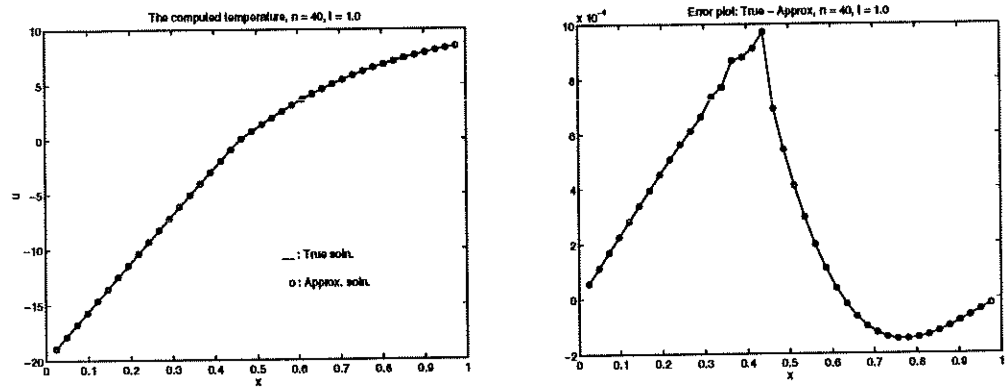

FIG. 2. *The comparison of the exact and computed solution at $t = 1.0$ for Example 1 with $N = 40$. (a) The solid line is the exact solution and the dots are the computed results at grid points. (b) The corresponding error plot.*

We see that the error in the solution u is relatively large around the interface compared to other grid points but not significantly so. Globally we obtain second order accurate results at all grid points (see Table (1)). We also tested a similar example described by Keller [11] for modeling a melting and freezing problem at constant speed. The results again agreed with our analysis.

Example 2. This simulation shows the temperature profile of an ice sheet during the process of a glaciation. Heine and McTigue [9] proposed a one-dimensional thermal model to study the temperature change of a glacier. The ice sheet gradually grows to a thickness of $3000m$ over approximately $10,000$ years. At the same time, a heat source, in the form of a geothermal heat flux, is warming the glacier from a depth of $4000m$ in the rock. The point of interest in this problem is the interface between ice and rock. If the temperature approaches the melting point, the glacier may begin to slide with catastrophic effect. A similar problem can be found in [10].

The mathematical model Heine and McTigue used is similar to Example 1, see Fig.3.

$$\rho c_p \frac{\partial T}{\partial t} = \frac{\partial}{\partial x}\left(k\frac{\partial T}{\partial x}\right), \qquad 0 \le x \le h(t), \ \ t > 0.$$

However both ρc_p and k are functions of T now. So this is a nonlinear model for the temperature. The estimated height of the ice sheet is

$$h(t) = h_0 + \delta_h + (h_\infty - h_0 - \delta_h)(1 - e^{-t/t_r}),$$

where h_0 is the thickness of rock, δ_h is an (arbitrarily) small initial ice layer thickness, h_∞ is the final total thickness (rock + ice), and t_r is the rise time for the ice sheet growth. The boundary conditions consist of constant heat flux from the inner layer of the earth and constant temperature at the top of the ice sheet:

$$T_x(0.t) = -q_0; \qquad T(h(t), t) = T_s.$$

The thermal properties are

$$k_{ice} = C_1\, e^{-c_2(T+273.15)}, \quad k_{rock} = constant,$$
$$(\rho c_p)_{ice} = r_1 + r_2 T, \quad (\rho c_p)_{rock} = constant.$$

