國立臺灣大學理學院數學所

碩士論文

Department of Mathematics

College of Science

National Taiwan University

Master Thesis

界面耦合法在移動界面問題上的應用

Solving Some Moving Interface Problems by the Coupling Interface Method

劉思瀚

Ssu-Han Liu

指導教授：陳宜良 教授

Advisor：Professor I-Liang Chern

中華民國102年7月

July 2013

國立臺灣大學碩士學位論文
口試委員會審定書

界面耦合法在移動界面問題上的應用

Solving Some Moving Interface Problems by the Coupling Interface Method

本論文係劉思瀚君（R98221001）在國立臺灣大學數學所完成之碩士學位論文，於民國 102 年 7 月 19 日承下列考試委員審查通過及口試及格，特此證明

口試委員：

陳宜良 （簽名）

（指導教授）

薛克民

舒宇宸

系主任、所長 （簽名）

誌謝

能完成這篇論文，我要特別感謝我的指導教授陳宜良老師，也要感謝李曉林教授提供FronTier library以及舒宇宸教授在程式修改上的指導。最後感謝每一個家人和朋友一路上的支持與鼓勵。

中文摘要

由陳宜良教授以及舒宇宸教授所提出的界面耦合法目的在笛卡兒網格下解橢圓介面問題，此方法已被證明是處理界面問題中非常有競爭力的方法。在這篇論文, 我們把界面耦合法應用在不同問題上, 例如:一維移動界面問題、二維固定界面的熱傳導問題以及二維的融化問題等, 我們也提供數值例子來驗證方法的收斂性。

關鍵字: 界面耦合法、移動界面問題、交替方向隱式法、克蘭克－尼科爾森方法、融化問題、界面追蹤法。

Abstract

The coupling interface method (CIM) proposed by Chern and Shu aims for solving elliptic complex interface problems in arbitrary dimensions under Cartesian grid. It has been proven that the method is very competitive in dealing with interface problems. In this thesis, we apply the CIM to various problems, including one dimensional moving interface problems, two dimensional diffusion equations with fixed interface, two dimensional melting problems, etc. Numerical examples are presented to test the accuracy of the method in these applications.

Keywords: coupling interface method, moving interface problems, ADI method, Crank-Nicolson scheme, melting problems, front tracking method

CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1
Coupling Interface Method in One Dimension

In this chapter we review the coupling interface method (CIM) in one dimension under Cartesian grid for solving interface problems proposed by Chern and Shu[5]. It contains a first-order method (CIM1) and a second-order method (CIM2).

Consider the elliptic interface problem on a domain $\Omega = [a,b]$, $(\varepsilon u')' = f$ on $[a,b]$. The elliptic coefficient $\varepsilon(x) > 0$ may have jumps across some interface points on $[a,b]$. We partition $[a,b]$ into $M+1$ subintervals evenly and define

$h \triangleq \frac{b-a}{M+1}$, (step size)

$x_i \triangleq a + ih, i = 0,1,\dots,M+1.$ (grid points)

One basic assumption is that there is at most one interface point in each subinterval (which is achievable by refining meshes). A grid point x_i is called on-front if either $[x_{i-1}, x_i)$ or $[x_i, x_{i+1})$ contains an interface point. Otherwise, it is called an interior point. At an interior point x_i, we approximate $(\varepsilon u')'_i$ by a standard central finite difference scheme. Namely,

$$(\varepsilon u')'_i = \frac{\varepsilon_{i-1/2}u_{i-1} - (\varepsilon_{i-1/2} + \varepsilon_{i+1/2})u_i + \varepsilon_{i+1/2}u_{i+1}}{h^2} + O(h^2). \quad (1)$$

At an on-front grid point x_i, we discuss the CIM1 and CIM2 below. In both methods, we call the side where x_i is located the Ω^- side, and the other side the Ω^+ side (see figure 1). Given an interface point $\hat{x}$ in $[x_{i-1}, x_{i+1})$, the jump conditions at $\hat{x}$ are the following:

$$[u]_{\hat{x}} \triangleq u^{+}(\hat{x}) - u^{-}(\hat{x}) = \tau, \tag{2}$$

$$[\varepsilon u_x]_{\hat{x}} \triangleq (\varepsilon u)^{+}(\hat{x}) - (\varepsilon u)^{-}(\hat{x}) = \sigma. \tag{3}$$

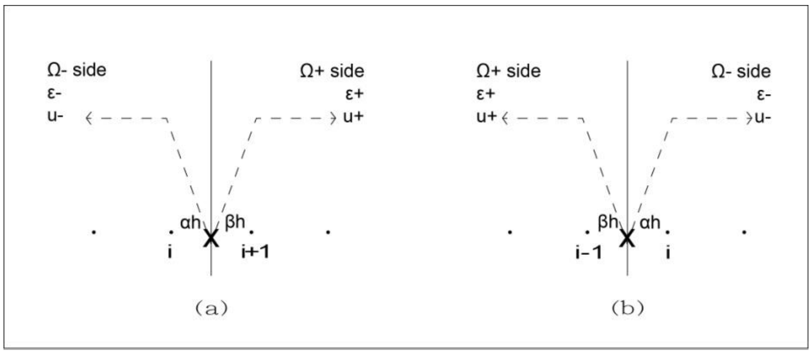

Figure 1.

CIM1

The idea is to approximate u by linear functions on both side of the interface point.

Suppose there is no interface point in $[x_i, x_{i+1})$, we approximate u_i' by

$$u_i' = \frac{u_{i+1} - u_i}{h} + O(h). \tag{4}$$

Suppose there is an interface point $\hat{x}$ in $[x_i, x_{i+1})$. First, we expand u in Taylor series about $\hat{x}$ from both side of the interface to get

$$u^{-}(x) = u_i + u_i'(x - x_i) + O(h^2), \qquad \text{for } x \in [x_i, \hat{x})$$

$$u^{+}(x) = u_{i+1} + u_{i+1}'(x - x_{i+1}) + O(h^2). \qquad \text{for } x \in (\hat{x}, x_{i+1})$$

In particular, we have

$$u^{-}(\hat{x}) = u_i + u_i'(\alpha h) + O(h^2),$$

$$u^{+}(\hat{x}) = u_{i+1} + u_{i+1}'(-\beta h) + O(h^2),$$

where $\alpha \triangleq \frac{\hat{x}-x_i}{h}$ and $\beta \triangleq 1-\alpha$.

The jump condition $[u]_{\hat{x}} = \tau$ gives

$$\tau = \left(u_{i+1} + u'_{i+1}(-\beta h)\right) - \left(u_i + u'_i(\alpha h)\right) + O(h^2)$$

$$= -(\alpha h)u'_i - (\beta h)u'_{i+1} + u_{i+1} - u_i + O(h^2). \tag{5}$$

Second, we approximate u' about $\hat{x}$ from both side of the interface to get

$$(u')^-(\hat{x}) = u'_i + O(h),$$

$$(u')^+(\hat{x}) = u'_{i+1} + O(h).$$

The jump condition $[\varepsilon u_x]_{\hat{x}} = \sigma$ gives

$$\sigma = \varepsilon^+ u'_{i+1} - \varepsilon^- u'_i + O(h). \tag{6}$$

Combining the two expressions (5) and (6), we can form a 2×2 linear system

$$A_{2\times2}\begin{bmatrix} u'_i \\ u'_{i+1} \end{bmatrix} = \begin{bmatrix} \tau \\ \sigma \end{bmatrix} - \begin{bmatrix} u_{i+1} - u_i \\ 0 \end{bmatrix} + \begin{bmatrix} O(h^2) \\ O(h) \end{bmatrix},$$

where

$$A = \begin{bmatrix} -\alpha h & -\beta h \\ -\varepsilon^- & \varepsilon^+ \end{bmatrix}.$$

Therefore, solving this linear equation, we have

$$u'_i = \frac{1}{h}\left(\bar{\rho}^+(u_{i+1} - u_i) - \bar{\rho}^+\tau - \beta h\frac{\sigma}{\bar{\varepsilon}}\right) + O(h), \tag{7}$$

$$u'_{i+1} = \frac{1}{h}\left(\bar{\rho}^-(u_{i+1} - u_i) - \bar{\rho}^-\tau + \alpha h\frac{\sigma}{\bar{\varepsilon}}\right) + O(h), \tag{8}$$

where $\bar{\varepsilon} \triangleq \alpha\varepsilon^+ + \beta\varepsilon^-$, $\bar{\rho}^\pm \triangleq \varepsilon^\pm/\bar{\varepsilon}$.

Let $\gamma_{i+1/2}$ be the number of the interface point in $[x_i, x_{i+1})$. Here, $\gamma_{i+1/2} = 0$ or 1. We can combine (4) and (7) into one formula:

$$u'_{i+1/2} = \frac{1}{h}\left(\left(1+\gamma_{i+1/2}(\bar{\rho}^{+}-1)\right)(u_{i+1}-u_i) + \gamma_{i+1/2}\left(-\bar{\rho}^{+}\tau - \beta h\frac{\sigma}{\bar{\varepsilon}}\right)\right) + O(h), \tag{9}$$

where $i + 1/2$ means that we approximate u'_i from right.

Similarly, we have

$$u'_{i-1/2} = \frac{1}{h}\left(\left(1+\gamma_{i-1/2}(\bar{\rho}^{+}-1)\right)(u_i-u_{i-1}) + \gamma_{i-1/2}\left(\bar{\rho}^{+}\tau - \beta h\frac{\sigma}{\bar{\varepsilon}}\right)\right) + O(h), \tag{10}$$

where $\alpha \triangleq \frac{x_i-\hat{x}}{h}$ (here and after we give a unified definition: $\alpha \triangleq \frac{|x_i-\hat{x}|}{h}$). Here $i - 1/2$ means that we approximate u'_i from left.

Combining (9) and (10), we get a unified formula:

$$u'_{i\pm1/2} = \frac{1}{h}\left(\left(1+\gamma_{i\pm1/2}(\bar{\rho}^{+}-1)\right)\bar{D}^{(\pm1/2)}u_i + \gamma_{i\pm1/2}\left(\mp\bar{\rho}^{+}\tau - \beta h\frac{\sigma}{\bar{\varepsilon}}\right)\right) + O(h), \tag{11}$$

where

$$\bar{D}^{(\pm1/2)}u_i \triangleq \pm u_{i\pm1} \mp u_i. \tag{12}$$

Finally, we approximate $(\varepsilon u')'_i$ by

$$(\varepsilon u')'_i = \frac{1}{h}\varepsilon_i\left(u'_{i+1/2} - u'_{i-1/2}\right) + O(1). \tag{13}$$

CIM2

Suppose there is an interface point $\hat{x}$ in $[x_i, x_{i+1})$. To derive CIM2, we further assume that there is no other interface points inside $[x_{i-1}, x_{i+2}]$. The idea is to approximate u by quadratic functions on both side of $\hat{x}$. These involve six coefficients. They are determined by the two jump conditions and realizing u at x_{i-1}, x_i, x_{i+1} and x_{i+2}. First, we expand u in Taylor series about $\hat{x}$ from both side of the interface to get

$$u^-(x) = u_i + u_i'(x - x_i) + \tfrac{1}{2}u_i''(x - x_i)^2 + O(h^3)$$

$$= u_i + \left(\frac{u_i - u_{i-1}}{h} + \frac{1}{2}hu_i''\right)(x - x_i) + \frac{1}{2}u_i''(x - x_i)^2 + O(h^3)$$

$$\text{for } x \in [x_{i-1}, \hat{x}) \qquad \left(\text{here we use } u_i' = \frac{u_i - u_{i-1}}{h} + \frac{1}{2}hu_i'' + O(h^2)\right),$$

$$u^+(x) = u_{i+1} + u_{i+1}'(x - x_{i+1}) + \tfrac{1}{2}u_{i+1}''(x - x_{i+1})^2 + O(h^3)$$

$$= u_{i+1} + \left(\frac{u_{i+2} - u_{i+1}}{h} - \frac{1}{2}hu_{i+1}''\right)(x - x_{i+1}) + \frac{1}{2}u_{i+1}''(x - x_{i+1})^2 + O(h^3)$$

$$\text{for } x \in (\hat{x}, x_{i+1}) \qquad \left(\text{here we use } u_{i+1}' = \frac{u_{i+2} - u_{i+1}}{h} - \frac{1}{2}hu_{i+1}'' + O(h^2)\right).$$

In particular, we have

$$u^-(\hat{x}) = u_i + \left(\frac{u_i - u_{i-1}}{h} + \frac{1}{2}hu_i''\right)(\alpha h) + \frac{1}{2}u_i''(\alpha h)^2 + O(h^3), \tag{14}$$

$$u^+(\hat{x}) = u_{i+1} + \left(\frac{u_{i+2} - u_{i+1}}{h} - \frac{1}{2}hu_{i+1}''\right)(-\beta h) + \frac{1}{2}u_{i+1}''(-\beta h)^2 + O(h^3). \tag{15}$$

The jump condition $[u]_{\hat{x}} = \tau$ gives

$$\tau = (u_{i+1} - u_i) + \left(\frac{u_{i+2} - u_{i+1}}{h} - \frac{1}{2}hu_{i+1}''\right)(-\beta h) - \left(\frac{u_i - u_{i-1}}{h} + \frac{1}{2}hu_i''\right)(\alpha h)$$

$$+\frac{1}{2}u_{i+1}''(-\beta h)^2 - \frac{1}{2}u_i''(\alpha h)^2 + O(h^3).$$

After some calculations we can get

$$\tau = \frac{-1}{2}h^2(\alpha + \alpha^2)u_i'' + \frac{1}{2}h^2(\beta + \beta^2)u_{i+1}'' - \beta u_{i+2} + (1 + \beta)u_{i+1} - (1 + \alpha)u_i + \alpha u_{i-1} + O(h^3). \tag{16}$$

Second, we expand u' in Taylor series about $\hat{x}$ from both side of the interface to get

$$(u')^-(x) = u_i' + u_i''(x - x_i) + O(h^2)$$

$$= \left(\frac{u_i - u_{i-1}}{h} + \frac{1}{2}hu_i''\right) + u_i''(x - x_i) + O(h^2),$$

$$(u')^+(x) = u_{i+1}' + u_{i+1}''(x - x_{i+1}) + O(h^2)$$

$$= \left(\frac{u_{i+2} - u_{i+1}}{h} - \frac{1}{2}hu_{i+1}''\right) + u_{i+1}''(x - x_{i+1}) + O(h^2).$$

In particular, we have

$$(u')^-(\hat{x}) = \left(\frac{u_i - u_{i-1}}{h} + \frac{1}{2}hu_i''\right) + u_i''(\alpha h) + O(h^2), \tag{17}$$

$$(u')^+(\hat{x}) = \left(\frac{u_{i+2} - u_{i+1}}{h} - \frac{1}{2}hu_{i+1}''\right) + u_{i+1}''(-\beta h) + O(h^2). \tag{18}$$

The jump condition $[\varepsilon u_x]_{\hat{x}} = \sigma$ gives

$$\sigma = \varepsilon^+ \left(\left(\frac{u_{i+2}-u_{i+1}}{h} - \frac{1}{2}hu_{i+1}''\right) - u_{i+1}''(\beta h)\right) - \varepsilon^- \left(\left(\frac{u_i-u_{i-1}}{h} + \frac{1}{2}hu_i''\right) + u_i''(\alpha h)\right) + O(h^2).$$

After some calculations we can get

$$\sigma = -h\varepsilon^-\left(\frac{1}{2} + \alpha\right)u_i'' - h\varepsilon^+\left(\frac{1}{2} + \beta\right)u_{i+1}'' + \varepsilon^+\left(\frac{u_{i+2} - u_{i+1}}{h}\right) - \varepsilon^-\left(\frac{u_i - u_{i-1}}{h}\right) + O(h^2). \tag{19}$$

Combining the two expressions (16) and (19), we can form a 2×2 linear system

$$A_{2\times 2}\begin{bmatrix} u_i'' \\ u_{i+1}'' \end{bmatrix} = \begin{bmatrix} \tau \\ \sigma \end{bmatrix} - \begin{bmatrix} b_{i,-1}u_{i-1} + b_{i,0}u_i + b_{i,1}u_{i+1} + b_{i,2}u_{i+2} \\ c_{i+1,-1}u_{i-1} + c_{i+1,0}u_i + c_{i+1,1}u_{i+1} + c_{i+1,2}u_{i+2} \end{bmatrix} + \begin{bmatrix} O(h^3) \\ O(h^2) \end{bmatrix},$$

where

$$\begin{cases} b_{i,-1} = \alpha, \\ b_{i,0} = -(1+\alpha), \\ b_{i,1} = (1+\beta), \\ b_{i,2} = -\beta. \end{cases}$$

$$\begin{cases} c_{i+1,-1} = \varepsilon^-/h, \\ c_{i+1,0} = -\varepsilon^-/h, \\ c_{i+1,1} = -\varepsilon^+/h, \\ c_{i+1,2} = -\varepsilon^+/h. \end{cases}$$

$$A = \begin{bmatrix} \frac{-1}{2}h^2(\alpha + \alpha^2) & \frac{1}{2}h^2(\beta + \beta^2) \\ -h\varepsilon^-\left(\frac{1}{2} + \alpha\right) & -h\varepsilon^+\left(\frac{1}{2} + \beta\right) \end{bmatrix}.$$

Therefore, solving this linear equation, we have

$$u_i'' = \frac{1}{h^2}\left(L^{(1)}u_i + J_i\right) + O(h), \tag{20}$$

$$u_{i+1}'' = \frac{1}{h^2}\left(L^{(-1)}u_{i+1} + J_{i+1}\right) + O(h), \tag{21}$$

where

$$\begin{cases} L^{(1)}u_i \triangleq a_{i,-1}u_{i-1} + a_{i,0}u_i + a_{i,1}u_{i+1} + a_{i,2}u_{i+2}, \\ L^{(-1)}u_{i+1} \triangleq a_{i+1,-1}u_{i-1} + a_{i+1,0}u_i + a_{i+1,1}u_{i+1} + a_{i+1,2}u_{i+2}. \end{cases}$$

$$\begin{cases} J_i \triangleq -(1+2\beta)\rho^+\tau - \left(\beta+\beta^2\right)\dfrac{\sigma h}{\hat{\varepsilon}}, \\ J_{i+1} \triangleq (1+2\alpha)\rho^-\tau - (\alpha+\alpha^2)\dfrac{\sigma h}{\hat{\varepsilon}}. \end{cases}$$

with the corresponding coefficients:

$$\hat{\varepsilon} \triangleq (\beta+\beta^2)\left(\frac{1}{2}+\alpha\right)\varepsilon^- + (\alpha+\alpha^2)\left(\frac{1}{2}+\beta\right)\varepsilon^+,$$

$$\rho^\pm \triangleq \frac{\varepsilon^\pm}{\hat{\varepsilon}},$$

$$\begin{cases} a_{i,-1} \triangleq (\beta+\beta^2)\rho^- + \alpha(1+2\beta)\rho^+, \\ a_{i,0} \triangleq -(\beta+\beta^2)\rho^- - (1+\alpha)(1+2\beta)\rho^+, \\ a_{i,1} \triangleq (1+\beta)^2\rho^+, \\ a_{i,2} \triangleq -\beta^2\rho^+. \end{cases}$$

$$\begin{cases} a_{i+1,-1} \triangleq -\alpha^2\rho^-, \\ a_{i+1,0} \triangleq (1+\alpha)^2\rho^-, \\ a_{i+1,1} \triangleq -(\alpha+\alpha^2)\rho^+ - (1+\beta)(1+2\alpha)\rho^-, \\ a_{i+1,2} \triangleq (\alpha+\alpha^2)\rho^+ + \beta(1+2\alpha)\rho^-. \end{cases}$$

Now, we try to unify the above finite difference formulae (20) and (21) into just one formula. At a grid point x_i, we assume x_i satisfies one of the following assumptions:

1. x_i is an interior point.
2. If there is an interface point $\hat{x}$ in $[x_i, x_{i+1})$, then there are no other interface points inside $[x_{i-1}, x_{i+2}]$.
3. If there is an interface point $\hat{x}$ in $[x_{i-1}, x_i)$, then there are no other interface points inside $[x_{i-2}, x_{i+1}]$.

Let us define an orientation indicator s to be

$$s \triangleq \begin{cases} 1 & \text{if } \hat{x} \in [x_i, x_{i+1}), \\ -1 & \text{if } \hat{x} \in [x_{i-1}, x_i), \\ 0 & x_i \text{ is an interior point.} \end{cases}$$

Define the following parameters:

$$\hat{\varepsilon} \triangleq (\beta + \beta^2)\left(\frac{1}{2} + \alpha\right)\varepsilon^- + (\alpha + \alpha^2)\left(\frac{1}{2} + \beta\right)\varepsilon^+,$$

$$\rho^\pm \triangleq \frac{\varepsilon^\pm}{\hat{\varepsilon}},$$

$$\begin{cases} a_{-s} \triangleq (\beta + \beta^2)\rho^- + \alpha(1 + 2\beta)\rho^+, \\ a_0 \triangleq -(\beta + \beta^2)\rho^- - (1 + \alpha)(1 + 2\beta)\rho^+, \\ a_s \triangleq (1 + \beta)^2\rho^+, \\ a_{2s} \triangleq -\beta^2\rho^+. \end{cases}$$

$$L^{(s)}u_i \triangleq \begin{cases} a_{-s}u_{i-s} + a_0u_i + a_su_{i+s} + a_{2s}u_{i+2s} & \text{if } s = \pm 1, \\ u_{i-1} - 2u_i + u_{i+1} & \text{if } s = 0. \end{cases} \tag{22}$$

$$J_{\hat{x}} \triangleq -\left(|s|(1 + 2\beta)\rho^+[u] + s(\beta + \beta^2)h\frac{[\varepsilon u']}{\hat{\varepsilon}}\right).$$

With these notations, we can express u_i'' as

$$u_i'' = \frac{1}{h^2}\left(L^{(s)}u_i + J_{\hat{x}}\right) + O\left(h^{2-|s|}\right). \tag{23}$$

Finally, we approximate $(\varepsilon u')_i'$ by

$$\begin{aligned} (\varepsilon u')_i' &= \varepsilon_i' u_i' + \varepsilon_i u_i'' \\ &= \frac{1}{h}D_{(s)}\varepsilon_i \frac{1}{h}D_{(s)}u_i + \varepsilon_i\left(\frac{1}{h^2}\left(L^{(s)}u_i + J_{\hat{x}}\right)\right) + O\left(h^{2-|s|}\right) \\ &= \frac{1}{h^2}\left(D_{(s)}\varepsilon_i D_{(s)}u_i + \varepsilon_i h^2\left(\frac{1}{h^2}\left(L^{(s)}u_i + J_{\hat{x}}\right)\right)\right) + O\left(h^{2-|s|}\right), \end{aligned} \tag{24}$$

where

$$D_{(s)}u_i = \begin{cases} \dfrac{u_{i+1} - u_{i-1}}{2} & \text{if } s = 0, \\ u_i - u_{i-1} & \text{if } s = 1, \\ u_{i+1} - u_i & \text{if } s = -1. \end{cases}$$

In summary, at an interior point we adopt a standard central finite difference scheme, which produces $O(h^2)$ local truncation error. At an on-front grid point, the CIM1 produces $O(1)$ local truncation error and the CIM2 produces $O(h)$ local truncation error, so the global error is $O(h)$ and $O(h^2)$, respectively. Refer to [5] for more information about the CIM (ex: stability of the coefficient matrix).

Chapter 2
Application to One Dimensional Moving Interface Problems

In this chapter we apply the coupling interface method to the one dimensional moving interface problems. The model problem we considered is the following:

$$u_t + \lambda u u_x = (\varepsilon u_x)_x - f(x,t) \qquad x \in [0,\xi) \cup (\xi,1], \tag{25}$$

$$\frac{d\xi}{dt} = w(t,\xi; u^-, u^+, u_x^-, u_x^+) \qquad t > 0. \tag{26}$$

Here, $\xi(t)$ is the trajectory of the moving interface. We denote its left hand side by Ω^- side and right hand side by Ω^+.The notation 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 hand side of $\xi(t)$. The function w represents the velocity of the interface, which is a known function of $t, \xi,$ u^-, u^+, u_x^- and u_x^+ and λ is a constant. The coefficient $\varepsilon(x,t) > 0$ and the source term $f(x,t)$ are assumed to be smooth on both side of $\xi(t)$ but may be discontinuous at $\xi(t)$. This is a parabolic problem and the solution in each domain $\big[0,\xi(t)\big)$ and $(\xi(t),1]$ is smooth. Across the interface, there are two kinds of jump conditions considered.

JC 1. Jump conditions of the form

$$[u](t) \triangleq u(\xi^+,t) - u(\xi^-,t) = \tau(t), \tag{27}$$

$$[\varepsilon u_x](t) \triangleq \varepsilon(\xi^+,t)u_x(\xi^+,t) - \varepsilon(\xi^-,t)u_x(\xi^-,t) = \sigma(t), \tag{28}$$

are given

JC 2. The solution on the interface

$$u(\xi, t) = r(t), \tag{29}$$

is given. One example is a mathematical model for solidification problems.