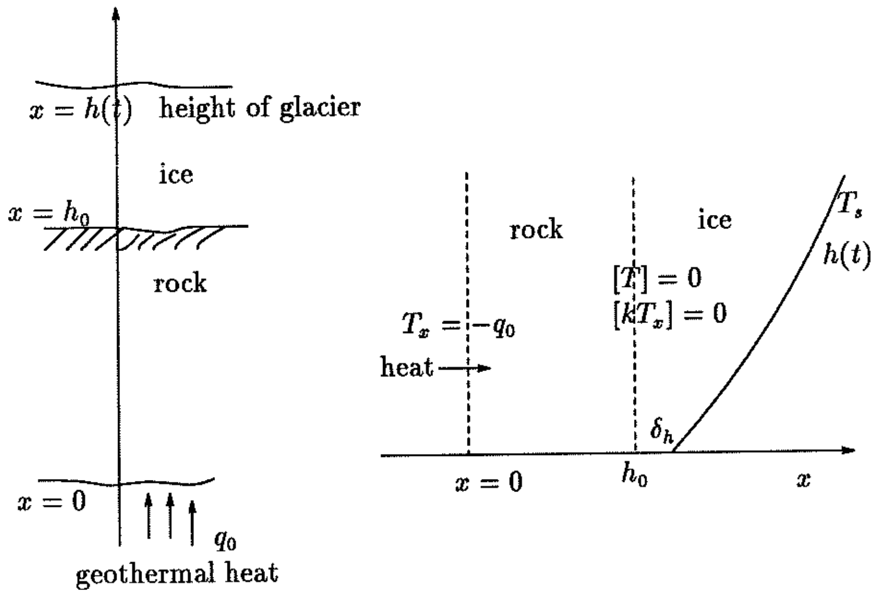

FIG. 3. *Geometrical frame of Heine and McTigue's glacier model.*

The initial conditions are

$$T = \begin{cases} T_{h_0} + \dfrac{q_0}{k_{rock}}(h_0 - x) & \text{if} \quad 0 \le x \le h_0 \\ -\dfrac{1}{C_2}\log\left(e^{-C_2 T_s} - C_2(h_0 + \delta_h - x)\Gamma\right) & \text{if} \quad h_0 < x < h_\infty, \end{cases}$$

where

$$\Gamma = \frac{q_0}{C_1} e^{-273.15 C_2},$$

$$T_{h_0} = -\frac{1}{C_2}\log\left(e^{-C_2 T_s} - C_2\delta_h\right).$$

The following thermal properties are used

$$C_1 = 9.828, \;\; C_2 = 0.0057, \;\; r_1 = 1.936 \times 10^6; \; r_2 = 6.600 \times 10^6$$

$$k_{rock} = 2.50, \;\; (\rho c_p)_{rock} = 2.30 \times 10^6; \; q_0 = 0.05 w/m^2, \;\; T_s = -25.0^o C.$$

The geometrical parameters are

$$h_0 = 4000m, \;\; \delta_h = 1m, \;\; h_\infty = 7000m, \;\; t_r = 1.9 \times 10^{12}\, s.$$

In this example, we have a fixed interface $x = h_0$ and natural jump conditions $[T] = 0$ and $[k\partial T/\partial x] = 0$ since temperature is continuous and there is no heat source at the interface. This corresponds to Case 2 in our previous discussion with $w \equiv 0$. On the other hand, at the moving boundary $h(t)$ we know the surface temperature T_s. So it is also a one-phase nonlinear moving boundary problem.

Fig. 4 shows the temperature history of the interface between rock and ice. The geo-physical implication of the result can be found in [9].

To check the correctness of our algorithm, we constructed a problem for which we have an exact solution within the same geometrical frame. The numerical results

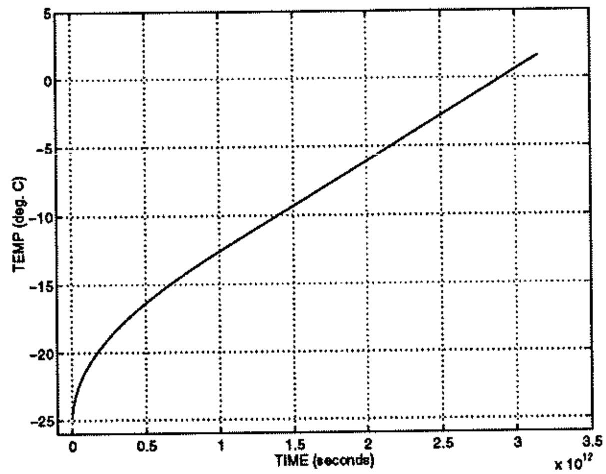

FIG. 4. *Temperature on the interface between the rock and ice.*

confirmed that our method converges to the exact solution with second order accuracy. The details are omitted here due to space limitation.