Numerical method

We partition $[0,1]$ into $M+1$ subintervals evenly and define

$h \triangleq \frac{1}{M+1}$, (step size in space)

$x_i \triangleq ih, i = 0,1,\dots,M+1.$ (grid points)

We use Δt as the temporal step size in time and assume the ratio $\Delta t/h$ is a constant (ex: equals to 1). Using the Crank-Nicholson scheme, the semi-discrete difference scheme for (25) can be written in the following form:

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} - C_i^n + \frac{\lambda}{2}\left(u_i^n u_{x,i}^n + u_i^{n+1} u_{x,i}^{n+1}\right) = \frac{1}{2}\left((\varepsilon u_x)_{x,i}^n + (\varepsilon u_x)_{x,i}^{n+1}\right) - \frac{1}{2}\left(f_i^n + f_i^{n+1}\right), \tag{30}$$

where $u_{x,i}^n$ and $(\varepsilon u_x)_{x,i}^n$ are u_x and $(\varepsilon u_x)_x$ at (x_i, t^n), and C_i^n is a correction term for time which meaning will be discussed later. The interface location is determined by the trapezoidal method applied to (26)

$$\frac{\xi^{n+1} - \xi^n}{\Delta t} = \frac{1}{2}(w^n + w^{n+1}), \tag{31}$$

where $w^n = w\left(t^n, \xi^n; u^{-,n}, u^{+,n}, u_x^{-,n}, u_x^{+,n}\right)$, and ξ^n, $u^{\pm,n}$, $u_x^{\pm,n}$ are $\xi(t^n)$, $u(\xi^\pm, t^n)$ and $u_x(\xi^\pm, t^n)$, respectively.

Spatial discretization

Since the discussion here doesn't concern with time, we will drop t or the superscript n for simplicity. At an interior point x_i, a standard central finite difference scheme is adopted. Namely,

$$u_{x,i} = \frac{u_{i+1} - u_{i-1}}{2h} + O(h^2),$$

$$(\varepsilon u_x)_{x,i} = \frac{\varepsilon_{i-1/2} u_{i-1} - (\varepsilon_{i-1/2} + \varepsilon_{i+1/2}) u_i + \varepsilon_{i+1/2} u_{i+1}}{h^2} + O(h^2).$$

At an on-front grid point x_i, we approximate $u_{x,i}$ by

$$u_{x,i} = \frac{1}{h} D_{(s)} u_i + O(h).$$

We study the following two kinds of jump conditions to compute $(\varepsilon u_x)_{x,i}$.

JC 1.

We use the CIM2 to get a first order approximation of $(\varepsilon u_x)_{x,i}$. As for u^-, u^+, u_x^- and u_x^+, suppose the interface point ξ is situated in $[x_i, x_{i+1})$. First, compute $u_{xx,i}$ and $u_{xx,i+1}$ by (20) and (21). Then by (14), (15), (17) and (18), we have

$$u^- = u_i + \left(\frac{u_i - u_{i-1}}{h} + \frac{1}{2} h u_{xx,i}\right)(\alpha h) + \frac{1}{2} u_{xx,i} (\alpha h)^2 + O(h^3),$$

$$u^+ = u_{i+1} + \left(\frac{u_{i+2} - u_{i+1}}{h} - \frac{1}{2} h u_{xx,i+1}\right)(-\beta h) + \frac{1}{2} u_{xx,i+1} (-\beta h)^2 + O(h^3),$$

$$u_x^- = \left(\frac{u_i - u_{i-1}}{h} + \frac{1}{2} h u_{xx,i}\right) + u_{xx,i}(\alpha h) + O(h^2),$$

$$u_x^+ = \left(\frac{u_{i+2} - u_{i+1}}{h} - \frac{1}{2} h u_{xx,i+1}\right) + u_{xx,i+1}(-\beta h) + O(h^2).$$

JC 2.

In this case, we can't directly apply the CIM2 since we don't know the value of $[\varepsilon u_x]$. So, we go back and check how we derive the CIM2. Suppose the interface point ξ is situated in $[x_i, x_{i+1})$ and recall the definition of the following two parameters

$$\alpha \triangleq \frac{\xi - x_i}{h}, \tag{32}$$

$$\beta \triangleq 1 - \alpha. \tag{33}$$

From (14) and (29), we can get

$$\begin{aligned} r = u^-(\xi) &= u_i + \left(\frac{u_i - u_{i-1}}{h} + \frac{1}{2}hu_{xx,i}\right)(\alpha h) + \frac{1}{2}u_{xx,i}(\alpha h)^2 + O(h^3) \\ &= (1+\alpha)u_i - \alpha u_{i-1} + \frac{1}{2}(\alpha + \alpha^2)h^2 u_{xx,i} + O(h^3). \end{aligned}$$

From the above equation, we can compute $u_{xx,i}$ as

$$u_{xx,i} = \frac{1}{\frac{1}{2}(\alpha+\alpha^2)h^2}(\alpha u_{i-1} - (1+\alpha)u_i + r) + O(h). \tag{34}$$

Combining (17) and (34) we can compute u_x^- as

$$\begin{aligned} u_x^- &= \frac{u_i - u_{i-1}}{h} + \left(\frac{1}{2} + \alpha\right)hu_{xx,i} + O(h^2) \\ &= \frac{u_i - u_{i-1}}{h} + \frac{1+2\alpha}{(\alpha+\alpha^2)h}(\alpha u_{i-1} - (1+\alpha)u_i + r) + O(h^2). \end{aligned} \tag{35}$$

These formulas have an easy explanation, first we give a lemma.

Lemma Given x_1, x_2, x_3, x four points with x_1, x_2, x_3 are distinct. Suppose U is a smooth function in an interval containing x_1, x_2, x_3, x, then

$$U'(x) = c_1U(x_1) + c_2U(x_2) + c_3U(x_3) + \left(c_1O(|x_1 - x|^3) + c_2O(|x_2 - x|^3) + c_3O(|x_3 - x|^3)\right),$$

$$U''(x) = d_1U(x_1) + d_2U(x_2) + d_3U(x_3) + \left(d_1O(|x_1 - x|^3) + d_2O(|x_2 - x|^3) + d_3O(|x_3 - x|^3)\right).$$

where

$$c_1 = \frac{(x - x_2) + (x - x_3)}{(x_1 - x_2)(x_1 - x_3)}\ , c_2 = \frac{(x - x_1) + (x - x_3)}{(x_2 - x_1)(x_2 - x_3)}\ , c_3 = \frac{(x - x_1) + (x - x_2)}{(x_3 - x_1)(x_3 - x_2)},$$

$$d_1 = \frac{2}{(x_1 - x_2)(x_1 - x_3)}\ , d_2 = \frac{2}{(x_2 - x_1)(x_2 - x_3)}\ , d_3 = \frac{2}{(x_3 - x_1)(x_3 - x_2)}.$$

Proof:

Expanding U in Taylor series about x_1, x_2 and x_3 respectively and comparing the coefficients, we get the desired result. □

Define

$$D^1u(x_1, x_2, x_3, x) \triangleq c_1u(x_1) + c_2u(x_2) + c_3u(x_3),$$

$$D^2u(x_1, x_2, x_3, x) \triangleq d_1u(x_1) + d_2u(x_2) + d_3u(x_3).$$

Then it's easy to find that

$$D^2u(x_{i-1}, x_i, \xi, x_i) = \frac{1}{\frac{1}{2}(\alpha + \alpha^2)h^2}(\alpha u_{i-1} - (1 + \alpha)u_i + r),$$

$$D^1u(x_{i-1}, x_i, \xi, , \xi) = \frac{u_i - u_{i-1}}{h} + \frac{1 + 2\alpha}{(\alpha + \alpha^2)h}(\alpha u_{i-1} - (1 + \alpha)u_i + r),$$

which are the same as (34) and (35). So we find out that (34) and (35) is just the Taylor expansion approximation using x_{i-1}, x_i and ξ three nearby points.

There remains one problem in computing the derivatives of u at an on-front point. If the interface $\xi(t)$ becomes closer and closer to a grid point as time changes, then the coefficients in (34) and (35) may become very large. To avoid this situation, at time level t^n , instead of using (34) and (35) to compute the derivatives, we use the following approximations:

$$u^n_{xx,i} = \alpha^n D^2u(x_{i-1}, x_i, \xi^n, x_i) + \beta^n D^2u(x_{i-2}, x_{i-1}, \xi^n, x_i) + O(h), \tag{36}$$

$u_x^{-,n} = \alpha^n D^1 u(x_{i-1}, x_i, \xi^n, \xi^n) + \beta^n D^1 u(x_{i-2}, x_{i-1}, \xi^n, \xi^n) + O(h^2),$ (37)

where α^n and β^n are (32) and (33) at time t^n, respectively. Each of the above formula is a linear combination of two approximations. By using these formulas, we still get the same order of accuracy as before and the magnitudes of the coefficients in (36) and (37) will be of order $O(1/h)$. Similarly, at x_{i+1}we use the following approximation

$u_{xx,i+1}^n = \beta^n D^2 u(\xi^n, x_{i+1}, x_{i+2}, x_{i+1}) + \alpha^n D^2 u(\xi^n, x_{i+2}, x_{i+3}, x_{i+1}) + O(h),$ (38)

$u_x^{+,n} = \beta^n D^1 u(\xi^n, x_{i+1}, x_{i+2}, \xi^n) + \alpha^n D^1 u(\xi^n, x_{i+2}, x_{i+3}, \xi^n) + O(h^2).$ (39)

For time level t^{n+1}, we still use (37) and (39) (change n to n+1) to compute $u_x^{\pm,n+1}$ (in order to compute w^{n+1}). However, in forming linear system, we don't want to pollute our tri-diagonal linear system, so in this case we use (34) with one exception:

If $|\xi^{n+1} - x_i| < h^2$, then set $u_i^{n+1} = r(t^{n+1})$ (which is also second-order accurate)

Grid crossing and the time correction term

When discretizing the time derivative term, we need to pay more attention since the interface is moving. In some situations we need to add a time correction term C_i^n discussed below.

Suppose (x_i, t^n) and (x_i, t^{n+1}) are on the same side of the interface $\xi(t)$, then

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = \frac{1}{2}\left((u_t)_i^{n+1} + (u_t)_i^n\right) + O(\Delta t^2),$$

so there is no need for any correction and we set $C_i^n = 0$. However, if the interface point

$\xi(t)$ crosses at x_i from time t^n to time t^{n+1}, say, at time $\tilde{t}$, $\xi(t) = x_i$, $t^n < \tilde{t} < t^{n+1}$ (see figure 2), then the time derivative of u may have a jump at $t = \tilde{t}$ and we need to add a correction term C_i^n. The following theorem tells us how to choose C_i^n.

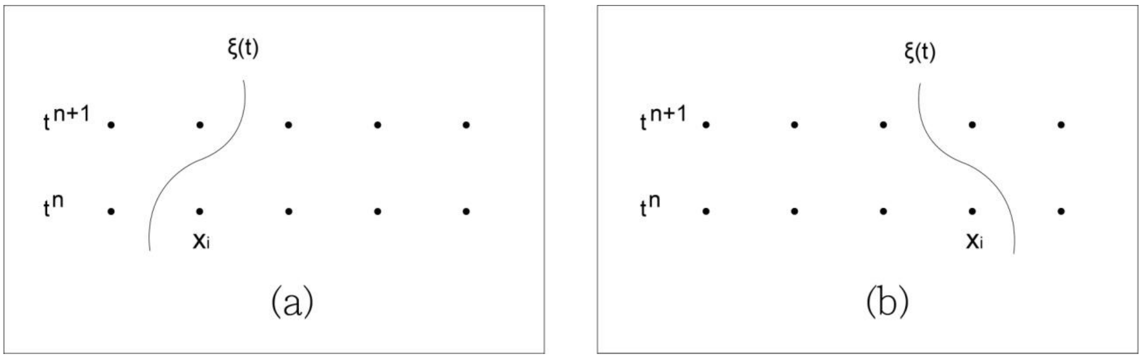

Figure 2. (a) $\xi(t)$ increases with time (b) $\xi(t)$ decreases with time

Theorem[11] Suppose the equation $\xi(t) = x_i$ has a unique solution $\tilde{t}$ in the interval (t^n, t^{n+1}). If we choose

$$C_i^n = \frac{[u]_{;\tilde{t}}}{\Delta t} + \frac{1}{\Delta t}\left(t^n + \frac{1}{2}\Delta t - \tilde{t}\right)[u_t]_{;\tilde{t}}, \tag{40}$$

then we can get a first order approximation of the time derivative of u

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} - C_i^n = \frac{1}{2}\left((u_t)_i^{n+1} + (u_t)_i^n\right) + O(\Delta t).$$

Here $[u]_{;\tilde{t}} \triangleq u(x_i, \tilde{t}^+) - u(x_i, \tilde{t}^-)$, where $u(x_i, \tilde{t}^+)$ and $u(x_i, \tilde{t}^-)$ are the limiting values of $u(x_i, \tilde{t})$ from up and down side of $\xi(t)$. Note that $[u]_{;\tilde{t}} = -[u](\tilde{t})$ for the case in figure 2(a), and $[u]_{;\tilde{t}} = [u](\tilde{t})$ for the case in figure 2(b).

Proof:

First, by Taylor expansion, we express u_i^n and u_i^{n+1} about time $\tilde{t}$ from each side of the interface to get

$$u_i^n = u(x_i,\tilde{t}^-) + (t^n - \tilde{t})u_t(x_i,\tilde{t}^-) + O(\Delta t^2), \tag{41}$$

$$u_i^{n+1} = u(x_i,\tilde{t}^+) + (t^{n+1} - \tilde{t})u_t(x_i,\tilde{t}^+) + O(\Delta t^2). \tag{42}$$

Subtracting (41) from (42), we obtain

$$\begin{aligned} u_i^{n+1} - u_i^n &= [u]_{;\tilde{t}} - \tilde{t}[u_t]_{;\tilde{t}} + t^{n+1}u_t(x_i,\tilde{t}^+) - t^n u_t(x_i,\tilde{t}^-) \\ &= [u]_{;\tilde{t}} - \tilde{t}[u_t]_{;\tilde{t}} + t^{n+1}[u_t]_{;\tilde{t}} + \Delta t u_t(x_i,\tilde{t}^-) + O(\Delta t^2). \end{aligned} \tag{43}$$

Since

$$(u_t)_i^n = u_t(x_i,\tilde{t}^-) + O(\Delta t),$$

$$\begin{aligned} (u_t)_i^{n+1} &= u_t(x_i,\tilde{t}^+) + O(\Delta t) \\ &= u_t(x_i,\tilde{t}^-) + [u_t]_{;\tilde{t}} + O(\Delta t), \end{aligned}$$

we have

$$u_t(x_i,\tilde{t}^-) = \frac{1}{2}\left((u_t)_i^n + (u_t)_i^{n+1}\right) - \frac{1}{2}[u_t]_{;\tilde{t}} + O(\Delta t). \tag{44}$$

Therefore, combining (43) and (44) gives

$$\begin{aligned} u_i^{n+1} - u_i^n &= [u]_{;\tilde{t}} - \tilde{t}[u_t]_{;\tilde{t}} + t^{n+1}[u_t]_{;\tilde{t}} + \Delta t\left(\frac{1}{2}\left((u_t)_i^n + (u_t)_i^{n+1}\right) - \frac{1}{2}[u_t]_{;\tilde{t}}\right) + O(\Delta t^2) \\ &= [u]_{;\tilde{t}} + \left(t^n + \frac{1}{2}\Delta t - \tilde{t}\right)[u_t]_{;\tilde{t}} + \Delta t\left(\frac{1}{2}\left((u_t)_i^n + (u_t)_i^{n+1}\right)\right) + O(\Delta t^2). \end{aligned} \tag{45}$$

Divide both sides of (45) by Δt

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = \frac{[u]_{;\tilde{t}}}{\Delta t} + \frac{1}{\Delta t}\left(t^n + \frac{1}{2}\Delta t - \tilde{t}\right)[u_t]_{;\tilde{t}} + \frac{1}{2}\left((u_t)_i^n + (u_t)_i^{n+1}\right) + O(\Delta t).$$

We get the desired result. □

Computing C_i^n

Next, we discuss how to compute C_i^n. The computation of $[u]_{;\tilde{t}}$ is easy.

JC 1.

(a)$\xi^n < \xi^{n+1}$ (figure 2(a))

$[u]_{;\tilde{t}} = -[u](\tilde{t}) = -\tau(\tilde{t})$.

(b)$\xi^n > \xi^{n+1}$ (figure 2(b))

$[u]_{;\tilde{t}} = [u](\tilde{t}) = \tau(\tilde{t})$.

JC 2.

$[u]_{;\tilde{t}} = 0$.

We also need to compute $\tilde{t}$ and $[u_t]_{;\tilde{t}}$. In order to get a first-order approximation of the correction term C_i^n, we need to get a second order approximation of $\tilde{t}$ and a first order approximation of $[u_t]_{;\tilde{t}}$. We follow the idea proposed by Li [11]. First we discuss how to find $\tilde{t}$.

Suppose we have already known ξ^{n+1} and w^{n+1} (we will expain why we can make this assumption later). Using the trapezoidal method twice, we get

$$\frac{\xi^{\tilde{t}} - \xi^n}{\tilde{t} - t^n} = \frac{1}{2}\left(w^n + w^{\tilde{t}}\right) + O(\Delta t^2),$$
$$\frac{\xi^{n+1} - \xi^{\tilde{t}}}{t^{n+1} - \tilde{t}} = \frac{1}{2}\left(w^{\tilde{t}} + w^{n+1}\right) + O(\Delta t^2).$$

Combining the above two equation and eliminate $w^{\tilde{t}}$, we get

$$\frac{x_i - \xi^n}{\tilde{t} - t^n} - \frac{\xi^{n+1} - x_i}{t^{n+1} - \tilde{t}} = \frac{1}{2}(w^n - w^{n+1}) + O(\Delta t^2). \tag{46}$$

Using (46), we can compute $\tilde{t}$. First, we simplify the equation.

Let $\Delta\tilde{t} \triangleq \tilde{t} - t^n$ and set

$$\begin{cases} q_1 = x_i - \xi^n, \\ q_2 = \xi^{n+1} - x_i, \\ q_3 = \frac{1}{2}(w^n - w^{n+1}), \\ q_4 = q_3\Delta t + q_2 + q_1. \end{cases}$$

With these notations, (46) becomes

$$\frac{q_1}{\Delta\tilde{t}} - \frac{q_2}{\Delta t - \Delta\tilde{t}} = q_3 + O(\Delta t^2).$$

After some arrangements, we get the following equation

$$\left(q_3 + O(\Delta t^2)\right)(\Delta\tilde{t})^2 - \left(q_4 + O(\Delta t^3)\right)\Delta\tilde{t} + q_1\Delta t = 0.$$

So we can solve this by quadratic formula

$$\Delta\tilde{t} = \frac{q_4 \pm \sqrt{(q_4)^2 - 4q_1q_3\Delta t}}{2q_3} + O(\Delta t^2). \tag{47}$$

It's very important to decide which solution is the correct one (which is not mentioned in [11]); otherwise we may get the wrong estimation of $\tilde{t}$. We separate it into four different situations.

(A) $q_3 < 0, q_1 > 0$

(I) $q_4 > 0$

In this case, since $q_3 < 0$ and $q_4 + \sqrt{(q_4)^2 - 4q_1q_3\Delta t} > 0$,

$$\frac{q_4 + \sqrt{(q_4)^2 - 4q_1q_3\Delta t}}{2q_3} < 0 \ \rightarrow |(\because \Delta\tilde{t}\ \text{should} > 0).$$

Therefore,

$$\Delta\tilde{t} = \frac{q_4 - \sqrt{(q_4)^2 - 4q_1q_3\Delta t}}{2q_3}.$$

(II) $q_4 < 0$

In this case, since $4q_1q_3\Delta t < 0$, we have $q_4 + \sqrt{(q_4)^2 - 4q_1q_3\Delta t} > 0$.

Combining this fact and $q_3 < 0$, we have

$$\frac{q_4 + \sqrt{(q_4)^2 - 4q_1q_3\Delta t}}{2q_3} < 0 \ \rightarrow | (\because \Delta\tilde{t} \text{ should} > 0).$$

Therefore,

$$\Delta\tilde{t} = \frac{q_4 - \sqrt{(q_4)^2 - 4q_1q_3\Delta t}}{2q_3}.$$

(B) $q_3 < 0, q_1 < 0$ (hence $q_2 < 0$)

$$\begin{aligned}
&\frac{q_4 - \sqrt{(q_4)^2 - 4q_1q_3\Delta t}}{2q_3} \\
&= \frac{-q_4 + \sqrt{(q_4)^2 - 4q_1q_3\Delta t}}{-2q_3} \\
&= \frac{1}{2}\Delta t + \frac{-q_1}{-2q_3} + \frac{-\ q_2}{-2q_3} + \frac{1}{-2q_3}\left(\sqrt{(q_1+q_2)^2 - 2q_1q_3\Delta t + 2q_2q_3\Delta t + (q_3\Delta t)^2}\right) \\
&\geq \frac{1}{2}\Delta t + \frac{-q_1}{-2q_3} + \frac{-q_2}{-2q_3} + \frac{1}{-2q_3}\left(\sqrt{\left(q_3\Delta t - (q_1+q_2)\right)^2}\right) \ (\because 2q_2q_3\Delta t > 0) \\
&= \frac{1}{2}\Delta t + \frac{-q_1}{-2q_3} + \frac{-q_2}{-2q_3} + \frac{1}{-2q_3}\left|\left(q_3\Delta t - (q_1+q_2)\right)\right| \\
&\geq \frac{1}{2}\Delta t + \frac{-q_1}{-2q_3} + \frac{-q_2}{-2q_3} + \frac{1}{-2q_3}\left(|q_3\Delta t| - |q_1+q_2|\right) \ \text{(triangle inequality)} \\
&= \frac{1}{2}\Delta t + \frac{-q_1}{-2q_3} + \frac{-q_2}{-2q_3} + \frac{1}{-2q_3}\left(-q_3\Delta t + (q_1+q_2)\right) \ (\because q_3 < 0 \text{ and } (q_1+q_2) < 0) \\
&= \Delta t \ \rightarrow | (\because \Delta\tilde{t} \text{ should} < \Delta t).
\end{aligned}$$

Therefore,

$$\Delta\tilde{t} = \frac{q_4 + \sqrt{(q_4)^2 - 4q_1q_3\Delta t}}{2q_3}.$$

(C) $q_3 > 0, q_1 < 0$

(I) $q_4 < 0$

Similar to (A)(I),

$$\frac{q_4 - \sqrt{(q_4)^2 - 4q_1q_3\Delta t}}{2q_3} < 0 \ \rightarrow | (\because \Delta\tilde{t} \text{ should} > 0).$$

Therefore,

$$\Delta\tilde{t} = \frac{q_4 + \sqrt{(q_4)^2 - 4q_1q_3\Delta t}}{2q_3}.$$

(II) $q_4 > 0$

Similar to (A)(II),

$$\frac{q_4 - \sqrt{(q_4)^2 - 4q_1q_3\Delta t}}{2q_3} < 0 \ \rightarrow \mid (\because \Delta\tilde{t} \text{ should} > 0).$$

Therefore,

$$\Delta\tilde{t} = \frac{q_4 + \sqrt{(q_4)^2 - 4q_1q_3\Delta t}}{2q_3}.$$

(D) $q_3 > 0, q_1 > 0$ (hence $q_2 > 0$)

$$\begin{aligned}
&\frac{q_4 + \sqrt{(q_4)^2 - 4q_1q_3\Delta t}}{2q_3} \\
&= \frac{1}{2}\Delta t + \frac{q_1}{2q_3} + \frac{q_2}{2q_3} + \frac{1}{2q_3}\left(\sqrt{(q_1+q_2)^2 - 2q_1q_3\Delta t + 2q_2q_3\Delta t + (q_3\Delta t)^2}\right) \\
&\geq \frac{1}{2}\Delta t + \frac{q_1}{2q_3} + \frac{q_2}{2q_3} + \frac{1}{2q_3}\left(\sqrt{\left(q_3\Delta t - (q_1+q_2)\right)^2}\right) \quad (\because 2q_2q_3\Delta t > 0) \\
&= \frac{1}{2}\Delta t + \frac{q_1}{2q_3} + \frac{q_2}{2q_3} + \frac{1}{2q_3}\left|\left(q_3\Delta t - (q_1+q_2)\right)\right| \\
&\geq \frac{1}{2}\Delta t + \frac{q_1}{2q_3} + \frac{q_2}{2q_3} + \frac{1}{2q_3}\left(q_3\Delta t - (q_1+q_2)\right) \\
&= \Delta t \ \rightarrow \mid (\because \Delta\tilde{t} \text{ should} < \Delta t).
\end{aligned}$$

Therefore,

$$\Delta\tilde{t} = \frac{q_4 - \sqrt{(q_4)^2 - 4q_1q_3\Delta t}}{2q_3}. \qquad \square$$

Next, we discuss how to compute $[u_t]_{;\tilde{t}}$.

JC 1.