Example 3. Nonlinear moving interface example for Case 2. In this example, we take $\lambda = 1$ in (1) and construct the following exact solution

$$u(x,t) = \begin{cases} \sin(\omega_1 x) e^{-\beta^- \omega_1^2 t} & \text{if } x \le \alpha(t) \\ \sin(\omega_2 - \omega_2 x)\, e^{-\beta^+ \omega_2^2 t} & \text{if } x > \alpha(t), \end{cases} \tag{36}$$

for some choice of ω_1, ω_2, β^-, and β^+. The source term $f(x,t)$ is discontinuous

$$f(x,t) = \begin{cases} -\frac{1}{2}\omega_1 \sin(2\omega_1 x) e^{-2\beta^- \omega_1^2 t} & \text{if } x \le \alpha(t) \\ \frac{1}{2}\omega_2 \sin(2\omega_2 - 2\omega_2 x) e^{-2\beta^+ \omega_2^2 t} & \text{if } x > \alpha(t)\,. \end{cases} \tag{37}$$

We assume that the solution $u(x,t)$ is continuous across $\alpha(t)$. So the interface $\alpha(t)$ can be determined from the scalar equation

$$\sin(\omega_1 \alpha)\, e^{-\beta^- \omega_1^2 t} = \sin(\omega_2 - \omega_2 \alpha)\, e^{-\beta^+ \omega_2^2 t}. \tag{38}$$

This equation has a unique solution if we take, for example, $\pi < \omega_1 < 2\pi$ and also $\pi < \omega_2 < 2\pi$. Figure 5 gives the plot of $\alpha(t)$ as the parameters (β^-, β^+) changes on a uniform grid. We can see how the interface crosses the grid. This example is adapted from [2].

The nonlinear ordinary differential equation for the motion is

$$\frac{d\alpha}{dt} = \frac{(\omega_1^2 - \omega_2^2)\, u(\alpha, t)}{u_x(\alpha^-, t) - u_x(\alpha^+, t)}. \tag{39}$$

The initial and boundary conditions are

$$u(0,t) = 0, \qquad u(1,t) = 0, \tag{40}$$

$$u(x,0) = \begin{cases} \sin(\omega_1 x) & \text{if } x \le \alpha(0), \\ \sin(\omega_2 - \omega_2\, x) & \text{if } x \ge \alpha(0). \end{cases} \tag{41}$$

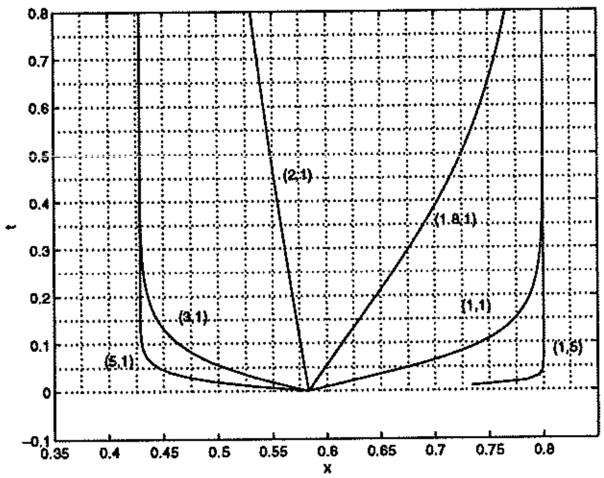

FIG. 5. *Moving interface* $\alpha(t)$, $0 \le t \le 1.$, *from left to right*, $(\beta^-, \beta^+) = (5,1)$, $(3,1)$, $(2,1)$, $(1.8,1)$, $(1,1)$ *and* $(1,5)$.

The jump conditions used are

$$
\begin{aligned}
[u] &= 0, \\
[\beta u_x(\alpha, t)] &= -\beta^+ \omega_2 \cos(\omega_2 - \omega_2 \alpha)\, e^{-\beta^+ {\omega_2}^2 t} - \beta^- \omega_1 \cos(\omega_1 \alpha)\, e^{-{\omega_1}^2 t}.
\end{aligned}
$$

Figure 6 shows a typical computed profile of $u(x,t)$ as time changes. We can clearly see how the interface moves and crosses the grid with time. This example is more challenging than traditional examples because the jump in the coefficient and in the derivative of the solution are significantly larger than in the previous examples.

(a) (b)

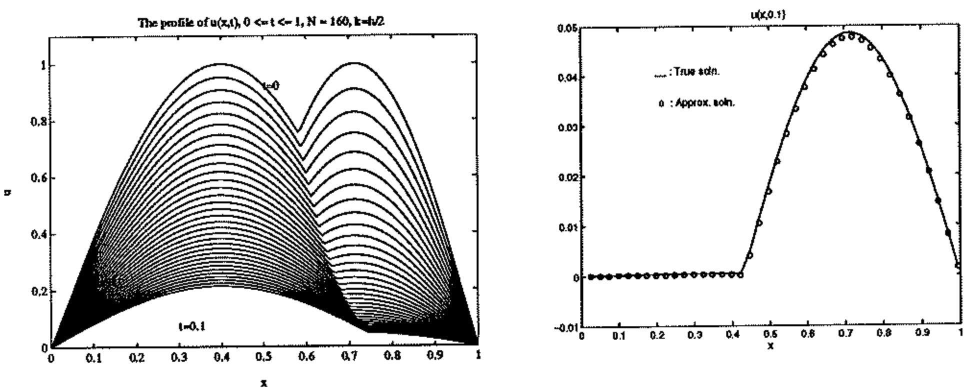

FIG. 6. (*a*) *The profile of the computed solution* $u(x,t)$ *from* $t=0$ *to* $t=0.1$ *with* $\beta^- = \beta^+ = 1$, *and* $N=160$; (*b*) *The comparison of the exact and computed solution at* $t = 1.0$ *with* $\beta^- = 5.0$, $\beta^+ = 1.0$, *and* $N = 80$. *The solid line is the exact solution and the dots are the computed results at the grid points.*

Table 2 shows the grid refinement analysis both for the computed solution and the interface. In this case, the interface crosses several grid points during the first few time steps (see Fig 5). However second order accuracy is still achieved.

TABLE 2
Grid refinement analysis for Example 3 at $t = 0.1$.

N	$\| E_N \|_\infty$	ratio	$\|E_\alpha\|$	ratio
40	6.5460×10^{-3}		2.2605×10^{-2}	
80	6.0108×10^{-3}	1.0890	8.6674×10^{-3}	2.6081
160	1.1557×10^{-3}	5.2009	2.7449×10^{-3}	3.1575
320	2.9347×10^{-4}	3.9381	6.9958×10^{-4}	3.9238
640	7.3175×10^{-5}	4.0106	1.7508×10^{-4}	3.9957
1280	1.8237×10^{-5}	4.0124	4.3725×10^{-5}	4.0042
2560	4.5462×10^{-6}	4.0115	1.0913×10^{-5}	4.0065
5120	1.1370×10^{-6}	3.9982	2.7311×10^{-6}	3.9960

8. Summary. In summary we have developed a second order accurate immersed interface method for a class of one dimensional nonlinear moving interface problems with two typical interface conditions: (i). Stefan-like problems which we know the solution on the interface. (ii). Problems in which we only know the jump conditions in the solution and the flux across the interface. Applications include the immersed boundary method and heat conduction in different materials. Numerical experiments have confirmed the efficiency of the methods proposed in this paper. Currently we are working on similar numerical methods in two dimensions and trying to explore the possibilities of using the level set technique [19] to track the interface.

9. Appendix — Proof of Theorem 4.1 . Here we present a brief proof of Theorem 4.1. We first derive one more interface relation using the known information and then give the proof of the theorem.

LEMMA 9.1. *Let $u(x,t)$ be the solution of (1)-(2) with the jump condition (3) and (4), then*

$$u_{xx}^+ = \rho u_{xx}^- + \frac{1}{\beta^+}\left\{ q' + [f] - \frac{v}{\beta^+}\left(w - \lambda u^+ + \beta_x^+\right) + u_x^-\left(w(1-\rho) + \lambda(u^+\rho - u^-) + \beta_x^- - \frac{\beta^-}{\beta^+}\beta_x^+\right)\right\}, \tag{42}$$

where $\rho = \beta^-/\beta^+$.