First, we differentiate the jump condition $u(\xi^+, t) - u(\xi^-, t) = \tau(t)$ to get

$$u_x(\xi^+,t)\frac{d\alpha}{dt}+u_t(\xi^+,t)-\left(u_x(\xi^-,t)\frac{d\alpha}{dt}+u_t(\xi^-,t)\right)=\frac{d\tau}{dt}.$$

From this we can find that

$$[u_t]=\frac{d\tau}{dt}-[u_x]w. \tag{48}$$

(a) $\xi^n < \xi^{n+1}$

By Taylor expansion, we have

$$u(x_j{}^+,\tilde{t}) = u_j^n + O(\Delta t),$$

$$u(x_j{}^-,\tilde{t}) = u_j^{n+1} + O(\Delta t),$$

$$u_x(x_j{}^+,\tilde{t}) = u_{x,j}^n + O(\Delta t),$$

$$u_x(x_j{}^-,\tilde{t}) = u_{x,j}^{n+1} + O(\Delta t).$$

We use these quantities to approximate $[u_x]w$

$$[u_x]w \approx \left(u_{x,j}^n - u_{x,j}^{n+1}\right)w\left(\tilde{t}, u_j^{n+1}, u_j^n, u_{x,j}^{n+1}, u_{x,j}^n\right).$$

Combining the above approximation and (48), we can estimate $[u_t]_{;\tilde{t}}$

$$\begin{aligned}[u_t]_{;\tilde{t}} &= -[u_t](\tilde{t})\\ &\approx -\frac{d\tau}{dt}(\tilde{t}) + \left(u_{x,j}^n - u_{x,j}^{n+1}\right)w\left(\tilde{t}, u_j^{n+1}, u_j^n, u_{x,j}^{n+1}, u_{x,j}^n\right)\\ &= -\frac{d\tau}{dt}(\tilde{t}) - \left(u_{x,j}^{n+1} - u_{x,j}^n\right)w\left(\tilde{t}, u_j^{n+1}, u_j^n, u_{x,j}^{n+1}, u_{x,j}^n\right).\end{aligned} \tag{49}$$

(b) $\xi^n > \xi^{n+1}$

Similarly, we can approximate $[u_x]w$ by

$$[u_x]w \approx \left(u_{x,j}^{n+1} - u_{x,j}^n\right)w\left(\tilde{t}, u_j^n, u_j^{n+1}, u_{x,j}^n, u_{x,j}^{n+1}\right),$$

and get an estimation of $[u_t]_{;\tilde{t}}$

$$[u_t]_{;\tilde{t}} = [u_t](\tilde{t}) \approx \frac{d\tau}{dt}(\tilde{t}) - \left(u_{x,j}^{n+1} - u_{x,j}^{n}\right) w\left(\tilde{t}, u_j^n, u_j^{n+1}, u_{x,j}^n, u_{x,j}^{n+1}\right), \quad (50)$$

JC 2.

In this case, with the knowledge of u_i^n, u_i^{n+1} and the solution on the interface, we can get a simple estimation of $[u_t]_{;\tilde{t}}$

$$[u_t]_{;\tilde{t}} = \frac{u_i^{n+1} - r(\tilde{t})}{t^{n+1} - \tilde{t}} - \frac{r(\tilde{t}) - u_i^n}{\tilde{t} - t^n} + O(\Delta t). \quad (51)$$

Note that the discussion above is still valid even if the interface crosses several grid points during one time step. However, if we choose smaller time step (ex：$\Delta t = h/2$) so that the interface crosses only one grid point during one time step, we will get a smaller error constant.

Time step restriction

Since we use the Crank Nicholson scheme, we don't worry about the time step restriction from finite difference discretization. However there is a restriction from moving interface based on the classical stability theory:

$$\Delta t \le \min\left(h, \left|\frac{1}{\partial w/\partial \xi}\right|\right).$$

Since

$$\frac{\partial w}{\partial \xi} = \frac{\partial w}{\partial t}/w,$$

we have

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

or in discrete form

$$\Delta t^{n+1} = \min\left(h, \left|\frac{\Delta t^n}{w^{n+1} - w^n} w^n\right|\right).$$

Numerical algorithm

Since we need to know the information at time level t^{n+1} in advance in order to compute the time correction term, we use the predict-correct approach to solve this problem. The local truncation errors are $O(h^2)$ at most grid points, but $O(h)$ at two on front grid points and those grid points where the interface crosses. So the global error is second order accurate at all grid points. Below is the detailed process.

Algorithm 1

Choose a tolerance δ (the meaning of δ will be clear soon). Suppose we have obtained all necessary quantities at the time t^n, in other word, we have computed u_i^n, ξ^n and Δt^n.

1. Find i_0 such that $x_{i_0} \le \xi^n < x_{i_0+1}$.

2. Compute $(\varepsilon u_x)_{x,i}^n$ and $u_{x,i}^n$.

3. Compute $u^{-,n}, u^{+,n}, u_x^{-,n}, u_x^{+,n}$.

   Set $\xi_1^{n+1} = \xi^n + \Delta t^n w\left(t^n, \xi^n; u^{-,n}, u^{+,n}, u_x^{-,n}, u_x^{+,n}\right)$,

   $u_{i,1}^{n+1} = u_i^n$ (initial guess).

4. Find i_1 such that $x_{i_1} \le \xi_1^{n+1} < x_{i_1+1}$.

5. Compute $C_{i,1}^n$ using $u_i^n, u_{i,1}^{n+1}, \xi^n, \xi_1^{n+1}$

$$\begin{cases} \text{If } i_0 < i_1 \text{ compute } C_{i,1}^n \text{ for } i = i_0 + 1, \dots, i_1, \\ \text{If } i_0 > i_1 \text{ compute } C_{i,1}^n \text{ for } i = i_1 + 1, \dots, i_0, \\ \text{If } i_0 = i_1 \text{ There is no need to comput } C_{i,1}^n. \end{cases}$$

6. Use the following difference scheme

$$\frac{u_i^{n+1}-u_i^n}{\Delta t} - C_{i,1}^n + \frac{\lambda}{2}\left(u_i^n u_{x,i}^n + u_{i,1}^{n+1} u_{x,i}^{n+1}\right) = \frac{1}{2}\left((\varepsilon u_x)_{x,i}^n + (\varepsilon u_x)_{x,i}^{n+1}\right) - \frac{1}{2}\left(f_i^n + f_i^{n+1}\right),$$

to form a linear system and solve for u_i^{n+1}

7. Set $u_{i,2}^{n+1} = u_i^{n+1}$.

8. Compute $u_{i,2}^{-,n+1}, u_{i,2}^{+,n+1}, u_{x,2}^{-,n+1}, u_{x,2}^{+,n}$ (However, we still use to ξ_1^{n+1} compute these quantities). Set

$$\xi_2^{n+1} = \xi^n + \frac{\Delta t^n}{2} w\left(t^n, \xi^n; u^{-,n}, u^{+,n}, u_x^{-,n}, u_x^{+,n}\right) + \frac{\Delta t^n}{2} w\left(t^{n+1}, \xi_1^{n+1}; u_{i,2}^{-,n+1}, u_{i,2}^{+,n+1}, u_{x,2}^{-,n+1}, u_{x,2}^{+,n}\right).$$

9. If $|\xi_2^{n+1} - \xi_1^{n+1}| > \delta$

set $\xi_1^{n+1} = \xi_2^{n+1}$,

$u_{i,1}^{n+1} = u_{i,2}^{n+1}$.

Go back to 4.

else if $|\xi_2^{n+1} - \xi_1^{n+1}| < \delta$

Accept $u_{i,2}^{n+1}$ and ξ_2^{n+1} as approximations at time level t^{n+1}.

Set $u_i^{n+1} = u_{i,2}^{n+1}$,

$\xi^{n+1} = \xi_2^{n+1}$,

$$\Delta t^{n+1} = \min\left(h, \left|\frac{\Delta t^n}{w^{n+1} - w^n} w^n\right|\right).$$

Go to next time level.

Numerical examples

JC 1.

Example 1.

$$u_t = (\varepsilon u_x)_x \qquad x \in [0,\xi) \cup (\xi,1],$$

$$\frac{d\xi}{dt} = \frac{(\varepsilon_1 w_1^2 - \varepsilon_2 w_2^2)u(\xi,t)}{u_x(\xi^-,t) - u_x(\xi^+,t)} \qquad t > 0,$$

with

$$\varepsilon = \begin{cases} \varepsilon_1 & \text{if } x \le \xi(t) \\ \varepsilon_2 & \text{if } x \ge \xi(t) \end{cases} \text{ for some choice of } \varepsilon_1 \text{ and } \varepsilon_2,$$

$$w_1 = \frac{5\pi}{4}, w_2 = \frac{7\pi}{4}.$$

The jump conditions used are

$$[u] = 0,$$
$$[\varepsilon u_x](\xi,t) = -\varepsilon_2 w_2 \cos(w_2 - w_2\xi)\exp(-\varepsilon_2 w_2^2 t) - \varepsilon_1 w_1 \cos(w_1\xi)\exp(-\varepsilon_1 w_1^2 t).$$

The exact solution is

$$u(x,t) = \begin{cases} \sin(w_1 x)\exp(-\varepsilon_1 w_1^2 t) & x \le \xi(t), \\ \sin(w_2 - w_2 x)\exp(-\varepsilon_2 w_2^2 t) & x > \xi(t). \end{cases}$$

Since we assume the solution u is continuous across interface $\xi(t)$, the interface $\xi(t)$ can be determined from the scalar equation:

$$\sin(w_1\xi)\exp(-\varepsilon_1 w_1^2 t) = \sin(w_2 - w_2\xi)\exp(-\varepsilon_2 w_2^2 t).$$

Figure 3 shows how the interface crosses the grid as $(\varepsilon_1, \varepsilon_2)$ change.

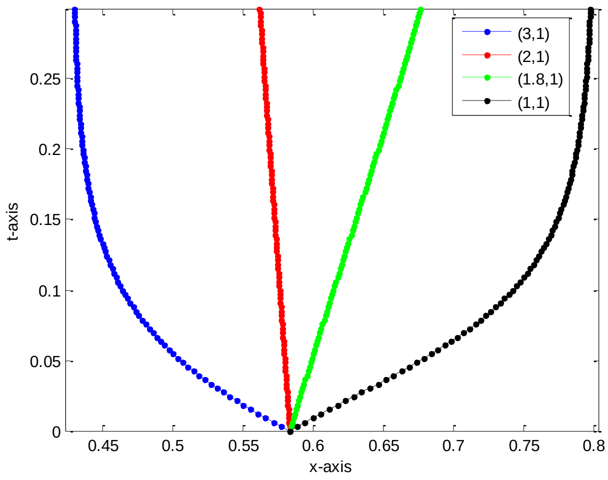

Figure 3. Moving interface $\xi(t)$, $0 \le t \le 0.3$, from left to right, $(\varepsilon_1, \varepsilon_2)$ =(3,1),(2,1),(1.8,1),(1,1).

Figure 4 shows the computed solution for $\mathrm{M} = 80$. Figure 5 shows the corresponding absolute error plot for $\mathrm{M} = 80$ (recall that M is the number of the grid points). We see that the error in the solution in u is relatively large around the interface compared to other grid points.

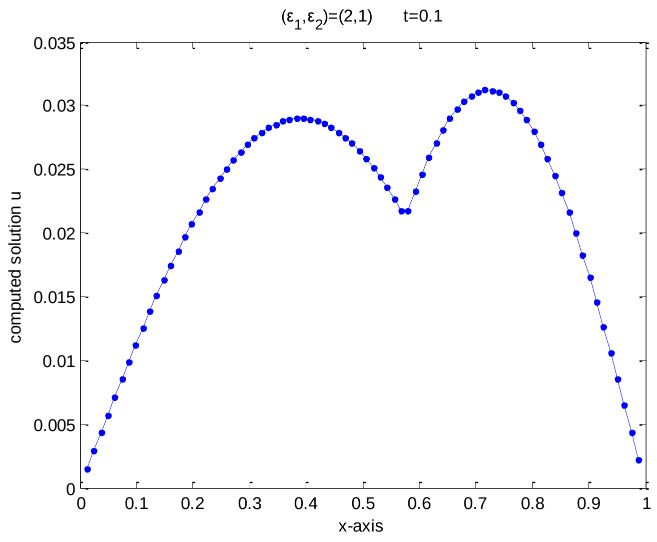

Figure4. The computed solution $\mathrm{u}(\mathrm{x},\mathrm{t})$ at $\mathrm{t} = 0.1$ with $(\varepsilon_1, \varepsilon_2)$ =(2,1) and $\mathrm{M} = 80$.

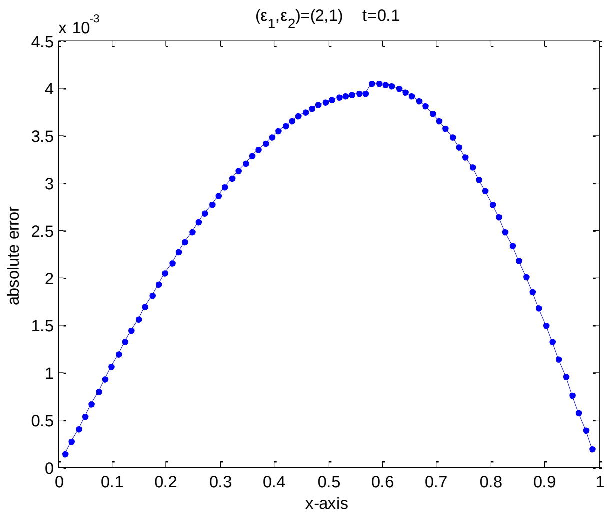

Figure 5. The absolute error at t=0.1 with $(\varepsilon_1, \varepsilon_2) =$ (2,1) and $\mathrm{M} = 80$.

Table 1 shows the results of the grid refinement analysis, where $|\mathrm{E}_\xi|$ is the error between $\xi(\mathrm{t})$ and the computed interface at the final time t.

Table 1. Numerical results for example 1.

t=0.1								
	$(\varepsilon_1,\varepsilon_2) = (3,1)$				$(\varepsilon_1,\varepsilon_2) = (2,1)$			
M	max-norm error	order	$\lvert E_\xi \rvert$	order	max-norm error	order	$\lvert E_\xi \rvert$	order
40	3.029×10^{-2}		5.342×10^{-2}		1.359×10^{-2}		1.486×10^{-3}	
80	7.131×10^{-3}	2.087	1.503×10^{-2}	1.830	4.048×10^{-3}	1.747	4.578×10^{-4}	1.699
160	1.899×10^{-3}	1.909	3.348×10^{-3}	2.167	1.092×10^{-3}	1.890	1.006×10^{-4}	2.186
320	4.942×10^{-4}	1.942	7.970×10^{-4}	2.071	2.818×10^{-4}	1.954	2.343×10^{-5}	2.102
640	1.257×10^{-4}	1.977	1.941×10^{-4}	2.038	7.178×10^{-5}	1.973	5.668×10^{-6}	2.042
1280	3.168×10^{-5}	1.988	4.800×10^{-5}	2.016	1.809×10^{-5}	1.988	1.391×10^{-6}	2.032
t=0.1								
	$(\varepsilon_1,\varepsilon_2) = (1.8,1)$				$(\varepsilon_1,\varepsilon_2) = (1,1)$			
M	max-norm error	order	$\lvert E_\xi \rvert$	order	max-norm error	order	$\lvert E_\xi \rvert$	order
40	1.468×10^{-2}		6.283×10^{-3}		1.801×10^{-2}		1.703×10^{-2}	
80	3.970×10^{-3}	1.887	1.624×10^{-3}	1.952	4.387×10^{-3}	2.038	3.633×10^{-3}	2.229
160	1.059×10^{-3}	1.906	3.543×10^{-4}	2.197	1.124×10^{-3}	1.965	8.665×10^{-4}	2.068
320	2.722×10^{-4}	1.960	8.248×10^{-5}	2.103	2.884×10^{-4}	1.963	2.126×10^{-4}	2.027
640	6.910×10^{-5}	1.978	1.990×10^{-5}	2.051	7.391×10^{-5}	1.964	5.310×10^{-5}	2.001
1280	1.741×10^{-5}	1.989	4.887×10^{-6}	2.785	1.842×10^{-5}	2.005	1.307×10^{-5}	2.023

As you can see, the faster the interface moves, the larger the $\lvert E_\xi \rvert$ becomes. In fact, for $(\varepsilon_1,\varepsilon_2)$ =(1,1) and $(\varepsilon_1,\varepsilon_2)$ =(3,1), the interface crosses two grid points during the first few time steps.

Example 2.[11](nonlinear case)

$$u_t + uu_x = (\varepsilon u_x)_x - f(x,t) \qquad x \in [0,\xi) \cup (\xi,1],$$

$$\frac{d\xi}{dt} = \frac{(\varepsilon_1 w_1^2 - \varepsilon_2 w_2^2)u(\xi,t)}{u_x(\xi^-,t) - u_x(\xi^+,t)} \qquad t > 0,$$

with

$$f(x,t) = \begin{cases} -\frac{1}{2} w_1 \sin(2w_1 x)\exp(-2\varepsilon_1 w_1^2 t) & x \le \xi(t), \\ -\frac{1}{2} w_2 \sin(2w_2 - 2w_2 x)\exp(-2\varepsilon_2 w_2^2 t) & x \ge \xi(t). \end{cases}$$

$\varepsilon = \begin{cases} \varepsilon_1 & \text{if } x \le \xi(t) \\ \varepsilon_2 & \text{if } x \ge \xi(t) \end{cases}$ for some choice of ε_1 and ε_2 , $w_1 = \frac{5\pi}{4}$, $w_2 = \frac{7\pi}{4}$.

The jump conditions used are

$[u] = 0$, $[\varepsilon u_x](\xi, t) = -\varepsilon_2 w_2 \cos(w_2 - w_2\xi)\exp(-\varepsilon_2 w_2^2 t) - \varepsilon_1 w_1 \cos(w_1\xi)\exp(-\varepsilon_1 w_1^2 t)$.

The exact solution is

$$u(x,t) = \begin{cases} \sin(w_1 x)\exp(-\varepsilon_1 w_1^2 t) & x \le \xi(t), \\ \sin(w_2 - w_2 x)\exp(-\varepsilon_2 w_2^2 t) & x > \xi(t). \end{cases}$$

Since we assume the solution u is continuous across interface $\xi(t)$, the interface $\xi(t)$ can be determined from the scalar equation:

$$\sin(w_1\xi)\exp(-\varepsilon_1 w_1^2 t) = \sin(w_2 - w_2\xi)\exp(-\varepsilon_2 w_2^2 t).$$

Basically, this example is similar to example 1, except that we add a nonlinear term and a forcing term.

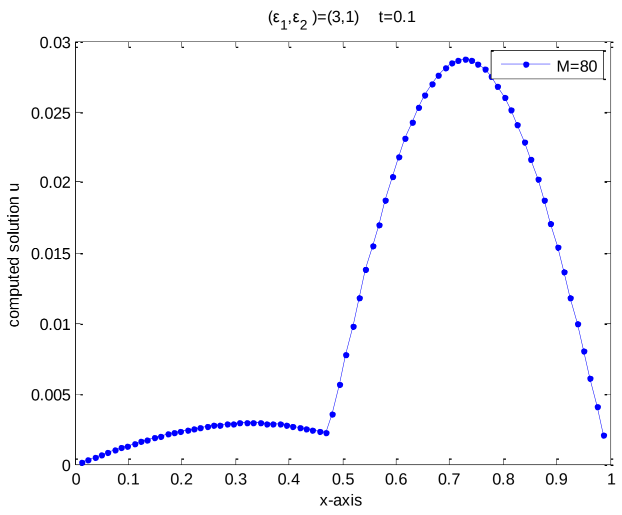

Figure 6. The computed solution $u(x,t)$ at $t = 0.1$ with $(\varepsilon_1, \varepsilon_2) = (3,1)$ and $M = 80$.

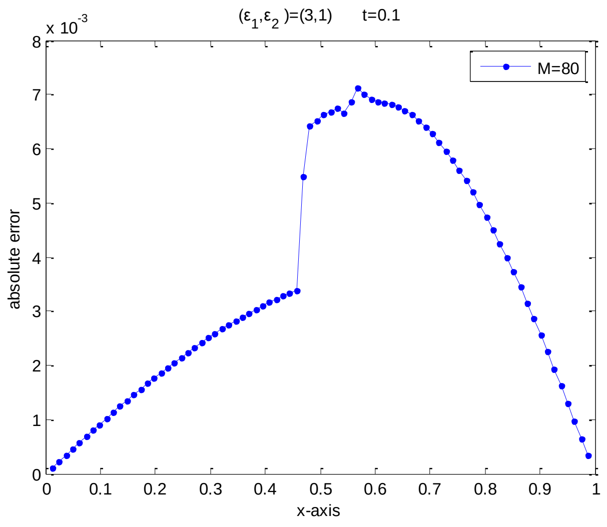

Figure 7. The absolute error at $t = 0.1$ with $(\varepsilon_1, \varepsilon_2)$ =(3,1) an $M = 80$.

Table 2. Numerical results for example 2.

t=0.1								
	$(\varepsilon_1, \varepsilon_2) = (3,1)$				$(\varepsilon_1, \varepsilon_2) = (2,1)$			
M	max-norm error	order	$\lvert E_\xi \rvert$	order	max-norm error	order	$\lvert E_\xi \rvert$	order
40	3.027×10^{-2}		5.340×10^{-2}		1.355×10^{-2}		1.487×10^{-3}	
80	7.130×10^{-3}	2.086	1.497×10^{-2}	1.835	4.030×10^{-3}	1.749	4.547×10^{-4}	1.709
160	1.899×10^{-3}	1.909	3.333×10^{-3}	2.167	1.092×10^{-3}	1.884	1.003×10^{-4}	2.181
320	4.942×10^{-4}	1.942	7.934×10^{-4}	2.071	2.812×10^{-4}	1.957	2.330×10^{-5}	2.106
640	1.257×10^{-4}	1.975	1.933×10^{-4}	2.037	7.169×10^{-5}	1.972	5.644×10^{-6}	2.046
1280	3.167×10^{-5}	1.989	4.779×10^{-5}	2.016	1.806×10^{-5}	1.989	1.385×10^{-6}	2.027
t=0.1								
	$(\varepsilon_1, \varepsilon_2) = (1.8,1)$				$(\varepsilon_1, \varepsilon_2) = (1,1)$			
M	max-norm error	order	$\lvert E_\xi \rvert$	order	max-norm error	order	$\lvert E_\xi \rvert$	order
40	1.464×10^{-2}		6.305×10^{-3}		1.792×10^{-2}		1.710×10^{-2}	
80	3.954×10^{-3}	1.889	1.616×10^{-3}	1.964	4.385×10^{-3}	2.031	3.637×10^{-3}	2.233
160	1.056×10^{-3}	1.905	3.528×10^{-4}	2.196	1.123×10^{-3}	1.965	8.677×10^{-4}	2.068
320	2.716×10^{-4}	1.959	8.214×10^{-5}	2.103	2.889×10^{-4}	1.959	2.128×10^{-4}	2.028
640	6.897×10^{-5}	1.977	1.982×10^{-5}	2.051	7.412×10^{-5}	1.963	5.314×10^{-5}	2.002
1280	1.738×10^{-5}	1.989	4.868×10^{-6}	2.026	1.846×10^{-5}	2.006	1.308×10^{-5}	2.022

JC 2.

Example 1.[11]

This is a classical Stefan problem of tracking a freezing front of ice and water.

$$u_t = \frac{k}{C} u_{xx} \qquad\qquad x \in [0, \xi) \cup (\xi, 1],$$

$$\frac{d\xi}{dt} = \frac{k_1}{\sigma} u_x(\xi^-, t) - \frac{k_2}{\sigma} u_x(\xi^+, t) \qquad t > t_0 = 0.5.$$

Here, $C = c\rho$ and $\sigma = L\rho$, where k is the heat conductivity, c is the specific heat , ρ is the density and L is the latent heat. The following thermal properties are used:

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

and

$$k = \begin{cases} k_1 & \text{if } x \le \xi(t), \\ k_2 & \text{if } x \ge \xi(t), \end{cases}$$

$$C = \begin{cases} C_1 & \text{if } x \le \xi(t), \\ C_2 & \text{if } x \ge \xi(t), \end{cases}$$

The interface condition used is

$$u^+ = u^- = 0.$$

The exact solution is

$$u(x,t) = \begin{cases} u^* \left\{1 - \dfrac{\operatorname{erf}\,(x/2\sqrt{\kappa_1 t})}{\operatorname{erf}\phi}\right\} & x \le \xi(t), \\ u_0 \left\{1 - \dfrac{\operatorname{erfc}\,(x/2\sqrt{\kappa_2 t})}{\operatorname{erfc}(\phi\sqrt{\kappa_1/\kappa_2})}\right\} & x > \xi(t). \end{cases}$$

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

where

$u^* = -20$, $u_0 = 10$, $\kappa_i = \frac{k_i}{C_i}$, erf(x) is the error function and ϕ is the root of the equation:

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

The value of ϕ is 0.2054269…

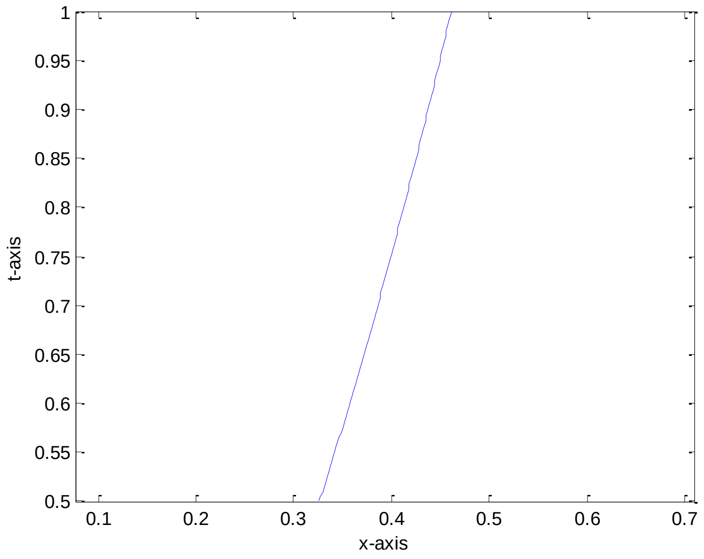

Figure 8. Moving interface $\xi(t)$, $0.5 \le t \le 1$.

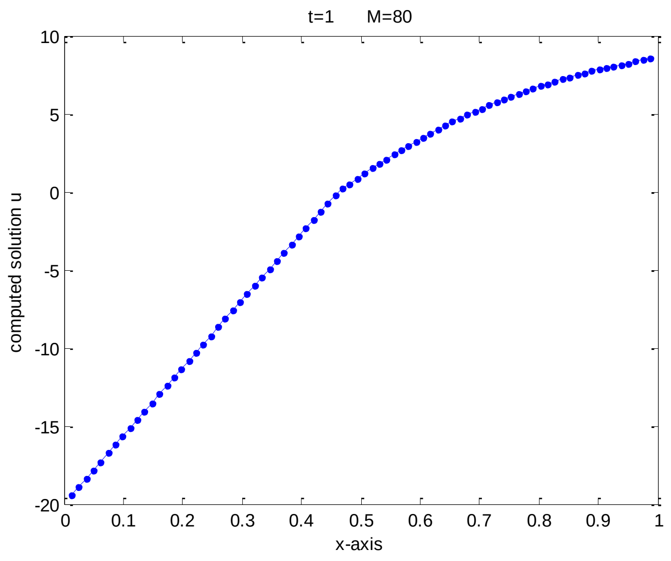

Figure 9. The computed solution $u(x,t)$ at $t = 1$ with $M = 80$.

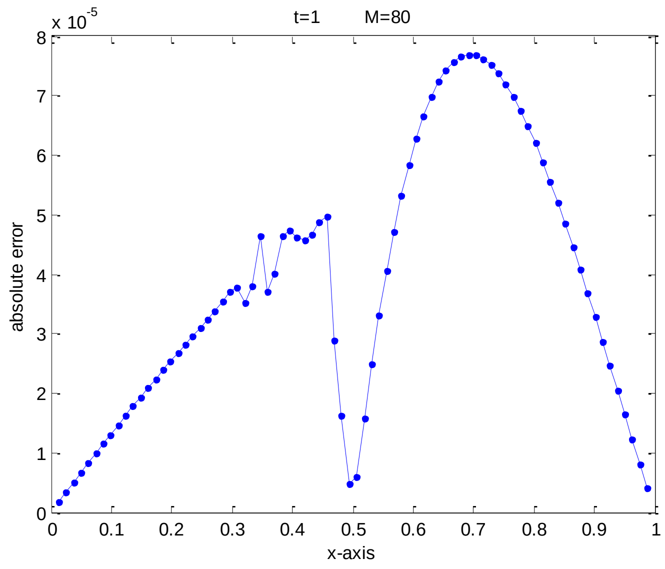

Figure 10. The absolute error at $t = 1$ and $M = 80$.

Table 3. Numerical results for example 1.

	t=1			
M	max-norm error	order	$\lvert E_\xi \rvert$	order
40	2.992×10^{-4}		4.632×10^{-6}	
80	7.688×10^{-5}	1.960	1.192×10^{-6}	1.958
160	1.955×10^{-5}	1.975	3.302×10^{-7}	1.852
320	4.949×10^{-6}	1.982	8.549×10^{-8}	1.950
640	1.235×10^{-6}	2.003	2.187×10^{-8}	1.967
1280	3.110×10^{-7}	1.990	5.318×10^{-9}	2.040

Example 2.

We retest example 1 in JC 1 and use the interface condition:

$u(\xi, t) = \sin(w_1 \xi)\exp(-\varepsilon_1 w_1^2 t)$.

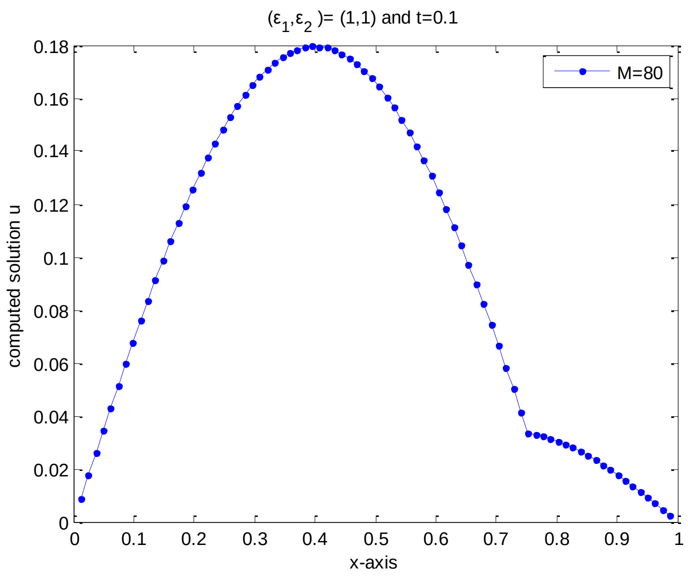

Figure 11. The computed solution $u(x, t)$ at $t = 0.1$ with $(\varepsilon_1, \varepsilon_2) = (1,1)$ and $M = 80$.

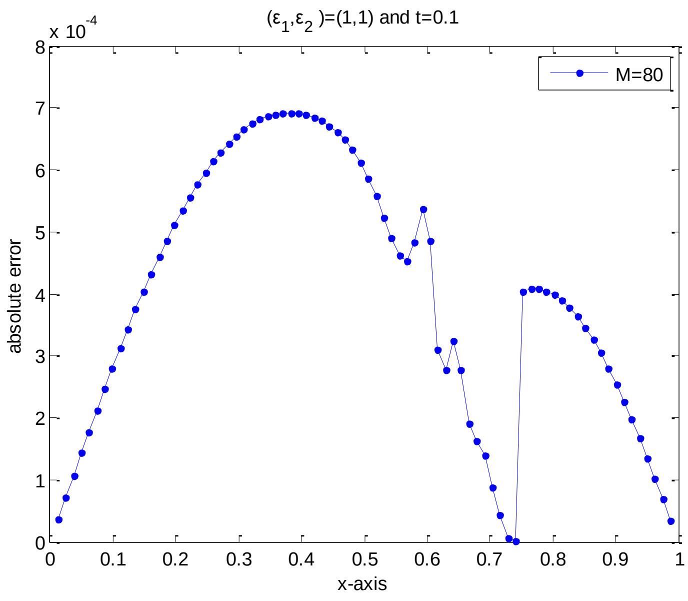

Figure 12. The absolute error at $t = 0.1$ with $(\varepsilon_1, \varepsilon_2) = (1,1)$ and $M = 80$.

Table 4. Numerical results for example 2.

t=0.1								
	$(\varepsilon_1, \varepsilon_2) = (3,1)$				$(\varepsilon_1, \varepsilon_2) = (2,1)$			
M	max-norm error	order	$\lvert E_\xi \rvert$	order	max-norm error	order	$\lvert E_\xi \rvert$	order
40	4.251×10^{-3}		2.551×10^{-2}		1.057×10^{-3}		1.063×10^{-3}	
80	5.567×10^{-4}	2.933	3.618×10^{-3}	2.818	4.176×10^{-4}	1.340	1.954×10^{-4}	2.446
160	1.903×10^{-4}	1.549	8.516×10^{-4}	2.087	1.249×10^{-4}	1.741	4.466×10^{-5}	2.129
320	5.114×10^{-5}	1.896	2.062×10^{-4}	2.046	3.412×10^{-5}	1.872	1.074×10^{-5}	2.056
640	1.366×10^{-5}	1.905	5.287×10^{-5}	1.964	8.908×10^{-6}	1.937	2.645×10^{-6}	2.022
1280	3.498×10^{-6}	1.965	1.317×10^{-5}	2.005	2.276×10^{-6}	1.969	6.546×10^{-7}	2.015
t=0.1								
	$(\varepsilon_1, \varepsilon_2) = (1.8,1)$				$(\varepsilon_1, \varepsilon_2) = (1,1)$			
M	max-norm error	order	$\lvert E_\xi \rvert$	order	max-norm error	order	$\lvert E_\xi \rvert$	order
40	1.552×10^{-3}		3.531×10^{-3}		2.609×10^{-3}		2.404×10^{-3}	
80	5.476×10^{-4}	1.503	7.088×10^{-4}	2.317	6.923×10^{-4}	1.914	6.150×10^{-4}	1.967
160	1.582×10^{-4}	1.791	1.606×10^{-4}	2.142	1.813×10^{-4}	1.933	1.643×10^{-4}	1.904
320	4.253×10^{-5}	1.895	3.893×10^{-5}	2.045	4.625×10^{-5}	1.971	4.119×10^{-5}	1.996
640	1.102×10^{-5}	1.948	9.542×10^{-6}	2.029	1.168×10^{-5}	1.985	1.023×10^{-5}	2.010
1280	2.806×10^{-6}	1.974	2.367×10^{-6}	2.011	2.936×10^{-6}	1.992	2.583×10^{-6}	1.986

Table 5 shows the results of the grid refinement analysis using (34) and (35) (instead of using (36) and (37)) to compute the derivatives. As you can see, some Instability occurs.

Table 5. Numerical results for example 2

t=0.1								
	$(\varepsilon_1, \varepsilon_2) = (3,1)$				$(\varepsilon_1, \varepsilon_2) = (2,1)$			
M	max-norm error	order	$\lvert E_\xi \rvert$	order	max-norm error	order	$\lvert E_\xi \rvert$	order
40	ERROR		ERROR		1.041×10^{-3}		1.426×10^{-3}	
80	6.229×10^{-4}		3.509×10^{-3}		4.159×10^{-4}	1.324	2.055×10^{-4}	2.795
160	1.943×10^{-4}	1.681	8.564×10^{-4}	2.035	1.247×10^{-4}	1.738	4.548×10^{-5}	2.176
320	5.176×10^{-5}	1.908	2.086×10^{-4}	2.038	3.406×10^{-5}	1.872	1.093×10^{-5}	2.057
640	1.396×10^{-5}	1.891	5.373×10^{-5}	1.957	8.893×10^{-6}	1.937	2.705×10^{-6}	2.015
1280	3.566×10^{-6}	1.967	1.341×10^{-5}	2.002	2.272×10^{-6}	1.969	6.693×10^{-7}	2.015

t=0.1								
	$(\varepsilon_1,\varepsilon_2)=(1.8,1)$				$(\varepsilon_1,\varepsilon_2)=(1,1)$			
M	max-norm error	order	$\lvert E_\xi \rvert$	order	max-norm error	order	$\lvert E_\xi \rvert$	order
40	1.565×10^{-3}		3.585×10^{-3}		ERROR		ERROR	
80	5.472×10^{-4}	1.516	7.236×10^{-4}	2.309	ERROR		ERROR	
160	1.581×10^{-4}	1.791	1.623×10^{-4}	2.157	1.783×10^{-4}		1.713×10^{-4}	
320	4.252×10^{-5}	1.895	3.973×10^{-5}	2.030	ERROR		ERROR	
640	1.102×10^{-5}	1.948	9.771×10^{-6}	2.024	1.164×10^{-5}		1.068×10^{-5}	
1280	2.805×10^{-6}	1.974	2.426×10^{-6}	2.010	2.931×10^{-6}	1.990	2.702×10^{-6}	1.983

Modification:

We use the predict-correct approach to solve the moving interface problems. The numerical results confirm that our method converges to the exact solution with second-order accuracy. However, this method has two defects when we want to generalize it into higher dimensional moving interface problems.

1. it's time consuming to solve an implicit system iteratively in each time step.
2. it's hard to generalize the idea of (46) of finding grid-crossing time $\tilde{t}$ to higher dimensional moving interface problems.

In order to improve the disadvantages mentioned above, we make the following modifications. First, assume the equation is linear. i.e. $\lambda = 0$.

1. Instead of using predict-correct approach, we use the Adams-Bashforth scheme to approximate the interface location. That is,

$$\xi^{n+1} = \xi^n + \frac{\Delta t}{2}(3w^n - w^{n-1}), \text{with } w^{-1} \triangleq w^0. \tag{52}$$

2. Instead of using (47) to approximate the grid-crossing time $\tilde{t}$, we use the following

simpler method. Suppose there is a grid crossing at x_i from time t^n to time t^{n+1} at time $\tilde{t}$, then by Taylor expansion we have

$$x_i = \xi(\tilde{t}) = \xi^n + w^n(\tilde{t} - t^n) + O(\Delta t^2).$$

So

$$\tilde{t} = t^n + \frac{x_i - \xi^n}{w^n} + O(\Delta t^2). \tag{53}$$

3. Instead of using (49) and (50) (in JC 1) or (51) (in JC 2) to approximate $[u_t]_{;\tilde{t}}$, we use the following modified method.

From (48), we have

$$[u_t] = \frac{d\tau}{dt} - [u_x]w.$$

So

$$[u_t]_{;\tilde{t}} = \begin{cases} \frac{d\tau}{dt}(\tilde{t}) - [u_x](\tilde{t})w(\tilde{t}) & \text{if } \xi^n > \xi^{n+1} \\ -\frac{d\tau}{dt}(\tilde{t}) + [u_x](\tilde{t})w(\tilde{t}) & \text{if } \xi^{n+1} > \xi^n \end{cases}$$

$$= \begin{cases} \frac{d\tau}{dt}(\tilde{t}) - [u_x](t^n)w^n + O(\Delta t) & \text{if } \xi^n > \xi^{n+1}, \\ -\frac{d\tau}{dt}(\tilde{t}) + [u_x](t^n)w^n + O(\Delta t) & \text{if } \xi^{n+1} > \xi^n. \end{cases} \tag{54}$$

Combining (52), (53) and (54), we have the following simplified algorithm:

Algorithm 2

Suppose we have obtained all necessary quantities at the time t^n, in other word, we have computed u_i^n, ξ^n and Δt^n.

1. Find i_0 such that $x_{i_0} \le \xi^n < x_{i_0+1}$.

2. Compute $u^{-,n}, u^{+,n}, u_x^{-,n}, u_x^{+,n}$

Set $\xi^{n+1} = \xi^n + \frac{\Delta t^n}{2}\left(3w\left(t^n, \xi^n;\ u^{-,n}, u^{+,n}, u_x^{-,n}, u_x^{+,n}\right) - w\left(t^{n-1}, \xi^{n-1};\ u^{-,n-1}, u^{+,n-1}, u_x^{-,n-1}, u_x^{+,n-1}\right)\right)$.

(Adams-Bashforth scheme, $w^{-1} = w^0$)

3. Compute $(\varepsilon u_x)_{x,i}^n$.

4. Find i_1 such that $x_{i_1} \leq \xi^{n+1} < x_{i_1+1}$.

5. Compute $C_{i,1}^n$ $\begin{cases} \text{If } i_0 < i_1 \text{ compute } C_{i,1}^n \text{ for } i = i_0+1, \dots, i_1, \\ \text{If } i_0 > i_1 \text{ compute } C_{i,1}^n \text{ for } i = i_1+1, \dots, i_0, \\ \text{If } i_0 = i_1 \text{ There is no need to comput } C_{i,1}^n. \end{cases}$

6. Use the following difference scheme

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} - C_{i,1}^n = \frac{1}{2}\left((\varepsilon u_x)_{x,i}^n + (\varepsilon u_x)_{x,i}^{n+1}\right) - \frac{1}{2}\left(f_i^n + f_i^{n+1}\right),$$

to form a linear system and solve for u_i^{n+1}

7. Accept u_i^{n+1} and ξ^{n+1} as approximations at time level t^{n+1}.

$$\Delta t^{n+1} = \min\left(h, \left|\frac{\Delta t^n}{w^{n+1} - w^n} w^n\right|\right).$$

Go to next time level.

Numerical results

Table 6. Numerical results for example 1 in JC 1.

t=0.1								
	$(\varepsilon_1, \varepsilon_2) = (3,1)$				$(\varepsilon_1, \varepsilon_2) = (2,1)$			
M	max-norm error	order	$\lvert E_\xi \rvert$	order	max-norm error	order	$\lvert E_\xi \rvert$	order
40	2.494×10^{-2}		4.407×10^{-2}		1.358×10^{-2}		1.313×10^{-3}	
80	6.707×10^{-3}	1.895	1.397×10^{-2}	1.658	4.046×10^{-3}	1.747	4.404×10^{-4}	1.576
160	1.792×10^{-3}	1.904	3.123×10^{-3}	2.161	1.092×10^{-3}	1.890	9.931×10^{-5}	2.149
320	4.624×10^{-4}	1.954	7.483×10^{-4}	2.061	2.818×10^{-4}	1.954	2.327×10^{-5}	2.094
640	1.174×10^{-4}	1.978	1.821×10^{-4}	2.039	7.178×10^{-5}	1.973	5.641×10^{-6}	2.044
1280	2.958×10^{-5}	1.989	4.499×10^{-5}	2.017	1.809×10^{-5}	1.988	1.385×10^{-6}	2.026

t=0.1								
	$(\varepsilon_1,\varepsilon_2)=(1.8,1)$				$(\varepsilon_1,\varepsilon_2)=(1,1)$			
M	max-norm error	order	$\lvert E_\xi\rvert$	order	max-norm error	order	$\lvert E_\xi\rvert$	order
40	1.456×10^{-2}		5.762×10^{-3}		1.343×10^{-2}		1.182×10^{-2}	
80	3.961×10^{-3}	1.878	1.578×10^{-3}	1.869	3.746×10^{-3}	1.842	2.661×10^{-3}	2.151
160	1.053×10^{-3}	1.911	3.511×10^{-4}	2.168	9.819×10^{-4}	1.932	6.458×10^{-4}	2.043
320	2.716×10^{-4}	1.955	8.250×10^{-5}	2.089	2.450×10^{-4}	2.003	1.517×10^{-4}	2.090
640	6.900×10^{-5}	1.977	1.995×10^{-5}	2.048	6.152×10^{-5}	1.994	3.712×10^{-5}	2.031
1280	1.737×10^{-5}	1.990	4.903×10^{-6}	2.025	1.540×10^{-5}	1.998	9.158×10^{-6}	2.019

Table 7. Numerical results for example 1 in JC 2.

	$t=1$			
M	max-norm error	order	$\lvert E_\xi\rvert$	order
40	9.417×10^{-4}		2.264×10^{-5}	
80	2.052×10^{-4}	2.198	5.232×10^{-6}	2.113
160	5.279×10^{-5}	1.956	1.282×10^{-6}	2.029
320	1.346×10^{-5}	1.972	3.198×10^{-7}	2.003
640	3.427×10^{-6}	1.974	7.934×10^{-8}	2.011
1280	8.374×10^{-7}	2.033	1.987×10^{-8}	1.998

Table 8. Numerical results for example 2 in JC 2.

t=0.1								
	$(\varepsilon_1,\varepsilon_2)=(3,1)$				$(\varepsilon_1,\varepsilon_2)=(2,1)$			
M	max-norm error	order	$\lvert E_\xi\rvert$	order	max-norm error	order	$\lvert E_\xi\rvert$	order
40	3.940×10^{-3}		2.023×10^{-2}		1.060×10^{-3}		7.578×10^{-4}	
80	9.034×10^{-4}	2.125	4.209×10^{-3}	2.265	4.174×10^{-4}	1.345	1.753×10^{-4}	2.112
160	2.317×10^{-4}	1.963	1.013×10^{-3}	2.055	1.248×10^{-4}	1.742	4.438×10^{-5}	1.982
320	6.335×10^{-5}	1.871	2.536×10^{-4}	1.998	3.409×10^{-5}	1.872	1.065×10^{-5}	2.059
640	1.649×10^{-5}	1.942	6.333×10^{-5}	2.002	8.900×10^{-6}	1.938	2.668×10^{-6}	1.997
1280	4.219×10^{-6}	1.967	1.584×10^{-5}	1.999	2.274×10^{-6}	1.969	6.588×10^{-7}	2.018

t=0.1								
	$(\varepsilon_1,\varepsilon_2)=(1.8,1)$				$(\varepsilon_1,\varepsilon_2)=(1,1)$			
M	max-norm error	order	$\lvert E_\xi \rvert$	order	max-norm error	order	$\lvert E_\xi \rvert$	order
40	1.549×10^{-3}		2.407×10^{-3}		2.836×10^{-3}		3.080×10^{-3}	
80	5.474×10^{-4}	1.501	6.182×10^{-4}	1.961	6.885×10^{-4}	2.042	2.719×10^{-4}	3.821
160	1.583×10^{-4}	1.790	1.551×10^{-4}	1.995	1.812×10^{-4}	1.926	1.988×10^{-4}	0.132
320	4.253×10^{-5}	1.896	3.821×10^{-5}	2.021	4.685×10^{-5}	1.952	6.088×10^{-5}	1.707
640	1.102×10^{-5}	1.948	9.464×10^{-6}	2.013	1.214×10^{-5}	1.948	1.532×10^{-5}	1.991
1280	2.806×10^{-6}	1.974	2.347×10^{-6}	2.012	3.107×10^{-6}	1.966	3.884×10^{-6}	1.980

If the equation is non-linear (i.e. $\lambda \neq 0$), the above method doesn't work because we need to approximate the $u_i^{n+1}u_{x,i}^{n+1}$ term. In this situation, we still use predict-correct approach. However we can use (53) and (54) to simplify our computation of the time correction term.

Table 9. Numerical results for example 2 in JC 1.

t=0.1								
	$(\varepsilon_1,\varepsilon_2)=(3,1)$				$(\varepsilon_1,\varepsilon_2)=(2,1)$			
M	max-norm error	order	$\lvert E_\xi \rvert$	order	max-norm error	order	$\lvert E_\xi \rvert$	order
40	2.948×10^{-2}		5.313×10^{-2}		1.355×10^{-2}		1.487×10^{-3}	
80	7.036×10^{-3}	2.067	1.494×10^{-2}	1.830	4.030×10^{-3}	1.749	4.547×10^{-4}	1.710
160	1.884×10^{-3}	1.901	3.309×10^{-3}	2.175	1.092×10^{-3}	1.884	1.003×10^{-4}	2.180
320	4.905×10^{-4}	1.942	7.881×10^{-4}	2.070	2.812×10^{-4}	1.957	2.330×10^{-5}	2.106
640	1.248×10^{-4}	1.975	1.920×10^{-4}	2.037	7.169×10^{-5}	1.972	5.644×10^{-6}	2.046
1280	3.146×10^{-5}	1.988	4.748×10^{-5}	2.016	1.806×10^{-5}	1.989	1.385×10^{-6}	2.027
t=0.1								
	$(\varepsilon_1,\varepsilon_2)=(1.8,1)$				$(\varepsilon_1,\varepsilon_2)=(1,1)$			
M	max-norm error	order	$\lvert E_\xi \rvert$	order	max-norm error	order	$\lvert E_\xi \rvert$	order
40	1.453×10^{-2}		6.392×10^{-3}		1.544×10^{-2}		1.476×10^{-2}	
80	3.949×10^{-3}	1.880	1.616×10^{-3}	1.984	4.104×10^{-3}	1.912	3.325×10^{-3}	2.150
160	1.051×10^{-3}	1.910	3.514×10^{-4}	2.201	1.084×10^{-3}	1.921	8.331×10^{-4}	1.997
320	2.709×10^{-4}	1.956	8.188×10^{-5}	2.102	2.718×10^{-4}	1.996	2.003×10^{-4}	2.056
640	6.884×10^{-5}	1.976	1.976×10^{-5}	2.051	6.875×10^{-5}	1.983	4.967×10^{-5}	2.012
1280	1.733×10^{-5}	1.990	4.853×10^{-6}	2.026	1.722×10^{-5}	1.997	1.232×10^{-5}	2.011

In summary, we apply the CIM to the one dimensional moving interface problems using Crank-Nicholson scheme to solve the pde and prediction correction approach to move the interface. We verify the method is second-order accurate. We also make some modifications to make the computation and the generalization to higher dimension easier.

Chapter 3
Coupling Interface Method in Two Dimensions

In this chapter we review the coupling interface method (CIM) in two dimensions under Cartesian grid for solving interface problems [5]. It contains a first-order method (CIM1) and a second-order method (CIM2).