Proof: From the equation (1) we know

$$\beta^+ u_{xx}^+ + \beta_x^+ u_x^+ - \lambda u^+ u_x^+ - f^+ - u_t^+ = \beta^- u_{xx}^- + \beta_x^- u_x^- - \lambda u^- u_x^- - f^- - u_t^- . \tag{43}$$

Plugging (32) and the second jump condition (4) into (43), arranging terms we get (42).

Proof of theorem 4.1: Expanding $u(x_{j-1},t)$, $u(x_j,t)$ from left side, and $u(x_{j+1},t)$ from the right side of α, we have

$$u(x_{j-1},\alpha) = u^- + (x_{j-1}-\alpha)\, u_x^- + \frac{1}{2}(x_{j-1}-\alpha)^2\, u_{xx}^- + O(h^3), \tag{44}$$

$$u(x_j,\alpha) = u^- + (x_j-\alpha)\,u_x^- + \frac{1}{2}(x_j-\alpha)^2\,u_{xx}^- + O(h^3), \tag{45}$$

$$u(x_{j+1},\alpha) = u^+ + (x_{j+1}-\alpha)\,u_x^+ + \frac{1}{2}(x_{j+1}-\alpha)^2\,u_{xx}^+ + O(h^3). \tag{46}$$

So

$$\begin{aligned}\sum_{k=j-1}^{j+1} s_{i,k-j+2}\,u(x_k,t) \;=\;& s_{i1}\left(u^- + (x_{j-1}-\alpha)\,u_x^- + \frac{1}{2}(x_{j-1}-\alpha)^2\,u_{xx}^-\right)\\ &+ s_{i2}\left(u^- + (x_j-\alpha)\,u_x^- + \frac{1}{2}(x_j-\alpha)^2\,u_{xx}^-\right)\\ &+ s_{i3}\left(u^+ + (x_{j+1}-\alpha)\,u_x^+ + \frac{1}{2}(x_{j+1}-\alpha)^2\,u_{xx}^+\right)\\ &+ O(h^{4-i}),\end{aligned} \tag{47}$$

for $i = 1,2$ and 3. Notice that we have used the fact that s_{ij} is order $O(h^{1-i})$. From interface relation (3), (4) and (42) we can solve for u^+, u_x^+, and u_{xx}^+ in terms of u^-, u_x^-, and u_{xx}^- and substitute them into the expression above and collect terms to obtain

$$\begin{aligned}\sum_{k=j-1}^{j+1} s_{i,k-j+2}u(x_k,t) &= (s_{i1}+s_{i2}+s_{i3})\;u^- + (\,s_{i1}(x_{j-1}-\alpha) + s_{i2}(x_j-\alpha)\\ &\quad + s_{i3}\,a_{23})\,u_x^- + \left(\frac{(x_{j-1}-\alpha)^2}{2}\,s_{i1} + \frac{(x_j-\alpha)^2}{2}\,s_{i2} + \frac{(x_{j+1}-\alpha)^2}{2}\,\rho\,s_{i3}\right)u_{xx}^-\\ &\quad + s_{i3}\left\{q + \frac{(x_{j+1}-\alpha)v}{\beta^+} + \frac{(x_{j+1}-\alpha)^2}{2\beta^+}\left(q' + [f] - \frac{v}{\beta^+}\,(w - \lambda u^+ + \beta_x^+)\right)\right\} + O(h^{4-i})\\ &= \left(\sum_{k=1}^{3} s_{ik}a_{k1}, \sum_{k=1}^{3} s_{ik}a_{k2}, \sum_{k=1}^{3} s_{ik}a_{k3}\right)\cdot\begin{pmatrix} u^- \\ u_x^- \\ u_{xx}^-\end{pmatrix} + C_{j,i} + O(h^{4-i})\\ &= e_i^T\cdot\begin{pmatrix} u^- \\ u_x^- \\ u_{xx}^-\end{pmatrix} + C_{j,i} + O(h^{4-i}).\end{aligned}$$

This completes the proof the theorem.

10. Acknowledgments. I am indebted to Prof. R. J. LeVeque for the original motivation and many useful discussions. I would like also thank him and D. Calhoun for reading, correcting and commenting on this paper. I am grateful to Prof. D. McTigue for useful discussions concerning his glacier model. It is very beneficial talking to B. Merriman, S. Osher, J. Zou and other people about the problems in this paper.