Let $[a, b] \times [a, b]$ be our domain Ω of consideration. Denote the interface in Ω by Γ. Consider the elliptic interface problem $\nabla \cdot (\varepsilon \nabla u) = f$ on Ω. The elliptic coefficient $\varepsilon(x, y) > 0$ may have jumps across Γ. We partition $[a, b]$ into $M + 1$ subintervals evenly and define

$$h \triangleq \frac{b-a}{M+1}, \text{(mesh size)}$$

$$x_i \triangleq a + ih, y_j \triangleq a + jh, 0 \leq i, j \leq M + 1, \text{(grid points)}$$

First, we classify the grid points into 2 categories. Let $\gamma_{i+1/2,j}$ and $\gamma_{i,j+1/2}$ denote the number of intersections of the interface Γ and the grid segment $[x_i, x_{i+1}) \times \{y_j\}$ and $\{x_i\} \times [y_j, y_{j+1})$, respectively. We assume $\gamma_{i+1/2,j} \leq 1$ and $\gamma_{i,j+1/2} \leq 1$ for all i and j. A grid point (x_i, y_j) is called an interior grid point if $\gamma_{i\pm1/2,j} = \gamma_{i,j\pm1/2} = 0$, i.e. none of its four neighboring segments intersects Γ. Otherwise, we call it an on-front grid point. At an interior point (x_i, y_j), we approximate $(\varepsilon u_x)_{x,i,j}$ and $(\varepsilon u_y)_{y,i,j}$ by a standard central finite difference scheme. Namely,

$$(\varepsilon u_x)_{x,i,j} = \frac{\varepsilon_{i-1/2,j} u_{i-1,j} - (\varepsilon_{i-1/2,j} + \varepsilon_{i+1/2,j}) u_{i,j} + \varepsilon_{i+1/2,j} u_{i+1,j}}{h^2} + O(h^2), \tag{55}$$

$$\left(\varepsilon u_y\right)_{y,i,j} = \frac{\varepsilon_{i,j-1/2}u_{i,j-1} - \left(\varepsilon_{i,j-1/2} + \varepsilon_{i,j+1/2}\right)u_{i,j} + \varepsilon_{i,j+1/2}u_{i,j+1}}{h^2} + O(h^2). \qquad (56)$$

At an on-front grid point $\left(x_i, y_j\right)$, we discuss the CIM1 and CIM2 below. In both methods, we call the region where $\left(x_i, y_j\right)$ is located the Ω^- region, and the other region the Ω^+ region. Given an interface point p in the x-direction, the jump conditions we know are $[u]_p$ and $[\varepsilon u_n]_p$. Now, let $\overrightarrow{n_p} = \left(n_p^x, n_p^y\right)$ and $\overrightarrow{t_p} = \left(t_p^x, t_p^y\right)$ be the unit normal and tangential vectors of Γ at p, respectively (Note that the directions of $\overrightarrow{n_p}$ and $\overrightarrow{t_p}$ are irrelevant). First, we try to decompose the jump datum $[\varepsilon u_x]_p$ into its tangential and normal directions. Since

$$\begin{aligned}(\nabla u^{\pm})_p &= \left((\nabla u^{\pm})_p \cdot \overrightarrow{n_p}\right)\overrightarrow{n_p} + \left((\nabla u^{\pm})_p \cdot \overrightarrow{t_p}\right)\overrightarrow{t_p} \\ &= \left(u_n^{\pm}\right)_p\left(n_p^x, n_p^y\right) + \left(u_t^{\pm}\right)_p\left(t_p^x, t_p^y\right) \\ &= \left(\left(u_n^{\pm}\right)_p n_p^x + \left(u_t^{\pm}\right)_p t_p^x, \left(u_n^{\pm}\right)_p n_p^y + \left(u_t^{\pm}\right)_p t_p^y\right),\end{aligned}$$

therefore,

$$\begin{bmatrix}\left(u_x^{\pm}\right)_p \\ \left(u_y^{\pm}\right)_p\end{bmatrix} = \begin{bmatrix}\left(u_n^{\pm}\right)_p n_p^x + \left(u_t^{\pm}\right)_p t_p^x \\ \left(u_n^{\pm}\right)_p n_p^y + \left(u_t^{\pm}\right)_p t_p^y\end{bmatrix}.$$

Hence, from the above formula, we can get

$$\begin{aligned}[\varepsilon u_x]_p &= \varepsilon_p^+(u_x^+)_p - \varepsilon_p^-(u_x^-)_p \\ &= \varepsilon_p^+\left((u_n^+)_p n_p^x + (u_t^+)_p t_p^x\right) - \varepsilon_p^-\left((u_n^-)_p n_p^x + (u_t^-)_p t_p^x\right) \\ &= \left(\varepsilon_p^+(u_n^+)_p - \varepsilon_p^-(u_n^-)_p\right)n_p^x + \left(\varepsilon_p^+(u_t^+)_p - \varepsilon_p^-(u_t^-)_p\right)t_p^x \\ &= [\varepsilon u_n]_p n_p^x + [\varepsilon u_t]_p t_p^x\end{aligned}$$

$$= [\varepsilon u_n]_p n_p^x + \left(\varepsilon_p^+ [u_t]_p + \left(\varepsilon_p^+ - \varepsilon_p^-\right)(u_t^-)_p\right) t_p^x. \tag{57}$$

Similarly, given an interface point q in the y-direction, the jump conditions we know are $[u]_q$ and $[\varepsilon u_n]_q$. And we have

$$[\varepsilon u_y]_q = [\varepsilon u_n]_q n_q^y + \left(\varepsilon_q^+ [u_t]_q + \left(\varepsilon_q^+ - \varepsilon_q^-\right)(u_t^-)_q\right) t_q^y. \tag{58}$$

where $\overrightarrow{n_q} = \left(n_q^x, n_q^y\right)$ and $\overrightarrow{t_q} = \left(t_q^x, t_q^y\right)$ are the unit normal and tangential vectors of Γ at q, respectively.

CIM1

We deal with $(u_x)_{i\pm1/2,j}$ first. We apply the one dimensional formula (11) to the x-direction to get

$$(u_x)_{i\pm1/2,j} = \frac{1}{h}\left(\left(1 + \gamma_{i\pm1/2,j}\left(\bar{\rho}_p^+ - 1\right)\right)\bar{D}_x^{(\pm1/2)} u_{i,j} + \gamma_{i\pm1/2,j}\left(\mp\bar{\rho}_p^+ [u]_p - \beta_p h \frac{[\varepsilon u_x]_p}{\bar{\varepsilon}_p}\right)\right) + O(h), \tag{59}$$

where $\bar{D}_x^{(\pm1/2)}$ is the extension of (12) in the x-direction and the coefficients are defined with respect to the intersection p. By (57) , we may rewrite (59) as

$$(u_x)_{i\pm1/2,j} = \frac{1}{h}\left(\left(1 + \gamma_{i\pm1/2,j}\left(\bar{\rho}_p^+ - 1\right)\right)\bar{D}_x^{(\pm1/2)} u_{i,j} + \gamma_{i\pm1/2,j}\left(h\bar{b}_p (u_t^-)_p t_p^x + \bar{J}_p^{(\pm1/2)}\right)\right) + O(h), \tag{60}$$

where

$$\begin{cases} \bar{b}_p = -\beta_p\left(\bar{\rho}_p^+ - \bar{\rho}_p^-\right), \\ \bar{J}_p^{(\pm1/2)} = \mp\bar{\rho}_p^+ [u]_p - \beta_p h\left(\dfrac{[\varepsilon u_n]_p}{\bar{\varepsilon}_p} n_p^x + \bar{\rho}_p^+ [u_t]_p t_p^x\right). \end{cases}$$

It remains to compute $(u_t^-)_p$. Since

$$(u_t^-)_p = (\nabla u^-)_p \cdot \overrightarrow{t_p}$$

$$= (u_x^-)_p t_p^x + \left(u_y^-\right)_p t_p^y, \tag{61}$$

we approximate $(u_x^-)_p$ and $(u_y^-)_p$ by the following formulas:

$$\begin{cases} (u_x^-)_p = (u_x)_{i\pm1/2,j} + O(h), \\ (u_y^-)_p = (u_y)_{i,j\pm1/2} + O(h). \end{cases} \tag{62}$$

Combining (60), (61) and (62), we have

$$(u_x)_{i\pm\frac{1}{2},j} = \frac{1}{h}\left(\left(1+\gamma_{i\pm\frac{1}{2},j}(\bar{\rho}_p^+ - 1)\right)\bar{D}_x^{\left(\pm\frac{1}{2}\right)}u_{i,j} + \gamma_{i\pm\frac{1}{2},j}\left(h\bar{b}_p\left((u_x)_{i\pm\frac{1}{2},j}t_p^x + (u_y)_{i,j\pm\frac{1}{2}}t_p^y\right)t_p^x + \bar{J}_p^{\left(\pm\frac{1}{2}\right)}\right)\right) + O(h).$$

After some arrangements of the above equation, we get

$$\frac{1}{h}\left(\left(1+\gamma_{i\pm1/2,j}(\bar{\rho}_p^+ - 1)\right)\bar{D}_x^{(\pm1/2)}u_{i,j} + \gamma_{i\pm1/2,j}\bar{J}_p^{(\pm1/2)}\right)$$
$$= \left(1-\gamma_{i\pm1/2,j}\bar{b}_p\left(t_p^x\right)^2\right)(u_x)_{i\pm1/2,j} - \left(\gamma_{i\pm1/2,j}\bar{b}_p t_p^x t_p^y\right)(u_y)_{i,j\pm1/2} + O(h). \tag{63}$$

Next, we consider $(u_y)_{i,j\pm1/2}$. We apply the one dimensional formula (11) to the y-direction to get

$$(u_y)_{i,j\pm1/2} = \frac{1}{h}\left(\left(1+\gamma_{i,j\pm1/2}(\bar{\rho}_q^+ - 1)\right)\bar{D}_y^{(\pm1/2)}u_{i,j} + \gamma_{i,j\pm1/2}\left(\mp\bar{\rho}_q^+[u]_q - \beta_q h\frac{[\varepsilon u_y]_q}{\bar{\varepsilon}_q}\right)\right) + O(h)\,, \tag{64}$$

where $\bar{D}_y^{(\pm1/2)}$ is the extension of (12) in the y-direction and the coefficients are defined with respect to the intersection q. Follow the similar procedure as in computing $(u_x)_{i\pm1/2,j}$, we have

$$\frac{1}{h}\left(\left(1+\gamma_{i,j\pm1/2}(\bar{\rho}_q^+ - 1)\right)\bar{D}_y^{(\pm1/2)}u_{i,j} + \gamma_{i,j\pm1/2}\bar{J}_q^{(\pm1/2)}\right)$$
$$= -\gamma_{i,j\pm1/2}\bar{b}_q t_q^x t_q^y (u_x)_{i\pm1/2,j} + \left(1-\gamma_{i,j\pm1/2}\bar{b}_q\left(t_q^y\right)^2\right)(u_y)_{i,j\pm1/2} + O(h), \tag{65}$$

where

$$\begin{cases} \bar{b}_q = -\beta_q\left(\bar{\rho}_q^+ - \bar{\rho}_q^-\right), \\ \bar{J}_q^{(\pm 1/2)} = \mp\bar{\rho}_q^+[u]_q - \beta_q h\left(\frac{[\varepsilon u_n]_q}{\bar{\varepsilon}_q} n_q^y + \bar{\rho}_q^+[u_t]_q t_q^y\right). \end{cases}$$

Combining (63) and (65), we can form a 2 by 2 linear system

$$\begin{bmatrix} 1-\gamma_{i\pm1/2,j}\bar{b}_p\left(t_p^x\right)^2 & -\gamma_{i\pm1/2,j}\bar{b}_p t_p^x t_p^y \\ -\gamma_{i,j\pm1/2}\bar{b}_q t_q^x t_q^y & 1-\gamma_{i,j\pm1/2}\bar{b}_q\left(t_q^y\right)^2 \end{bmatrix} \begin{bmatrix} (u_x)_{i\pm1/2,j} \\ \left(u_y\right)_{i,j\pm1/2} \end{bmatrix}$$

$$= \frac{1}{h}\begin{bmatrix} \left(\left(1+\gamma_{i\pm1/2,j}\left(\bar{\rho}_p^+ - 1\right)\right)\bar{D}_x^{(\pm1/2)} u_{i,j} + \gamma_{i\pm1/2,j}\bar{J}_p^{(\pm1/2)}\right) \\ \left(\left(1+\gamma_{i,j\pm1/2}\left(\bar{\rho}_q^+ - 1\right)\right)\bar{D}_y^{(\pm1/2)} u_{i,j} + \gamma_{i,j\pm1/2}\bar{J}_q^{(\pm1/2)}\right) \end{bmatrix} + \begin{bmatrix} O(h) \\ O(h) \end{bmatrix}.$$

Solving this linear system, we can get a first order approximation of $(u_x)_{i\pm1/2,j}$ and $\left(u_y\right)_{i,j\pm1/2}$. Finally, we approximate $(\varepsilon u_x)_{x,i,j}$ and $\left(\varepsilon u_y\right)_{y,i,j}$ by

$$(\varepsilon u_x)_{x,i,j} = \begin{cases} \frac{\varepsilon_{i-1/2,j}u_{i-1,j} - (\varepsilon_{i-1/2,j}+\varepsilon_{i+1/2,j})u_{i,j} + \varepsilon_{i+1/2,j}u_{i+1,j}}{h^2} + O(h^2) & \text{if } \gamma_{i+1/2,j}+\gamma_{i-1/2,j} = 0, \\ \frac{\varepsilon_{i,j}}{h}\left((u_x)_{i+1/2,j} - (u_x)_{i-1/2,j}\right) + O(1) & \text{if } \gamma_{i+1/2,j}+\gamma_{i-1/2,j} \neq 0. \end{cases} \quad (66)$$

$$\left(\varepsilon u_y\right)_{y,i,j} = \begin{cases} \frac{\varepsilon_{i,j-1/2}u_{i,j-1} - (\varepsilon_{i,j-1/2}+\varepsilon_{i,j+1/2})u_{i,j} + \varepsilon_{i,j+1/2}u_{i,j+1}}{h^2} + O(h^2) & \text{if } \gamma_{i,j+1/2}+\gamma_{i,j-1/2} = 0, \\ \frac{\varepsilon_{i,j}}{h}\left(\left(u_y\right)_{i,j+1/2} - \left(u_y\right)_{i,j-1/2}\right) + O(1) & \text{if } \gamma_{i,j+1/2}+\gamma_{i,j-1/2} \neq 0. \end{cases} \quad (67)$$

CIM2

First, we classify the on-front grid points into normal and exceptional. We define the interface orientation indicator $s_{x,(i,j)}$ and $s_{y,(i,j)}$ to be $\gamma_{i+1/2,j} - \gamma_{i-1/2,j}$ and $\gamma_{i,j+1/2} - \gamma_{i,j-1/2}$, respectively. A normal on-front grid point (x_i, y_j) is defined to satisfy the following conditions:

(a) $\gamma_{i+1/2,j} + \gamma_{i-1/2,j} = 0$ or 1

So $s_{x,(i,j)}$ have three possible values: $-1, 0, 1$ and

$s_{x,(i,j)} = -1 \Leftrightarrow$ the only interface point in $[x_{i-1}, x_{i+1}) \times \{y_j\}$ is contained in $[x_{i-1}, x_i) \times \{y_j\}$.

$s_{x,(i,j)} = 1 \iff$ the only interface point in $[x_{i-1}, x_{i+1}) \times \{y_j\}$ is contained in $[x_i, x_{i+1}) \times \{y_j\}$.

$s_{x,(i,j)} = 0 \iff$ there is no interface point in $[x_{i-1}, x_{i+1}) \times \{y_j\}$.

(A parallel argument applies to the y-direction)

(b) If $s_{x,(i,j)} = \pm 1$, then $\gamma_{i\pm 3/2,j} = 0$.

(A parallel argument applies to the y-direction)

(c) If $s_{x,(i,j)} = \pm 1$, then

$$\begin{cases} \text{if } s_{y,(i,j)} = 0, \text{then } \gamma_{i-s_{x,(i,j)},j+1/2} = \gamma_{i-s_{x,(i,j)},j-1/2} = 0, \\ \text{if } s_{y,(i,j)} = 1, \text{then } \gamma_{i-s_{x,(i,j)},j-1/2} = 0, \\ \text{if } s_{y,(i,j)} = -1, \text{then } \gamma_{i-s_{x,(i,j)},j+1/2} = 0. \end{cases}$$

(A parallel argument applies to the y-direction)

Condition (a) and (b) mean that we can apply the CIM2 to both x and y-directions. Condition (c) allows us to approximate the cross derivatives by one-side interpolation (its meaning will be clear soon). Figure 13 contains all 8 possible cases of a normal on-front grid point.

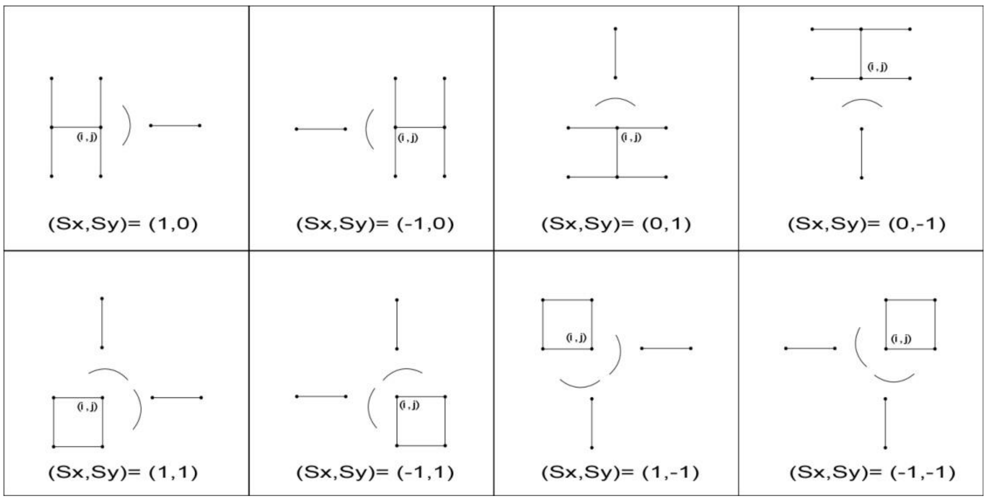

Figure 13. 8 possible cases of a normal on-front grid point. Bullet: grid point. Curve: interface. No interface crosses through straight lines

Given a normal on-front grid point (x_i, y_j), our purpose is to derive a first order approximation of $(u_{xx})_{i,j}$ and $(u_{yy})_{i,j}$. We deal with $(u_{xx})_{i,j}$ first. For simplicity, we drop the sub-index (i,j) from $s_{x,(i,j)}$. We apply the one dimensional formula (23) to the x-direction to get

$$(u_{xx})_{i,j} = \frac{1}{h^2}\left(L_x^{(s_x)}u_{i,j} - |s_x|(1+2\beta_p)\rho_p^+[u]_p - s_x(\beta_p+\beta_p^2)h\frac{[\varepsilon u_x]_p}{\hat{\varepsilon}_p}\right) + O(h^{2-|s_x|}), \qquad (68)$$

where $L_x^{(s_x)}$ is the extension of (22) in the x-direction and the coefficients are defined with respect to the intersection p. By (57) , we may rewrite (68) as

$$(u_{xx})_{i,j} = \frac{1}{h^2}\left(L_x^{(s_x)}u_{i,j} + hs_xb_p(u_t^-)_pt_p^x + J_p\right) + O(h^{2-|s_x|}), \qquad (69)$$

where

$$\begin{cases} b_p = -(\beta_p+\beta_p^2)(\rho_p^+ - \rho_p^-), \\ J_p = -\left(|s_x|(1+2\beta_p)\rho_p^+[u]_p + s_x(\beta_p+\beta_p^2)h\left(\frac{[\varepsilon u_n]_p}{\widehat{\varepsilon_p}}n_p^x + \rho_p^+[u_t]_pt_p^x\right)\right). \end{cases}$$

It remains to compute $(u_t^-)_p$. Since

$$\begin{aligned}(u_t^-)_p &= (\nabla u^-)_p \cdot \overrightarrow{t_p} \\ &= (u_x^-)_pt_p^x + (u_y^-)_pt_p^y, \qquad (70)\end{aligned}$$

we need to approximate $(u_x^-)_p$ and $(u_y^-)_p$. First, we approximate $(u_x^-)_p$.

Assume $s_x = 1$. By (17) we may compute $(u_x^-)_p$ as

$$(u_x^-)_p = \frac{u_{i,j} - u_{i-1,j}}{h} + \left(\frac{1}{2}+\alpha_p\right)h(u_{xx})_{i,j} + O(h^2).$$

By a similar derivation for the case $s_x = -1$ we can get a unified formula:

$$(u_x^-)_p = \frac{1}{h}D_{x,(s_x)}u_{i,j} + s_x\left(\frac{1}{2}+\alpha_p\right)h(u_{xx})_{i,j} + O(h^2), \qquad (71)$$

where $D_{x,(s_x)}u_{i,j}$ is defined as

$$D_{x,(s_x)}u(x,y) = \begin{cases} \frac{1}{2}\left(u(x+h,y) - u(x-h,y)\right) & \text{if } s_x = 0, \\ u(x,y) - u(x-h,y) & \text{if } s_x = 1, \\ u(x+h,y) - u(x,y) & \text{if } s_x = -1. \end{cases}$$

Next, we approximate $\left(u_y^-\right)_p$. By Taylor expansion we have

$$\begin{aligned}\left(u_y^-\right)_p &= \left(u_y\right)_{i,j} + \left(u_{yx}^-\right)_{i,j}\left(x_p - x_i\right) + O(h^2) \\ &= \left(u_y\right)_{i,j} + s_x\alpha_p h\left(u_{yx}^-\right)_{i,j} + O(h^2).\end{aligned}$$

By one-side interpolation we know

$$\left(u_{yx}^-\right)_{i,j} = s_x\left(\frac{\left(u_y\right)_{i,j} - \left(u_y\right)_{i-s_x,j}}{h}\right) + O(h),$$

so

$$\left(u_y^-\right)_p = \left(1+\alpha_p\right)\left(u_y\right)_{i,j} - \alpha_p\left(u_y\right)_{i-s_x,j} + O(h^2). \tag{72}$$

It suffices to approximate $\left(u_y\right)_{i,j}$ and $\left(u_y\right)_{i-s_x,j}$. We separate it into three cases.

(A) $s_x = \pm 1 \; s_y = 0$

$$\left(u_y\right)_{i,j} = \frac{u_{i,j+1} - u_{i,j-1}}{2h} + O(h^2),$$

$$\left(u_y\right)_{i-s_x,j} = \frac{u_{i-s_x,j+1} - u_{i-s_x,j-1}}{2h} + O(h^2).$$

So from (72) we get

$$\left(u_y^-\right)_p = \left(1+\alpha_p\right)\left(\frac{u_{i,j+1} - u_{i,j-1}}{2h}\right) - \alpha_p\left(\frac{u_{i-s_x,j+1} - u_{i-s_x,j-1}}{2h}\right) + O(h^2). \tag{73}$$

(B) $s_x = \pm 1 \; s_y = 1$

$$\left(u_y\right)_{i,j} = \frac{u_{i,j} - u_{i,j-1}}{h} + \frac{1}{2}\left(u_{yy}\right)_{i,j}h + O(h^2),$$

$$\left(u_y\right)_{i-s_x,j} = \frac{u_{i-s_x,j} - u_{i-s_x,j-1}}{h} + \frac{1}{2}\left(u_{yy}\right)_{i-s_x,j} h + O(h^2).$$

So from (72) we get

$$\left(u_y^-\right)_p = \left(1+\alpha_p\right)\left(\frac{u_{i,j} - u_{i,j-1}}{h}\right) - \alpha_p\left(\frac{u_{i-s_x,j} - u_{i-s_x,j-1}}{h}\right) + \frac{1}{2}\left(u_{yy}\right)_{i,j} h + O(h^2). \qquad (74)$$

(C) $s_x = \pm 1\ s_y = -1$

$$\left(u_y\right)_{i,j} = \frac{u_{i,j+1} - u_{i,j}}{h} - \frac{1}{2}\left(u_{yy}\right)_{i,j} h + O(h^2),$$

$$\left(u_y\right)_{i-s_x,j} = \frac{u_{i-s_x,j+1} - u_{i-s_x,j}}{h} - \frac{1}{2}\left(u_{yy}\right)_{i-s_x,j} h + O(h^2).$$

So from (72) we get

$$\left(u_y^-\right)_p = \left(1+\alpha_p\right)\left(\frac{u_{i,j+1} - u_{i,j}}{h}\right) - \alpha_p\left(\frac{u_{i-s_x,j+1} - u_{i-s_x,j}}{h}\right) - \frac{1}{2}\left(u_{yy}\right)_{i,j} h + O(h^2). \qquad (75)$$

Combining (73), (74) and (75), we can derive a unified formula for $\left(u_y^-\right)_p$:

$$\left(u_y^-\right)_p = \left(1+\alpha_p\right)\frac{1}{h} D_{y,(s_y)} u_{i,j} - \alpha_p \frac{1}{h} D_{y,(s_y)} u_{i-s_x,j} + s_y \frac{h}{2}\left(u_{yy}\right)_{i,j} + O(h^2), \qquad (76)$$

where $D_{y,(s_y)}u(x,y)$ is defined as

$$D_{y,(s_y)}u(x,y) = \begin{cases} \frac{1}{2}\left(u(x,y+h) - u(x,y-h)\right) & \text{if } s_y = 0, \\ u(x,y) - u(x,y-h) & \text{if } s_y = 1, \\ u(x,y+h) - u(x,y) & \text{if } s_y = -1. \end{cases}$$

Combining (70), (71) and (76), we can write $(u_t^-)_p$ as

$$\begin{aligned} (u_t^-)_p &= \left(\frac{1}{h} D_{x,(s_x)} u_{i,j} + s_x\left(\frac{1}{2} + \alpha_p\right) h (u_{xx})_{i,j}\right) t_p^x \\ &\quad + \left(\left(1+\alpha_p\right)\frac{1}{h} D_{y,(s_y)} u_{i,j} - \alpha_p \frac{1}{h} D_{y,(s_y)} u_{i-s_x,j} + s_y \frac{h}{2}\left(u_{yy}\right)_{i,j}\right) t_p^y + O(h^2) \\ &= \frac{1}{h} T_x u_{i,j} + h\left(s_x\left(\frac{1}{2} + \alpha_p\right)(u_{xx})_{i,j} t_p^x + \frac{1}{2} s_y \left(u_{yy}\right)_{i,j} t_p^y\right) + O(h^2), \end{aligned} \qquad (77)$$

where

$$T_x u_{i,j} \triangleq D_{x,(s_x)} u_{i,j} t_p^x + \left(\left(1+\alpha_p\right) D_{y,(s_y)} u_{i,j} - \alpha_p D_{y,(s_y)} u_{i-s_x,j}\right) t_p^y.$$

Finally, combining (69) and (77), we derive

$$(u_{xx})_{i,j} = \frac{1}{h^2}\left(L_x^{(s_x)}u_{i,j} + hs_xb_p\left(\frac{1}{h}T_xu_{i,j} + h\left(s_x\left(\frac{1}{2}+\alpha_p\right)(u_{xx})_{i,j}t_p^x + \frac{1}{2}s_y\left(u_{yy}\right)_{i,j}t_p^y\right)\right)t_p^x + J_p\right) + O\left(h^{2-|s_x|}\right),$$

After some arrangements of the above equation, we get

$$\frac{1}{h^2}\left(L_x^{(s_x)}u_{i,j} + s_xb_pT_xu_{i,j}t_p^x + J_p\right)$$
$$= \left(1 - |s_x|\left(\frac{1}{2}+\alpha_p\right)b_p\left(t_p^x\right)^2\right)(u_{xx})_{i,j} + \left(\frac{-1}{2}s_xs_yb_pt_p^xt_p^y\right)\left(u_{yy}\right)_{i,j} + O\left(h^{2-|s_x|}\right). \quad (78)$$

Next, we consider $\left(u_{yy}\right)_{i,j}$. We apply the one dimensional formula (23) to the y-direction to get

$$\left(u_{yy}\right)_{i,j} = \frac{1}{h^2}\left(L_y^{(s_y)}u_{i,j} - |s_y|\left(1+2\beta_q\right)\rho_q^+[u]_q - s_y\left(\beta_q+\beta_q^2\right)h\frac{\left[\varepsilon u_y\right]_q}{\hat{\varepsilon}_q}\right) + O\left(h^{2-|s_y|}\right), \quad (79)$$

where $L_y^{(s_y)}$ is the extension of (22) in the y-direction and the coefficients are defined with respect to the intersection q. Follow the similar procedure as in computing $(u_{xx})_{i,j}$, we have

$$\left(u_{yy}\right)_{i,j} = \frac{1}{h^2}\left(L_y^{(s_y)}u_{i,j} + hs_yb_q(u_t^-)_qt_q^y + J_q\right) + O\left(h^{2-|s_y|}\right), \quad (80)$$

where

$$\begin{cases} b_q = -\left(\beta_q+\beta_q^2\right)\left(\rho_q^+ - \rho_q^-\right), \\ J_q = -\left(|s_y|\left(1+2\beta_q\right)\rho_q^+[u]_q + s_y\left(\beta_q+\beta_q^2\right)h\left(\frac{[\varepsilon u_n]_q}{\widehat{\varepsilon_q}}n_q^y + \rho_q^+[u_t]_qt_q^y\right)\right). \end{cases}$$

With the following estimation of $\left(u_y^-\right)_q$ and $(u_x^-)_q$

$$\left(u_y^-\right)_q = \frac{1}{h}D_{y,(s_y)}u_{i,j} + s_y\left(\frac{1}{2}+\alpha_q\right)h\left(u_{yy}\right)_{i,j} + O(h^2),$$

$$(u_x^-)_q = (1+\alpha_q)\frac{1}{h}D_{x,(s_x)}u_{i,j} - \alpha_q\frac{1}{h}D_{x,(s_x)}u_{i,j-s_y} + s_x\frac{h}{2}(u_{xx})_{i,j} + O(h^2),$$

we can compute $(u_t^-)_q$ as

$$\begin{aligned}(u_t^-)_q &= (u_x^-)_q t_q^x + \left(u_y^-\right)_q t_q^y \\ &= \frac{1}{h}T_y u_{i,j} + h\left(s_y\left(\frac{1}{2}+\alpha_q\right)\left(u_{yy}\right)_{i,j}t_q^y + \frac{1}{2}s_x(u_{xx})_{i,j}t_q^x\right) + O(h^2),\end{aligned} \tag{81}$$

where

$$T_y u_{i,j} \triangleq D_{y,(s_y)}u_{i,j}t_q^y + \left((1+\alpha_q)D_{x,(s_x)}u_{i,j} - \alpha_q D_{x,(s_x)}u_{i,j-s_y}\right)t_q^x.$$

Finally, combining (80) and (81), after some arrangements, we can get

$$\begin{aligned}&\frac{1}{h^2}\left(L_y^{(s_y)}u_{i,j} + s_y b_q T_y u_{i,j} t_q^y + J_q\right) \\ &= \left(\frac{-1}{2}s_x s_y b_q t_q^x t_q^y\right)(u_{xx})_{i,j} + \left(1-|s_y|\left(\frac{1}{2}+\alpha_q\right)b_q\left(t_q^y\right)^2\right)\left(u_{yy}\right)_{i,j} + O\left(h^{2-|s_y|}\right).\end{aligned} \tag{82}$$

Combining (78) and (82), we can form a 2 by 2 linear system

$$\begin{bmatrix} 1-|s_x|\left(\frac{1}{2}+\alpha_p\right)b_p\left(t_p^x\right)^2 & \frac{-1}{2}s_x s_y b_p t_p^x t_p^y \\ \frac{-1}{2}s_x s_y b_q t_q^x t_q^y & 1-|s_y|\left(\frac{1}{2}+\alpha_q\right)b_q\left(t_q^y\right)^2 \end{bmatrix}\begin{bmatrix}(u_{xx})_{i,j} \\ \left(u_{yy}\right)_{i,j}\end{bmatrix}$$
$$= \frac{1}{h^2}\begin{bmatrix} L_x^{(s_x)}u_{i,j} + s_x b_p T_x u_{i,j} t_p^x + J_p \\ L_y^{(s_y)}u_{i,j} + s_y b_q T_y u_{i,j} t_q^y + J_q \end{bmatrix} + \begin{bmatrix} O\left(h^{2-|s_x|}\right) \\ O\left(h^{2-|s_y|}\right)\end{bmatrix}.$$

Solving this linear system, we can get a first order approximation of $(u_{xx})_{i,j}$ and $\left(u_{yy}\right)_{i,j}$. Finally we approximate $(\varepsilon u_x)_{x,i,j}$ and $\left(\varepsilon u_y\right)_{y,i,j}$ by

$$(\varepsilon u_x)_{x,i,j} = \begin{cases} \frac{\varepsilon_{i-1/2,j}u_{i-1,j} - (\varepsilon_{i-1/2,j}+\varepsilon_{i+1/2,j})u_{i,j} + \varepsilon_{i+1/2,j}u_{i+1,j}}{h^2} + O(h^2) & \text{if } s_x = 0, \\ \frac{1}{h^2}\left(D_{x,(s_x)}\varepsilon_{i,j}D_{x,(s_x)}u_{i,j} + \varepsilon_{i,j}h^2(u_{xx})_{i,j}\right) + O(h) & \text{if } s_x = \pm 1. \end{cases} \tag{83}$$

$$\left(\varepsilon u_y\right)_{y,i,j} = \begin{cases} \frac{\varepsilon_{i,j-1/2}u_{i,j-1} - (\varepsilon_{i,j-1/2}+\varepsilon_{i,j+1/2})u_{i,j} + \varepsilon_{i,j+1/2}u_{i,j+1}}{h^2} + O(h^2) & \text{if } s_y = 0, \\ \frac{1}{h^2}\left(D_{y,(s_y)}\varepsilon_{i,j}D_{y,(s_y)}u_{i,j} + \varepsilon_{i,j}h^2\left(u_{yy}\right)_{i,j}\right) + O(h) & \text{if } s_y = \pm 1. \end{cases} \tag{84}$$

In summary, at an interior point we adopt a standard central finite difference scheme, which produces $O(h^2)$ local truncation error. At an exceptional on-front grid point, the CIM1 produces $O(1)$ local truncation error and at a normal on-front grid point, the CIM2 produces $O(h)$ local truncation error. Since the number of exceptional points is $O(1)$, so the global error is $O(h^2)$. Refer to [5] for numerical examples and more information about the CIM (ex: non-singularity of the coupling matrix).

Chapter 4
Application to Two Dimensional Diffusion Equations with Fixed Interface

In this chapter we apply the coupling interface method to the two dimensional diffusion equations with fixed interface. Let $[a,b] \times [a,b]$ be our domain Ω of consideration. Denote the interface in Ω by Γ. The model problem we considered is the following:

$$\frac{\partial u}{\partial t} = \nabla \cdot (\varepsilon \nabla u) - f(x,y,t) \quad (x,y) \in \Omega \setminus \Gamma. \tag{85}$$

The coefficient $\varepsilon(x,y,t) > 0$ and the source term $f(x,y,t)$ are smooth functions in $\Omega \setminus \Gamma$ but may have discontinuity across the interface Γ. This is a parabolic problem and the solution in each domain is smooth. Recall that given an on-front grid point (x_i, y_j), we call the region where (x_i, y_j) is situated the Ω^- region, whereas the other the Ω^+ region. Across the interface, there are two kinds of jump conditions considered.

JC 1. Jump conditions at the interface of the form

$$[u](\Gamma,t) \triangleq u^+ - u^- = \tau(\Gamma,t), \tag{86}$$

$$[\varepsilon u_n](\Gamma,t) \triangleq \varepsilon^+ {u_n}^+ - \varepsilon^- {u_n}^- = \sigma(\Gamma,t), \tag{87}$$

are given. As a model problem consider heat conduction with a heat source applied only along the interface Γ. Then $f(x,y,t)$ can be written as

$$f(x,y,t) = \int_\Gamma C(s,t)\delta(x - X(s))\delta(y - Y(s))ds.$$

From the differential equation we know that across Γ, the jump in temperature is zero. But there is a jump in the normal derivative which equals the strength of the source $C(s,t)$.

JC 2. The solution on the interface

$$u(\Gamma,t)=r(\Gamma,t), \tag{88}$$

is given.

Numerical Method

We partition $[a,b]$ into $M+1$ subintervals evenly and define

$h \triangleq \dfrac{b-a}{M+1}$, (mesh size)

$x_i \triangleq a+ih, y_j \triangleq a+jh, 0 \le i,j \le M+1,$ (grid points)

We use Δt as the temporal step size in time and assume the ratio $\Delta t/h$ is a constant (ex: equals to 1). Using the Crank-Nicholson scheme, the semi-discrete difference scheme for (85) can be written in the following form:

$$\frac{u_{i,j}^{n+1}-u_{i,j}^{n}}{\Delta t}=\frac{1}{2}\left((\varepsilon u_x)_{x,i,j}^{n}+(\varepsilon u_x)_{x,i,j}^{n+1}\right)+\frac{1}{2}\left(\left(\varepsilon u_y\right)_{y,i,j}^{n}+\left(\varepsilon u_y\right)_{y,i,j}^{n+1}\right)-\frac{1}{2}\left(f_{i,j}^{n}+f_{i,j}^{n+1}\right), \tag{89}$$

where $(\varepsilon u_x)_{x,i,j}^{n}$ and $\left(\varepsilon u_y\right)_{y,i,j}^{n}$ are $(\varepsilon u_x)_x$ and $\left(\varepsilon u_y\right)_y$ at $\left(x_i,y_j,t^n\right)$, respectively.

Spatial discretization

Since the discussion here doesn't concern with time, we will drop t or the superscript n for simplicity. At an interior point $\left(x_i,y_j\right)$, a standard central finite difference scheme is adopted. Namely,

$$(\varepsilon u_x)_{x,i,j} = \frac{\varepsilon_{i-1/2,j}u_{i-1,j} - (\varepsilon_{i-1/2,j} + \varepsilon_{i+1/2,j})u_{i,j} + \varepsilon_{i+1/2,j}u_{i+1,j}}{h^2} + O(h^2),$$

$$(\varepsilon u_y)_{y,i,j} = \frac{\varepsilon_{i,j-1/2}u_{i,j-1} - (\varepsilon_{i,j-1/2} + \varepsilon_{i,j+1/2})u_{i,j} + \varepsilon_{i,j+1/2}u_{i,j+1}}{h^2} + O(h^2).$$

At an on-front grid point (x_i, y_j), we study the following two kinds of jump conditions to compute $(\varepsilon u_x)_{x,i,j}$ and $(\varepsilon u_y)_{y,i,j}$.

JC 1.

At a normal on-front gird point (x_i, y_j), we use the CIM2 to approximate $(\varepsilon u_x)_{x,i,j}$ and $(\varepsilon u_y)_{y,i,j}$. At an exceptional on-front gird point (x_i, y_j), we use the CIM1 to approximate $(\varepsilon u_x)_{x,i,j}$ and $(\varepsilon u_y)_{y,i,j}$.

JC 2.

In this case, we can't directly apply the CIM since we don't know the value of $[\varepsilon u_n]$. So, we follow the idea of JC 2 in chapter 2 (1-D moving interface problems). First we consider the x-direction; we separate it into three parts.

(A) $\gamma_{i+1/2,j} + \gamma_{i-1/2,j} = 2$

Denote the interface point in $[x_{i-1}, x_i) \times \{y_j\}$ and $[x_i, x_{i+1}) \times \{y_j\}$ by $p_- = (x_{p_-}, y_{p_-})$ and $p_+ = (x_{p_+}, y_{p_+})$, respectively. Then

$$(u_x)_{i\pm1/2,j} = \frac{u_{p_\pm} - u_{i,j}}{x_{p_\pm} - x_i} + O(h),$$

and we approximate $(\varepsilon u_x)_{x,i,j}$ by

$$(\varepsilon u_x)_{x,i,j} = \frac{\varepsilon_{i,j}}{h}\left((u_x)_{i+1/2,j} - (u_x)_{i-1/2,j}\right) + O(1).$$

(B) $\gamma_{i+1/2,j} + \gamma_{i-1/2,j} = 1$

Given (x_1, y), (x_2, y), (x_3, y) and (x, y) four points. Define

$$D_x^2 u\big(x_1, x_2, x_3, (x, y)\big) \triangleq d_{1,x} u(x_1, y) + d_{2,x} u(x_2, y) + d_{3,x} u(x_3, y)$$

where

$$d_{1,x} = \frac{2}{(x_1 - x_2)(x_1 - x_3)} \ , d_{2,x} = \frac{2}{(x_2 - x_1)(x_2 - x_3)} \ , d_{3,x} = \frac{2}{(x_3 - x_1)(x_3 - x_2)}, (90)$$

Then

$$(u_{xx})_{i,j} = D_x^2 u\left(x_{i-s_x}, x_i, x_p, \left(x_i, y_j\right)\right) + O(h),$$

(Since the interface is fixed, we don't need to make any adjustment like we did in chapter 2), where x_p is the x-component of the interface point p and we approximate $(\varepsilon u_x)_{x,i,j}$ by

$$(\varepsilon u_x)_{x,i,j} = \frac{1}{h^2}\left(D_{x,(s_x)} \varepsilon_{i,j} D_{x,(s_x)} u_{i,j} + \varepsilon_{i,j} h^2 (u_{xx})_{i,j}\right) + O(h).$$

(C) $\gamma_{i+1/2,j} + \gamma_{i-1/2,j} = 0$

We approximate $(\varepsilon u_x)_{x,i,j}$ by

$$(\varepsilon u_x)_{x,i,j} = \frac{\varepsilon_{i-1/2,j} u_{i-1,j} - \left(\varepsilon_{i-1/2,j} + \varepsilon_{i+1/2,j}\right) u_{i,j} + \varepsilon_{i+1/2,j} u_{i+1,j}}{h^2} + O(h^2).$$

A parallel argument applies to the y-direction.

Numerical examples

In the test problems below, the computational domain is $\Omega = [-1,1] \times [-1,1]$, and the interface Γ is the circle $x^2 + y^2 = (1/2)^2$.

JC 1.

Example 1. (update from [5])

$$\varepsilon(x,y,t)=\begin{cases}(1+r^2)e^{(-t)} & \text{if } r\le \frac{1}{2},\\ 10e^{(-t)} & \text{if } r\ge \frac{1}{2}.\end{cases}$$

$$f(x,y,t)=\begin{cases} r^2e^{(-t)}+(8r^2+4)e^{(-2t)} & \text{if } r\le\frac{1}{2},\\ \left(\frac{\left(r^4/2+r^2+0.1\log(2r)\right)}{10}-\frac{\left(0.5^4/2+0.5^2\right)}{10}+0.5^2\right)e^{(-t)}+(8r^2+4)e^{(-2t)} & \text{if } r\ge\frac{1}{2}.\end{cases}$$

where $r=\sqrt{x^2+y^2}$.

The jump conditions are chosen from the following exact solution:

$$u(x,y,t)=\begin{cases} r^2e^{(-t)} & \text{if } r\le\frac{1}{2},\\ \left(\frac{\left(r^4/2+r^2+0.1\log(2r)\right)}{10}-\frac{\left(0.5^4/2+0.5^2\right)}{10}+0.5^2\right)e^{(-t)} & \text{if } r\ge\frac{1}{2}.\end{cases}$$

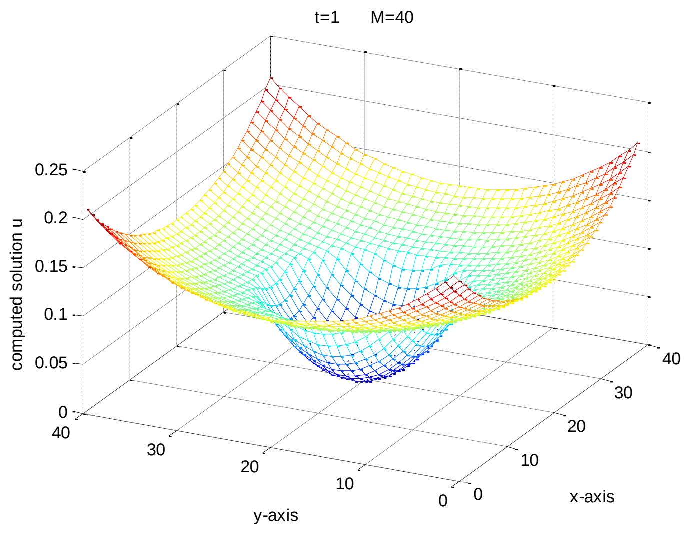

Figure 14. The computed solution $u(x,y,t)$ at $t=1$ with $M=40$.

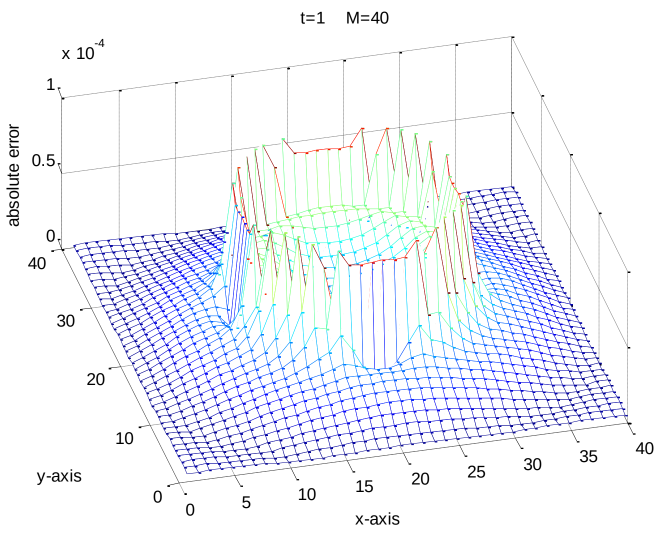

Figure 15. The absolute error at $t=1$ and $M=40$.

Table 10. Numerical results for example 1.

	t=1	
M	Max-norm error	order
20	5.588×10^{-4}	
40	9.884×10^{-5}	2.499
80	1.899×10^{-5}	2.380
160	4.203×10^{-6}	2.176

Example 2.

$$\varepsilon(x,y,t) = \begin{cases} e^{(xy)}\sin(t) & \text{if } r \le \frac{1}{2}, \\ (x^2+y^2)\cos(t) & \text{if } r \ge \frac{1}{2}. \end{cases}$$

$$f(x,y,t) = \begin{cases} -10\sin^2(t)e^{(xy)}\big(\sin(10xy) + 10\cos(10xy)\big)(x^2+y^2) - \cos(10xy)\cos(t) & \text{if } r \le \frac{1}{2}, \\ 40xy\cos^2(t)\cos(10xy) - 100(x^2+y^2)^2\cos^2(t)\sin(10xy) + \sin(10xy)\sin(t) & \text{if } r \ge \frac{1}{2}. \end{cases}$$

The jump conditions are chosen from the following exact solution:

$$u(x,y,t) = \begin{cases} \cos(10xy)\sin(t) & \text{if } r \le \frac{1}{2}, \\ \sin(10xy)\cos(t) & \text{if } r \ge \frac{1}{2}. \end{cases}$$

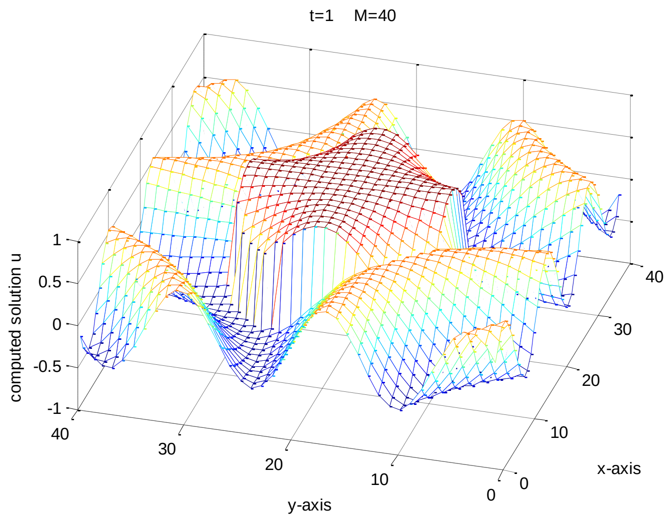

Figure 16. The computed solution $u(x,y,t)$ at $t = 1$ with $M = 40$.

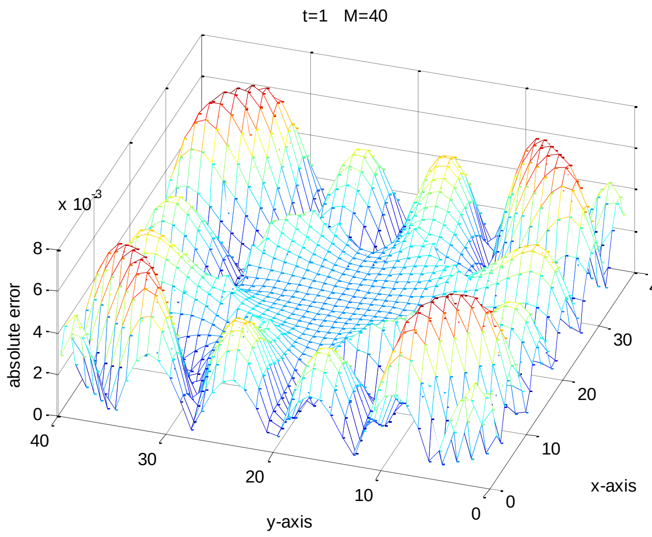

Figure 17. The absolute error at $t = 1$ and $M = 40$.

Table 11. Numerical results for example 2.

	t=1	
M	Max-norm error	order
20	3.382×10^{-2}	
40	7.161×10^{-3}	2.240
80	1.828×10^{-3}	1.970
160	4.652×10^{-4}	1.974

JC 2.

Example 1.(update from [5])

We retest example 1 in JC 1 and use the interface condition

$$u(r = 1/2, t) = \frac{1}{4}e^{(-t)}.$$

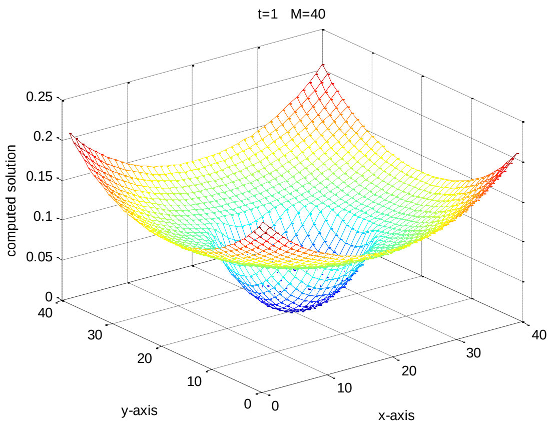

Figure 18. The computed solution $u(x, y, t)$ at $t = 1$ with $M = 40$.

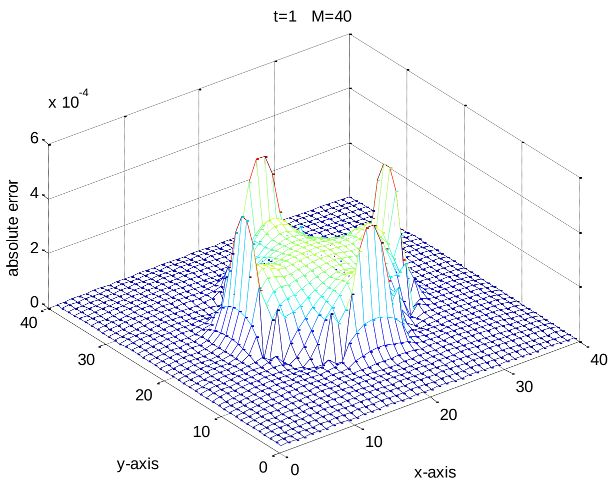

Figure 19. The absolute error at $t = 1$ and $M = 40$.

Table 12. Numerical results for example 1.

	$t = 1$	
M	Max-norm error	order
20	1.386×10^{-3}	
40	4.729×10^{-4}	1.551
80	1.352×10^{-4}	1.806
160	4.030×10^{-5}	1.746

Example 2.