REFERENCES

[1] R. Almgren, *Variational algorithms and patter formulation in dendritic solidification*, J. Comput. Phys., 106 (1993), pp. 337-354.

[2] R. P. Beyer and R. J. LeVeque, *Analysis of a one-dimensional model for the immersed boundary method*, SIAM J. Num. Anal., 29 (1992), pp. 332–364.

[3] edited by J. M. Chadam and H. Rasmussen, *Free Boundary Problems in Fluid Flow with Applications*, Priman Advanced publishing Program, 1993.

[4] S. Chen and S. Osher, *private communications*

[5] J. Crank, *Free and Moving Boundary Problems*, Oxford University Press, 1984.

[6] B. A. Finlayson, *Numerical Methods for Problems with Moving Fronts*, Ravenna Park Publishing, Inc., 1992.

[7] B. Fornberg and R. Meyer-Spasche, *A finite difference procedure for a class of free boundary problems*, J. Comput. Phys., 102 (1992), pp. 72–77.

[8] R. M. Furzeland, *A comparative study of numerical methods for moving boundary problems*, J. Inst. Maths Applics., 26 (1980), pp. 411-429.

[9] J. T. Heine and D. F. McTigue, *A cold-based Laurentide ice sheet during the early to middle Wisconsin glaciation based on a transient thermal model*, EOS (Transactions of the American Geophysical Union), November, 1994.

[10] K. Hutter, A. Zryd and H. Röthlisberger, *On the numerical solution of Stefan problems in temperature ice*, J. of Glaciology, Vol. 36(1990), No. 122.

[11] J. B. Keller, Melting and freezing at constant speed, Phys. Fluids, 39 (6), 1986.

[12] R. J. LeVeque and Z. Li, *The Immersed Interface Method for Elliptic Equations with Discontinuous Coefficients and Singular Sources*, SIAM J. Num. Anal., 31 (1994), pp. 1019-1014.

[13] R. J. LeVeque and Z. Li, *Immersed Interface Methods for Stokes Flow with Elastic Boundaries or Surface Tension*, submitted to SIAM J. Sci. Comput., 1995.

[14] R. J. LeVeque and C. Zhang, *Immersed interface methods for wave equations with discontinuous coefficients*. submitted.

[15] Z. Li, *The Immersed Interface Method – A Numerical Approach for Partial Differential Equations with Interfaces*, Ph. D. Thesis, University of Washington, 1994.

[16] Z. Li and A. Mayo, *ADI methods for heat equations with discontinuities along an arbitrary interface*, Proceedings of Symposia in Applied Mathematics, 48 (1994), pp. 311-315.

[17] Z. Li, *A note on immersed interface method for three dimensional elliptic equations*, submitted to Applied Mathematics Letters, 1995.

[18] R. H. Nochetto and M. Paolini and C. Verdi, *An adaptive finite element method for two-phase Stefan problems in two space dimensions*, SIAM J. Sci. Stat. Comput., 12 (1991), No. 5, pp. 1207-1244.

[19] S. Osher and J.A. Sethian, *Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations"*, J. Comput. Phys., 79 (1988), pp. 12-49.

[20] M. E. Rose, *An enthalpy scheme for Stefan problem in several dimensions*, App. Numerical Mathematics, 12 (1993), pp. 229-238.

[21] C. S. Peskin, *Numerical analysis of blood flow in the heart*, J. Comput. Phys., 25 (1977), pp. 220–252.

[22] C. S. Peskin, *Lectures on mathematical aspects of physiology*, Lectures in Appl. Math., 19 (1981), pp. 69–107.

[23] C. S. Peskin and B. F. Printz, *Improved volume conservation in the computation of flows with immersed elastic boundaries*, J. of Comput. Phys., 105 (1993), pp. 33-46.

[24] D. Sulsky and J. Brackbill, *A numerical method for suspension flow* J. of Comput. Phys., 96 (1991), pp. 339-368.

[25] edited by L. C. Wrobel and C. A. Brebbla, *Computational Modeling of Free and Moving Boundary Problems II*, Computational Mechanics Publications, 1993.

[26] A. Wiegmann K. Bube *Immersed interface method for 1D non-linear differential equations with discontinuous coefficients and singular sources*, 1995, preprints