$\varepsilon(x, y, t) = 1.$

$$f(x,y,t) = \begin{cases} \sin\left(\frac{3}{4}\pi\right)\sin(t) & \text{if } r \le \frac{1}{2}, \\ 8\pi\left(-2\pi r^2 \sin\left(2\pi\left(r^2+\frac{1}{8}\right)\right) + \cos\left(2\pi\left(r^2+\frac{1}{8}\right)\right)\right)\cos(t) + \sin\left(2\pi\left(r^2+\frac{1}{8}\right)\right)\sin(t) & \text{if } r \ge \frac{1}{2}. \end{cases}$$

The interface condition is chosen from the following exact solution:

$$u(x,y,t) = \begin{cases} \sin\left(\frac{3}{4}\pi\right)\cos(t) & \text{if } r \le \frac{1}{2}, \\ \sin\left(2\pi\left(r^2+\frac{1}{8}\right)\right)\cos(t) & \text{if } r \ge \frac{1}{2}, \end{cases}$$

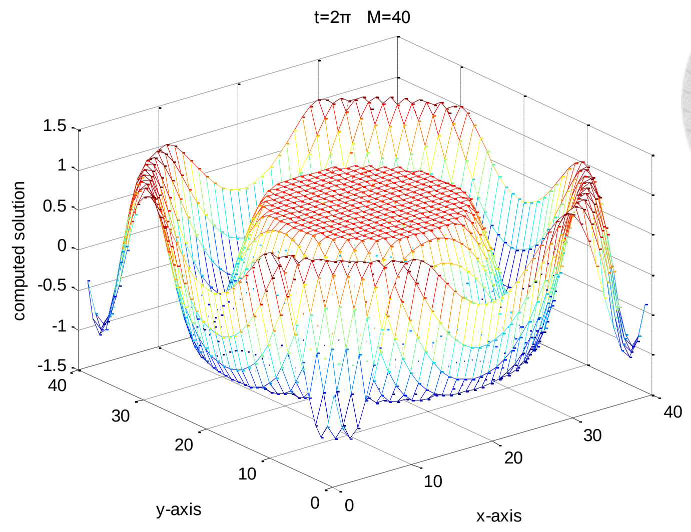

Figure 20. The computed solution $u(x, y, t)$ at $t = 2\pi$ with $M = 40$.

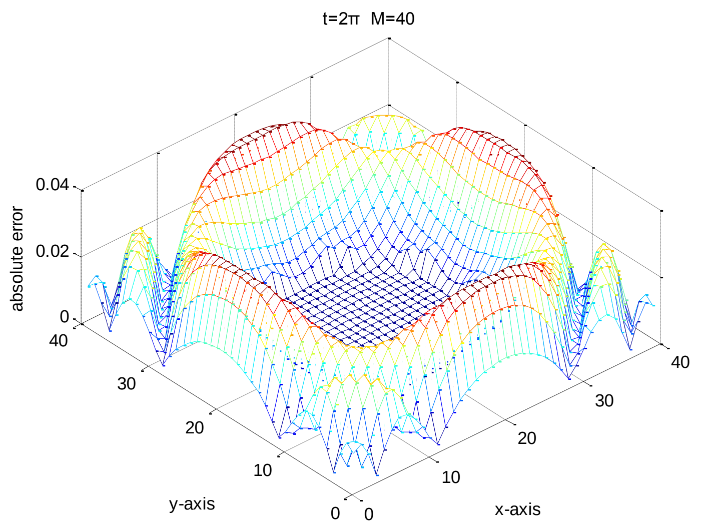

Figure 21. The absolute error at $t = 2\pi$ and $M = 40$.

Table 13. Numerical results for example 2.

	$t = 2\pi$	
M	Max-norm error	order
20	1.440×10^{-1}	
40	3.643×10^{-2}	1.983
80	9.372×10^{-3}	1.959
160	2.382×10^{-3}	1.976

Combining CIM with ADI method (Alternating Direction Implicit method)

The classical ADI method for the problem $u_t = \nabla \cdot (\varepsilon \nabla u) - f(x, y, t)$ is

$$\frac{u_{i,j}^{n+1/2} - u_{i,j}^{n}}{\Delta t/2} = (\varepsilon u_x)_{x,i,j}^{n+1/2} + \left(\varepsilon u_y\right)_{y,i,j}^{n} - \frac{1}{2}\left(f_{i.j}^{n} + f_{i,j}^{n+1}\right),$$

$$\frac{u_{i,j}^{n+1} - u_{i,j}^{n+1/2}}{\Delta t/2} = (\varepsilon u_x)_{x,i,j}^{n+1/2} + \left(\varepsilon u_y\right)_{y,i,j}^{n+1} - \frac{1}{2}\left(f_{i.j}^{n} + f_{i,j}^{n+1}\right).$$

In other word, a single multidimensional implicit time step is replaced by a sequence of steps, each of which is implicit in only one coordinate direction. In addition, the equations can be solved along one line of grid points at a time, which can be solved easily. If there is no interface, this gives decoupled tri-diagonal systems to solve in each step:

$$\left(u_{i,j}^{n+1/2} - \frac{\Delta t}{2}(\varepsilon u_x)_{x,i,j}^{n+1/2}\right) = \left(u_{i,j}^{n} + \frac{\Delta t}{2}\left(\varepsilon u_y\right)_{y,i,j}^{n}\right) - \frac{\Delta t}{2}\frac{1}{2}\left(f_{i.j}^{n} + f_{i,j}^{n+1}\right),$$

$$\left(u_{i,j}^{n+1} - \frac{\Delta t}{2}\left(\varepsilon u_y\right)_{y,i,j}^{n+1}\right) = \left(u_{i,j}^{n+1/2} + \frac{\Delta t}{2}(\varepsilon u_x)_{x,i,j}^{n+1/2}\right) - \frac{\Delta t}{2}\frac{1}{2}\left(f_{i.j}^{n} + f_{i,j}^{n+1}\right).$$

With this method, each of the two steps can be shown to give a first order approximation to the full heat equation over time $\Delta t/2$, so that $u_{i,j}^{n+1/2}$represents a first order approximation to the solution at time $t^{n+1/2}$. Because of the symmetry of the two steps, the local error in the second step almost cancels the local error in the first steps, so that the combine method is second order over full time step. Because $u_{i,j}^{n+1/2}$ does approximate the solution at time $t^{n+1/2}$, it's possible to simply evaluate the given boundary conditions at time $t^{n+1/2}$. First, without any modification, we apply the ADI method to our problem.

JC 1.

Unfortunately, ADI method cannot be applied when $\varepsilon(x,y,t)$ has jump across the interface. This is because when forming the linear system at time level $t^{n+1/2}$ and t^{n+1}, we need to compute the coefficients used to approximate $(u_{xx})_{i,j}^{n+1/2}$ and $\left(u_{yy}\right)_{i,j}^{n+1}$ at normal on-front point (exceptional point is similar). When computing this, we need to compute the coefficients of $b_p^{n+1/2}T_x u_{i,j}^{n+1/2}$ and $b_q^{n+1}T_y u_{i,j}^{n+1}$. From this, we find that $b_p^{n+1/2}$ and b_q^{n+1} must both be zero and consequently $\varepsilon(x,y,t)$ is required to be continuous across the interface. Assuming $\varepsilon(x,y,t)$ is continuous across the interface, we redefine the normal on-front grid points to be the on-front grid points satisfy condition (a) and (b) in p.52~53 and the formulae in computing u_{xx} and u_{yy} by CIM can be simplified as follows:

$$(u_{xx})_{i,j} = \begin{cases} \dfrac{\varepsilon_{i-1/2,j}u_{i-1,j}-(\varepsilon_{i-1/2,j}+\varepsilon_{i+1/2,j})u_{i,j}+\varepsilon_{i+1/2,j}u_{i+1,j}}{h^2} + O(h^2) & \text{if } \gamma_{i+1/2,j}+\gamma_{i-1/2,j}=0\,, \\ \dfrac{1}{h^2}\left(L_x^{(s_x)}u_{i,j}+J_p\right)+O\left(h^{2-|s_x|}\right) & \text{if } \begin{cases}\gamma_{i+1/2,j}+\gamma_{i-1/2,j}=1 \text{ and} \\ \gamma_{i\pm3/2,j}=0 \text{ if } s_{x,(i,j)}=\pm1\end{cases}, \\ \dfrac{1}{h}\left((u_x)_{i+1/2,j}-(u_x)_{i-1/2,j}\right)+O(1) & \text{else}\,. \end{cases}$$

$$\left(u_{yy}\right)_{i,j} = \begin{cases} \dfrac{\varepsilon_{i,j-1/2}u_{i,j-1}-(\varepsilon_{i,j-1/2}+\varepsilon_{i,j+1/2})u_{i,j}+\varepsilon_{i,j+1/2}u_{i,j+1}}{h^2} + O(h^2) & \text{if } \gamma_{i,j+1/2}+\gamma_{i,j-1/2}=0, \\ \dfrac{1}{h^2}\left(L_y^{(s_y)}u_{i,j}+J_q\right)+O\left(h^{2-|s_y|}\right) & \text{if } \begin{cases}\gamma_{i,j+1/2}+\gamma_{i,j-1/2}=1 \text{ and} \\ \gamma_{i,j\pm3/2}=0 \text{ if } s_{y,(i,j)}=\pm1\end{cases}, \\ \dfrac{1}{h}\left(\left(u_y\right)_{i,j+1/2}-\left(u_y\right)_{i,j-1/2}\right)+O(1) & \text{else.} \end{cases}$$

where

$$J_p = -\left(|s_x|(1+2\beta_p)\rho_p[u]_p + s_x(\beta_p+\beta_p^2)h\left(\frac{[\varepsilon u_n]_p}{\widehat{\varepsilon_p}} n_p^x + \rho_p[u_t]_p t_p^x\right)\right),$$

$$J_q = -\left(|s_y|(1+2\beta_q)\rho_q[u]_q + s_y(\beta_q+\beta_q^2)h\left(\frac{[\varepsilon u_n]_q}{\widehat{\varepsilon_q}} n_q^y + \rho_q[u_t]_q t_q^y\right)\right),$$

$$L_x^{(s_x)}u_{i,j} = \begin{cases} a_{p,-s_x}u_{i-s_x,j} + a_{p,0}u_{i,j} + a_{p,s_x}u_{i+s_x,j} + a_{p,2s_x}u_{i+2s_x,j} & \text{if } s_x = \pm 1, \\ u_{i-1,j} - 2u_{i,j} + u_{i+1,j} & \text{if } s_x = 0. \end{cases}$$

$$L_y^{(s_y)}u_{i,j} = \begin{cases} a_{q,-s_y}u_{i,j-s_y} + a_{q,0}u_{i,j} + a_{q,s_y}u_{i,j+s_y} + a_{q,2s_y}u_{i,j+2s_y} & \text{if } s_y = \pm 1, \\ u_{i,j-1} - 2u_{i,j} + u_{i,j+1} & \text{if } s_y = 0. \end{cases}$$

with the corresponding simplified coefficients:

$$\widehat{\varepsilon_p} \triangleq \left((\beta_p+\beta_p^2)\left(\frac{1}{2}+\alpha_p\right) + (\alpha_p+\alpha_p^2)\left(\frac{1}{2}+\beta_p\right)\right)\varepsilon_p,$$

$$\rho_p \triangleq \frac{\varepsilon_p}{\widehat{\varepsilon_p}} = \frac{1}{(\beta_p+\beta_p^2)\left(\frac{1}{2}+\alpha_p\right) + (\alpha_p+\alpha_p^2)\left(\frac{1}{2}+\beta_p\right)},$$

$$\begin{cases} a_{p,-s_x} \triangleq \left((\beta_p+\beta_p^2) + \alpha_p(1+2\beta_p)\right)\rho_p, \\ a_{p,0} \triangleq \left(-(\beta_p+\beta_p^2) - (1+\alpha_p)(1+2\beta_p)\right)\rho_p, \\ a_{p,s_x} \triangleq (1+\beta_p)^2\rho_p, \\ a_{p,2s_x} \triangleq -\beta_p^2\rho_p. \end{cases}$$

(Replace x by y and p by q we get the coefficients for y-direction)

$$(u_x)_{i\pm1/2,j} = \frac{1}{h}\left(\overline{D}_x^{(\pm1/2)}u_{i,j} + \gamma_{i\pm1/2,j}\overline{J}_p^{(\pm1/2)}\right) + O(h),$$

$$(u_y)_{i,j\pm1/2} = \frac{1}{h}\left(\overline{D}_y^{(\pm1/2)}u_{i,j} + \gamma_{i,j\pm1/2}\overline{J}_q^{(\pm1/2)}\right) + O(h),$$

where

$$\overline{J}_p^{(\pm1/2)} = \mp[u]_p - \beta_p h\left(\frac{[\varepsilon u_n]_p}{\varepsilon_p} n_p^x + [u_t]_p t_p^x\right),$$

$$\overline{J}_q^{(\pm1/2)} = \mp[u]_q - \beta_q h\left(\frac{[\varepsilon u_n]_q}{\varepsilon_q} n_q^y + [u_t]_q t_q^y\right).$$

JC 2.

In this case, no restriction on $\varepsilon(x,y,t)$. The formula in computing u_{xx} and u_{yy} by

CIM at grid point is

$$D_{xx}u_{i,j} = \begin{cases} \dfrac{u_{i-1,j} - 2u_{i,j} + u_{i+1,j}}{h^2} + O(h^2) & \text{if } \gamma_{i+1/2,j} + \gamma_{i-1/2,j} = 0, \\ D_x^2 u\left(x_{i-s_x}, x_i, x_p, (x_i, y_j)\right) + O(h) & \text{if } \gamma_{i+1/2,j} + \gamma_{i-1/2,j} = 1, \\ \dfrac{1}{h}\left((u_x)_{i+1/2,j} - (u_x)_{i-1/2,j}\right) + O(1) & \text{if } \gamma_{i+1/2,j} + \gamma_{i-1/2,j} = 2. \end{cases}$$

$$D_{yy}u_{i,j} = \begin{cases} \dfrac{u_{i,j-1} - 2u_{i,j} + u_{i,j+1}}{h^2} + O(h^2) & \text{if } \gamma_{i,j+1/2} + \gamma_{i,j-1/2} = 0, \\ D_y^2 u\left(y_{j-s_y}, y_j, y_q, (x_i, y_j)\right) + O(h) & \text{if } \gamma_{i,j+1/2} + \gamma_{i,j-1/2} = 1, \\ \dfrac{1}{h}\left((u_y)_{i,j+1/2} - (u_y)_{i,j-1/2}\right) + O(1) & \text{if } \gamma_{i,j+1/2} + \gamma_{i,j-1/2} = 2. \end{cases}$$

where

$$(u_x)_{i\pm1/2,j} = \frac{u_{p_\pm} - u_{i,j}}{x_{p_\pm} - x_i} + O(h).$$

Here $p_- = (x_{p_-}, y_{p_-})$ and $p_+ = (x_{p_+}, y_{p_+})$ are the interface point in $[x_{i-1}, x_i) \times \{y_j\}$ and $[x_i, x_{i+1}) \times \{y_j\}$, respectively. (A parallel argument applies to the y-direction)

Numerical examples

JC 1.

Example 1.[12]

$\varepsilon(x, y, t) = 1.$

$$f(x,y,t) = \begin{cases} 0 & \text{if } r \le \frac{1}{2}, \\ \left(\sin(t) - \frac{\pi^2}{8}\cos(t)\right)\sin\left(\frac{\pi}{4}(x+1)\right)\sin\left(\frac{\pi}{4}(y+1)\right) & \text{if } r \ge \frac{1}{2}. \end{cases}$$

The jump conditions are chosen from the following exact solution:

$$u(x,y,t) = \begin{cases} 1 & \text{if } r \le \frac{1}{2}, \\ \cos(t)\sin\left(\frac{\pi}{4}(x+1)\right)\sin\left(\frac{\pi}{4}(y+1)\right) & \text{if } r \ge \frac{1}{2}. \end{cases}$$

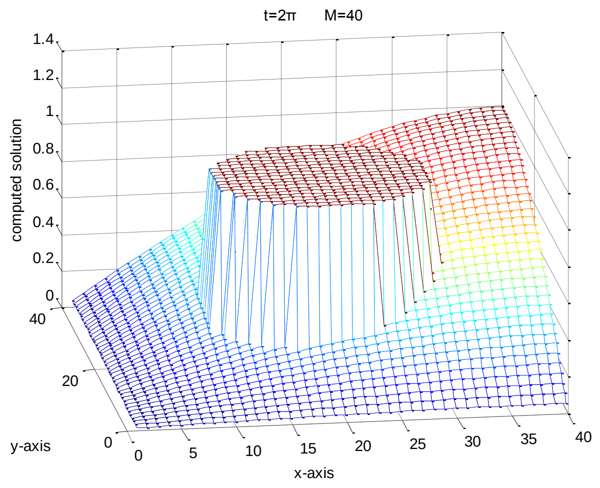

Figure 22. The computed solution $u(x,y,t)$ at $t = 2\pi$ with $M = 40$.

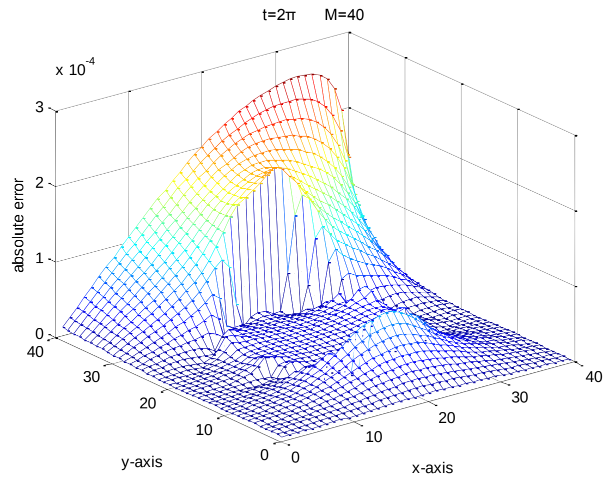

Figure 23. The absolute error at $t = 2\pi$ and $M = 40$.

Table 14. Numerical results for example 1.

	$t=2\pi$	
M	Max-norm error	order
20	8.853×10^{-4}	
40	2.589×10^{-4}	1.774
80	6.856×10^{-5}	1.917
160	1.831×10^{-5}	1.905
320	4.674×10^{-6}	1.970

Example 2.

$\varepsilon(x, y, t) = 1.$

$$f(x,y,t) = \begin{cases} (16 + 4(x^2 + y^2))e^{(-t)} & \text{if } r \le \frac{1}{2}, \\ \left(1 - (x^2 + y^2)\right)\sin(xy)e^{(-t)} & \text{if } r \ge \frac{1}{2}. \end{cases}$$

The jump conditions are chosen from the following exact solution:

$$u(x,y,t) = \begin{cases} 4(x^2 + y^2)e^{(-t)} & \text{if } r \le \frac{1}{2}, \\ \sin(xy)e^{(-t)} & \text{if } r \ge \frac{1}{2}. \end{cases}$$

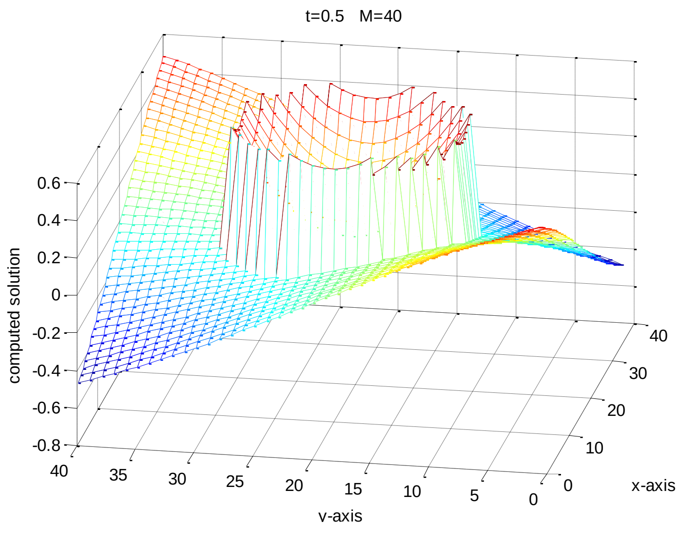

Figure 24. The computed solution $u(x, y, t)$ at $t = 0.5$ with $M = 40$.

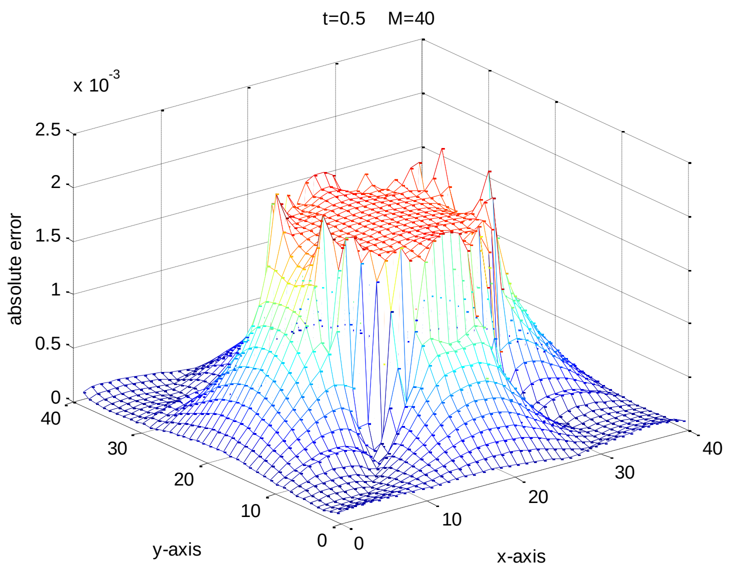

Figure 25. The absolute error at $t = 0.5$ and $M = 40$.

Table 15. Numerical results for example 2.

	t=0.5	
M	Max-norm error	order
20	8.189×10^{-3}	
40	2.162×10^{-3}	1.921
80	5.890×10^{-4}	1.876
160	1.665×10^{-4}	1.823
320	3.804×10^{-5}	2.130

JC 2.

Example 1. (update from [5])

We retest example1 JC 2 in C-N case.

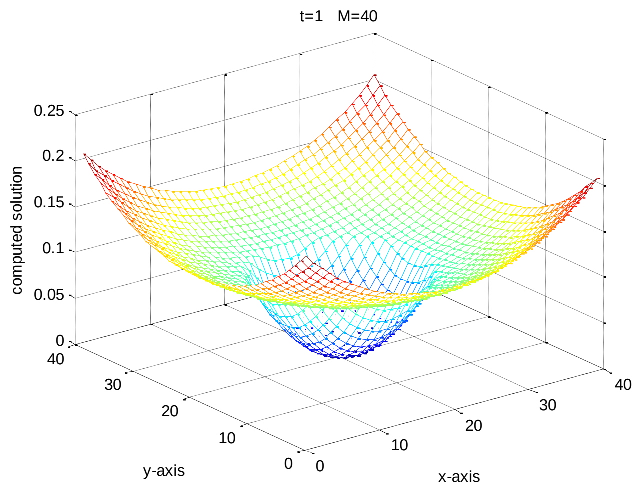

Figure 26. The computed solution $u(x, y, t)$ at $t = 1$ with $M = 40$.

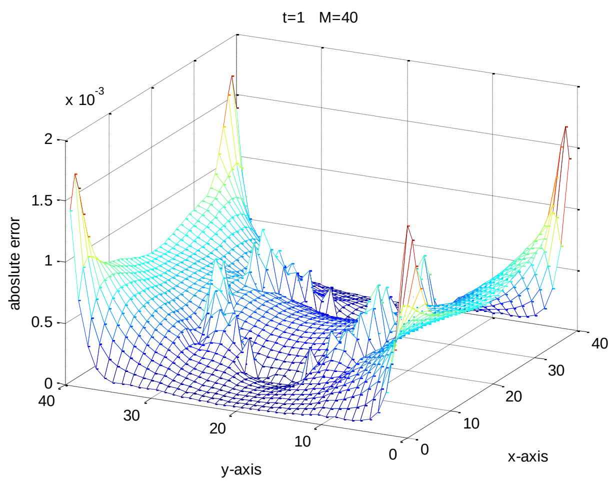

Figure 27. The absolute error at $t = 1$ and $M = 40$.

Table 16. Numerical results for example 1.

	t=1	
M	Max-norm error	order
20	5.647×10^{-3}	
40	1.683×10^{-3}	1.746
80	4.612×10^{-4}	1.868
160	1.208×10^{-4}	1.933
320	3.092×10^{-5}	1.966

Example 2.

We retest example 2 JC 2 in C-N case.

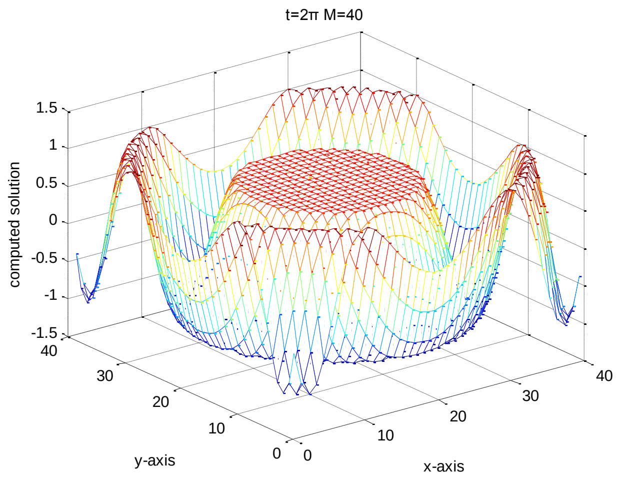

Figure 28. The computed solution $u(x, y, t)$ at $t = 2\pi$ with $M = 40$.

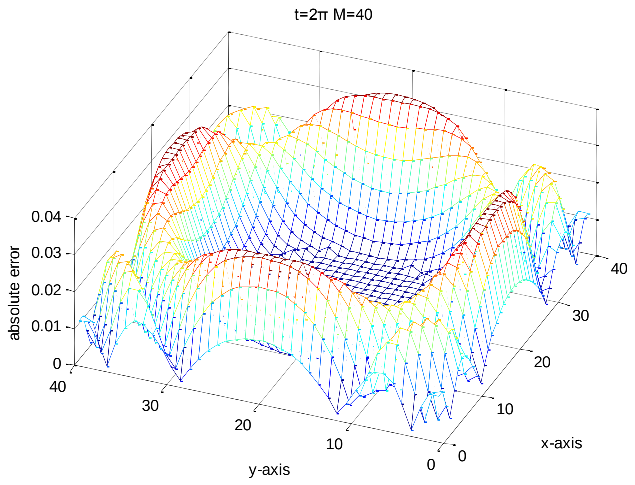

Figure 29. The absolute error at $t = 2\pi$ and $M = 40$.

Table 17. Numerical results for example 2.

	$t = 2\pi$	
M	Max-norm error	order
20	1.383×10^{-1}	
40	3.528×10^{-2}	1.971
80	9.071×10^{-3}	1.960
160	2.294×10^{-3}	1.983
320	5.769×10^{-4}	1.992

Numerical tests give promising results and the method appears to be second order accurate. However, from local truncation error table for interior points and normal on-front grid points (see table 18 and 19)

Table 18. Local truncation error for JC 1.

	Example1 $t = \pi/2$		Example2 $t = 0.25$	
M	Max-norm LTE error	order	Max-norm LTE error	order
20	0.221		4.730	
40	0.206	0.101	4.641	0.027
80	0.259	-0.330	6.104	-0.395
160	0.244	0.086	5.676	0.105
320	0.258	-0.081	5.887	-0.053

Table 19. Local truncation error for JC 2.

	Example1 $t = 1$		Example2 $t = \pi$	
M	Max-norm LTE error	order	Max-norm LTE error	order
20	3.778		16.529	
40	4.283	-0.166	6.644	1.315
80	130.686	-4.947	21.481	-1.693
160	18.515	2.819	3.753	2.517
320	47.997	-1.374	6.084	-0.697

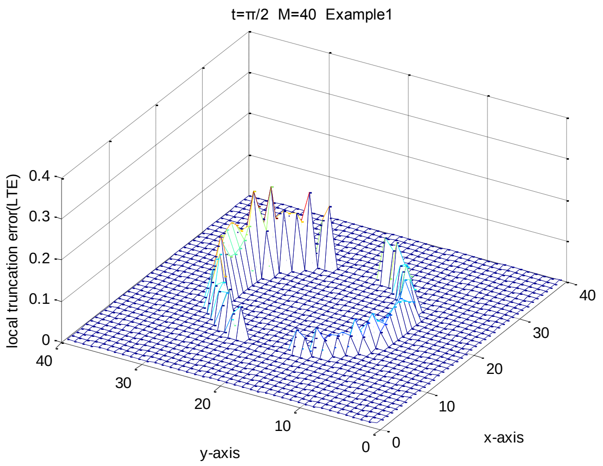

Figure 30. Local truncation error at t= $\pi/2$ M=40 (example 1 JC 1).

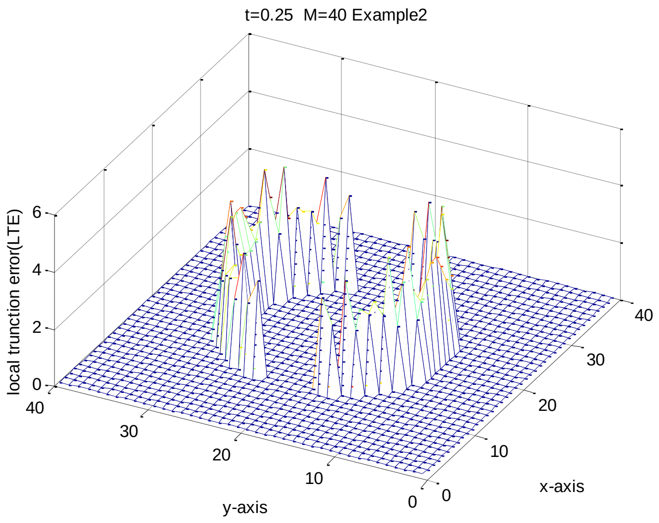

Figure 31. Local truncation error at t=0.25 M=40 (example 2 JC 1).

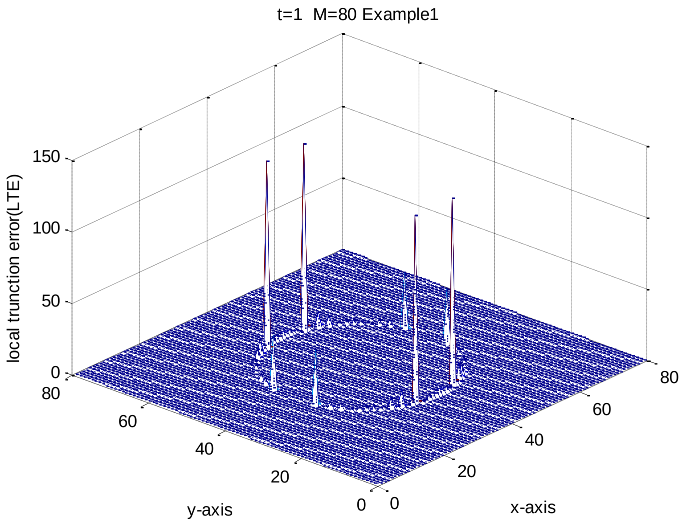

Figure 32. Local truncation error at t=1 M=80 (example 1 JC 2).

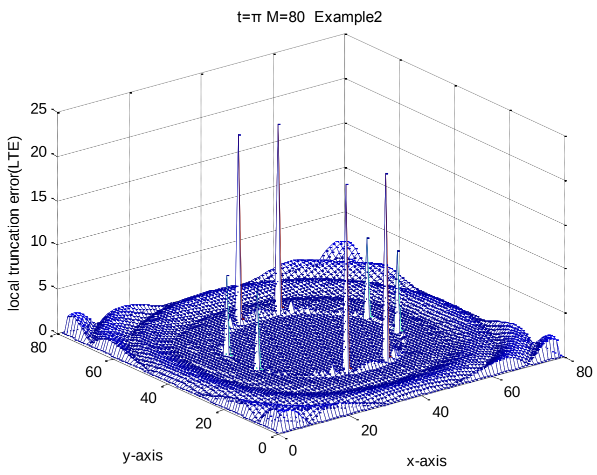

Figure 33. Local truncation error at t=π M=80 (example 2 JC 2).

We find out that the local truncation error at these points is only $O(1)$. So there must be some cancellation in the errors. We are not sure whether such cancellation will always occur. For safety, we are going to do the local truncation error analysis and modify the ADI scheme by adding correction terms so that the local truncation errors at normal on-front points are $O(h)$ so we are guaranteed to get a second-order accurate solution.

Local truncation error analysis

For simplicity, assume $\varepsilon = 1$.

JC 1.

We define $D_{xx}u_{i,j}$ and $D_{yy}u_{i,j}$ to be the approximations of $(u_{xx})_{i,j}$ and $\left(u_{yy}\right)_{i,j}$ by CIM, respectively.

The difference scheme is

$$\frac{u_{i,j}^{n+1/2} - u_{i,j}^{n}}{\Delta t/2} = D_{xx}^{n+1/2}u_{i,j}^{n+1/2} + D_{yy}^{n}u_{i,j}^{n} - \frac{1}{2}\left(f_{i.j}^{n} + f_{i,j}^{n+1}\right), \tag{91}$$

$$\frac{u_{i,j}^{n+1} - u_{i,j}^{n+1/2}}{\Delta t/2} = D_{xx}^{n+1/2}u_{i,j}^{n+1/2} + D_{yy}^{n+1}u_{i,j}^{n+1} - \frac{1}{2}\left(f_{i.j}^{n} + f_{i,j}^{n+1}\right). \tag{92}$$

Adding the equations (91) and (92), we get

$$\frac{u_{i,j}^{n+1}-u_{i,j}^{n}}{\Delta t}=D_{xx}^{n+1/2}u_{i,j}^{n+1/2}+\frac{1}{2}\left(D_{yy}^{n}u_{i,j}^{n}+D_{yy}^{n+1}u_{i,j}^{n+1}\right)-\frac{1}{2}\left(f_{i.j}^{n}+f_{i,j}^{n+1}\right). \tag{93}$$

Subtracting the equations (91) from (92), we get

$$u_{i,j}^{n+1/2}=\frac{1}{2}\left(u_{i,j}^{n}+u_{i,j}^{n+1}\right)+\frac{\Delta t}{4}\left(D_{yy}^{n}u_{i,j}^{n}-D_{yy}^{n+1}u_{i,j}^{n+1}\right).$$

Substituting this into (93), we get

$$\frac{u_{i,j}^{n+1}-u_{i,j}^{n}}{\Delta t}=D_{xx}^{n+1/2}\left(\frac{1}{2}\left(u_{i,j}^{n}+u_{i,j}^{n+1}\right)+\frac{\Delta t}{4}\left(D_{yy}^{n}u_{i,j}^{n}-D_{yy}^{n+1}u_{i,j}^{n+1}\right)\right)+\frac{1}{2}\left(D_{yy}^{n}u_{i,j}^{n}+D_{yy}^{n+1}u_{i,j}^{n+1}\right)-\frac{1}{2}\left(f_{i.j}^{n}+f_{i,j}^{n+1}\right).$$

This is the difference scheme which we actually use to get the next time solution $u_{i,j}^{n+1}$.

We check each term one by one.

Assume Δt=h.

(A)

$$\frac{u_{i,j}^{n+1}-u_{i,j}^{n}}{\Delta t}=(u_t)_{i,j}^{n+1/2}+O(\Delta t^2).$$

Nothing needs to be modified.

(B)

$$\frac{1}{2}\left(D_{yy}^{n}u_{i,j}^{n}+D_{yy}^{n+1}u_{i,j}^{n+1}\right)$$

$$=\begin{cases}\frac{1}{2}\left(\left(u_{yy}\right)_{i,j}^{n}+\left(u_{yy}\right)_{i,j}^{n+1}\right)+O(h^2) & \text{if } \gamma_{i,j+1/2}+\gamma_{i,j-1/2}=0,\\ \frac{1}{2}\left(\left(u_{yy}\right)_{i,j}^{n}+\left(u_{yy}\right)_{i,j}^{n+1}\right)+O\left(h^{2-|s_y|}\right) & \text{if}\begin{cases}\gamma_{i,j+1/2}+\gamma_{i,j-1/2}=1 \text{ and}\\ \gamma_{i,j\pm 3/2}=0 \text{ if } s_{y,(i,j)}=\pm 1\end{cases},\\ \frac{1}{2}\left(\left(u_{yy}\right)_{i,j}^{n}+\left(u_{yy}\right)_{i,j}^{n+1}\right)+O(1) & \text{else.}\end{cases}$$

$$=\begin{cases}\left(u_{yy}\right)_{i,j}^{n+1/2}+O(h^2) & \text{if } \gamma_{i,j+1/2}+\gamma_{i,j-1/2}=0,\\ \left(u_{yy}\right)_{i,j}^{n+1/2}+O\left(h^{2-|s_y|}\right) & \text{if}\begin{cases}\gamma_{i,j+1/2}+\gamma_{i,j-1/2}=1 \text{ and}\\ \gamma_{i,j\pm 3/2}=0 \text{ if } s_{y,(i,j)}=\pm 1\end{cases},\\ \left(u_{yy}\right)_{i,j}^{n+1/2}+O(1) & \text{else.}\end{cases}$$

Nothing needs to be modified.

(C)

$$\frac{1}{2}\left(f_{i.j}^{n}+f_{i,j}^{n+1}\right)=f_{i,j}^{n+1/2}+O(\Delta t^{2}).$$

Nothing needs to be modified.

(D)

Before we discuss the $D_{xx}^{n+1/2}\left(\frac{1}{2}\left(u_{i,j}^{n}+u_{i,j}^{n+1}\right)+\frac{\Delta t}{4}\left(D_{yy}^{n}u_{i,j}^{n}-D_{yy}^{n+1}u_{i,j}^{n+1}\right)\right)$ term, we give some estimations of $D_{yy}^{n}u_{i,j}^{n}-D_{yy}^{n+1}u_{i,j}^{n+1}$. Since all quantities are continuous in time, we can write

(i) $\gamma_{i,j+1/2}+\gamma_{i,j-1/2}=0$

$$D_{yy}^{n}u_{i,j}^{n}=\left(u_{yy}\right)_{i,j}^{n}+N_{i,j}^{n}h^{2}+O(h^{3}),$$

$$D_{yy}^{n+1}u_{i,j}^{n+1}=\left(u_{yy}\right)_{i,j}^{n+1}+N_{i,j}^{n+1}h^{2}+O(h^{3}).$$

with $N_{i,j}^{n}-N_{i,j}^{n+1}=O(h)$

So

$$D_{yy}^{n}u_{i,j}^{n}-D_{yy}^{n+1}u_{i,j}^{n+1}=\left(u_{yy}\right)_{i,j}^{n}-\left(u_{yy}\right)_{i,j}^{n+1}+O(h^{3}). \qquad (94)(i)$$

(ii) $\begin{cases}\gamma_{i,j+1/2}+\gamma_{i,j-1/2}=1 \text{ and}\\ \gamma_{i,j\pm 3/2}=0 \text{ if } s_{y,(i,j)}=\pm 1\end{cases}$

$$D_{yy}^{n}u_{i,j}^{n}=\left(u_{yy}\right)_{i,j}^{n}+N_{i,j}^{n}h+O(h^{2}),$$

$$D_{yy}^{n+1}u_{i,j}^{n+1}=\left(u_{yy}\right)_{i,j}^{n+1}+N_{i,j}^{n+1}h+O(h^{2}).$$

with $N_{i,j}^{n}-N_{i,j}^{n+1}=O(h)$

So

$$D_{yy}^{n}u_{i,j}^{n} - D_{yy}^{n+1}u_{i,j}^{n+1} = \left(u_{yy}\right)_{i,j}^{n} - \left(u_{yy}\right)_{i,j}^{n+1} + O(h^2). \qquad (94)(ii)$$

(iii) $\gamma_{i,j+1/2} + \gamma_{i,j-1/2} = 0$, $\begin{cases}\gamma_{i,j+1/2} + \gamma_{i,j-1/2} = 1 \text{ and} \\ \gamma_{i,j\pm 3/2} = 0 \text{ if } s_{y,(i,j)} = \pm 1\end{cases}$ both conditions are false

$$D_{yy}^{n}u_{i,j}^{n} = \left(u_{yy}\right)_{i,j}^{n} + N_{i,j}^{n} + O(h),$$

$$D_{yy}^{n+1}u_{i,j}^{n+1} = \left(u_{yy}\right)_{i,j}^{n+1} + N_{i,j}^{n+1} + O(h).$$

with $N_{i,j}^{n} - N_{i,j}^{n+1} = O(h)$

So

$$D_{yy}^{n}u_{i,j}^{n} - D_{yy}^{n+1}u_{i,j}^{n+1} = \left(u_{yy}\right)_{i,j}^{n} - \left(u_{yy}\right)_{i,j}^{n+1} + O(h). \qquad (94)(iii)$$

We separate $D_{xx}^{n+1/2}\left(\frac{1}{2}\left(u_{i,j}^{n} + u_{i,j}^{n+1}\right) + \frac{\Delta t}{4}\left(D_{yy}^{n}u_{i,j}^{n} - D_{yy}^{n+1}u_{i,j}^{n+1}\right)\right)$ term into three cases.

(I) $\gamma_{i+1/2,j} + \gamma_{i-1/2,j} = 0$

$$D_{xx}^{n+1/2}\left(\frac{1}{2}\left(u_{i,j}^{n} + u_{i,j}^{n+1}\right) + \frac{\Delta t}{4}\left(D_{yy}^{n}u_{i,j}^{n} - D_{yy}^{n+1}u_{i,j}^{n+1}\right)\right)$$

$$= \frac{1}{2}\left((u_{xx})_{i,j}^{n} + (u_{xx})_{i,j}^{n+1}\right) + O(h^2) + \frac{\Delta t}{4}D_{xx}^{n+1/2}\left(D_{yy}^{n}u_{i,j}^{n} - D_{yy}^{n+1}u_{i,j}^{n+1}\right)$$

$$= \begin{cases}\frac{1}{2}(u_{xx})_{i,j}^{n+1/2} + +O(h^2) + \frac{\Delta t}{4}D_{xx}^{n+1/2}\left(\left(u_{yy}\right)_{i,j}^{n} - \left(u_{yy}\right)_{i,j}^{n+1}\right) + O(h^2) \ \ (\text{by}(94(\text{i}))) \\ \frac{1}{2}(u_{xx})_{i,j}^{n+1/2} + +O(h^2) + \frac{\Delta t}{4}D_{xx}^{n+1/2}\left(\left(u_{yy}\right)_{i,j}^{n} - \left(u_{yy}\right)_{i,j}^{n+1}\right) + O(h) \ \ (\text{by}(94(\text{ii}))) \\ \frac{1}{2}(u_{xx})_{i,j}^{n+1/2} + +O(h^2) + \frac{\Delta t}{4}D_{xx}^{n+1/2}\left(\left(u_{yy}\right)_{i,j}^{n} - \left(u_{yy}\right)_{i,j}^{n+1}\right) + O(1) \ \ (\text{by}(94(\text{iii})))\end{cases}$$

$$= \begin{cases}\frac{1}{2}(u_{xx})_{i,j}^{n+1/2} + O(h^2) + \frac{\Delta t}{4}D_{xx}^{n+1/2}\left(-\left(u_{yyt}\right)_{i,j}^{n+1/2}\Delta t + O(\Delta t^3)\right) \\ \frac{1}{2}(u_{xx})_{i,j}^{n+1/2} + O(h) + \frac{\Delta t}{4}D_{xx}^{n+1/2}\left(-\left(u_{yyt}\right)_{i,j}^{n+1/2}\Delta t + O(\Delta t^3)\right) \\ \frac{1}{2}(u_{xx})_{i,j}^{n+1/2} + O(1) + \frac{\Delta t}{4}D_{xx}^{n+1/2}\left(-\left(u_{yyt}\right)_{i,j}^{n+1/2}\Delta t + O(\Delta t^3)\right)\end{cases}$$

$$= \begin{cases}\frac{1}{2}(u_{xx})_{i,j}^{n+1/2} + O(h^2) & \text{if (i) holds for } (i,j), (i-1,j) \text{ and } (i+1,j), \\ \frac{1}{2}(u_{xx})_{i,j}^{n+1/2} + O(h) & \text{if (ii) holds for some } (i,j), (i-1,j) \text{ and } (i+1,j), \text{and (i) holds for the rest,} \\ \frac{1}{2}(u_{xx})_{i,j}^{n+1/2} + O(1) & \text{else.}\end{cases}$$

Nothing needs to be modified.

(II) $\gamma_{i+1/2,j}+\gamma_{i-1/2,j}=0$, $\begin{cases}\gamma_{i+1/2,j}+\gamma_{i-1/2,j}=1 \text{ and} \\ \gamma_{i\pm3/2,j}=0 \text{ if } s_{x,(i,j)}=\pm1\end{cases}$ both conditions are false

$$D_{xx}^{n+1/2}\left(\frac{1}{2}\left(u_{i,j}^{n}+u_{i,j}^{n+1}\right)+\frac{\Delta t}{4}\left(D_{yy}^{n}u_{i,j}^{n}-D_{yy}^{n+1}u_{i,j}^{n+1}\right)\right)$$

$$=\frac{1}{2}\left((u_{xx})_{i,j}^{n}+(u_{xx})_{i,j}^{n+1}\right)+O(1)$$

$$=\frac{1}{2}(u_{xx})_{i,j}^{n+1/2}+O(1).$$

Nothing needs to be modified.

(III) $\gamma_{i+1/2,j}+\gamma_{i-1/2,j}=1$ and $\gamma_{i\pm3/2,j}=0$ if $s_{x,(i,j)}=\pm1$

$$D_{xx}^{n+1/2}\left(\frac{1}{2}\left(u_{i,j}^{n}+u_{i,j}^{n+1}\right)+\frac{\Delta t}{4}\left(D_{yy}^{n}u_{i,j}^{n}-D_{yy}^{n+1}u_{i,j}^{n+1}\right)\right)$$

$$=\left(\frac{1}{h^2}L_x^{(s_x),n+1/2}\left(\frac{1}{2}\left(u_{i,j}^{n}+u_{i,j}^{n+1}\right)\right)+J_p^{n+1/2}\right)+\frac{\Delta t}{4}\frac{1}{h^2}L_x^{(s_x),n+1/2}\left(D_{yy}^{n}u_{i,j}^{n}-D_{yy}^{n+1}u_{i,j}^{n+1}\right).$$

We separate it into two parts

1. $\left(\frac{1}{h^2}L_x^{(s_x),n+1/2}\left(\frac{1}{2}\left(u_{i,j}^{n}+u_{i,j}^{n+1}\right)\right)+J_p^{n+1/2}\right)$ term

$$\frac{1}{h^2}L_x^{(s_x),n+1/2}\left(\frac{1}{2}\left(u_{i,j}^{n}+u_{i,j}^{n+1}\right)\right)+J_p^{n+1/2}$$

$$=\frac{1}{2}\frac{1}{h^2}\left(L_x^{(s_x),n+1/2}u_{i,j}^{n}+J_p^{n+1/2}\right)+\frac{1}{2}\frac{1}{h^2}\left(L_x^{(s_x),n+1/2}u_{i,j}^{n+1}+J_p^{n+1/2}\right)$$

$$=\frac{1}{2}\frac{1}{h^2}\left(L_x^{(s_x),n}u_{i,j}^{n}+J_p^{n}+J_p^{n+1/2}-J_p^{n}\right)+\frac{1}{2}\frac{1}{h^2}\left(L_x^{(s_x),n+1}u_{i,j}^{n+1}+J_p^{n+1}+J_p^{n+1/2}-J_p^{n+1}\right)$$

$$\left(\begin{array}{l}\text{Here we use the fact that } L_x^{(s_x),n+1/2}=L_x^{(s_x),n}=L_x^{(s_x),n+1}\text{since} \\ \text{the coefficients } a_{p,-s_x},\ a_{p,0}, a_{p,s_x}\text{and } a_{p,2s_x} \text{ only depend on } \alpha_p\text{and } \beta_p\end{array}\right)$$

$$=\frac{1}{2}(u_{xx})_{i,j}^{n}+\frac{1}{2}(u_{xx})_{i,j}^{n+1}+\frac{1}{2}\frac{1}{h^2}\left(2J_p^{n+1/2}-J_p^{n}-J_p^{n+1}\right)+O(h)$$

$$=(u_{xx})_{i,j}^{n+1/2}+O(\Delta t^2)+\frac{1}{h^2}\left(O(\Delta t^2)\right)+O(h)$$

$$=(u_{xx})_{i,j}^{n+1/2}+O(1).$$

We need to modify this term.

2. $\frac{\Delta t}{4}\frac{1}{h^2}L_x^{(s_x),n+1/2}\left(D_{yy}^n u_{i,j}^n - D_{yy}^{n+1}u_{i,j}^{n+1}\right)$ term

(a) either (i) or (ii) holds for (i,j), (i-1,j) and (i+1,j)

$$\frac{\Delta t}{4}\frac{1}{h^2}L_x^{(s_x),n+1/2}\left(D_{yy}^n u_{i,j}^n - D_{yy}^{n+1}u_{i,j}^{n+1}\right)$$
$$= \frac{\Delta t}{4}\frac{1}{h^2}L_x^{(s_x),n+1/2}\left(\left(u_{yy}\right)_{i,j}^n - \left(u_{yy}\right)_{i,j}^{n+1}\right) + O(h) \qquad (\text{by}(94(\text{i})(\text{ii})))$$
$$= \frac{\Delta t}{4}\frac{1}{h^2}L_x^{(s_x),n+1/2}\left(-\left(u_{yyt}\right)_{i,j}^{n+1/2}\Delta t + O(\Delta t^3)\right) + O(h)$$
$$= -\frac{1}{4}L_x^{(s_x),n+1/2}\left(u_{yyt}\right)_{i,j}^{n+1/2} + O(h)$$
$$= O(1).$$

We need to modify this term.

(b) else

$$\frac{\Delta t}{4}\frac{1}{h^2}L_x^{(s_x),n+1/2}\left(D_{yy}^n u_{i,j}^n - D_{yy}^{n+1}u_{i,j}^{n+1}\right)$$
$$= \frac{\Delta t}{4}\frac{1}{h^2}L_x^{(s_x),n+1/2}\left(\left(u_{yy}\right)_{i,j}^n - \left(u_{yy}\right)_{i,j}^{n+1}\right) + O(1) \qquad (\text{by}(94(\text{iii})))$$
$$= \frac{\Delta t}{4}\frac{1}{h^2}L_x^{(s_x),n+1/2}\left(-\left(u_{yyt}\right)_{i,j}^{n+1/2}\Delta t + O(\Delta t^3)\right) + O(1)$$
$$= O(1).$$

Nothing needs to be modified.

So we add two correction terms $(C_x)_{i,j}^n$ and $\left(C_y\right)_{i,j}^n$ to the difference scheme:

$$\frac{u_{i,j}^{n+1/2} - u_{i,j}^n}{\Delta t/2} = D_{xx}^{n+1/2}u_{i,j}^{n+1/2} + D_{yy}^n u_{i,j}^n - \frac{1}{2}\left(f_{i.j}^n + f_{i,j}^{n+1}\right) - (C_x)_{i,j}^n - \left(C_y\right)_{i,j}^n,$$
$$\frac{u_{i,j}^{n+1} - u_{i,j}^{n+1/2}}{\Delta t/2} = D_{xx}^{n+1/2}u_{i,j}^{n+1/2} + D_{yy}^{n+1}u_{i,j}^{n+1} - \frac{1}{2}\left(f_{i.j}^n + f_{i,j}^{n+1}\right) - (C_x)_{i,j}^n - \left(C_y\right)_{i,j}^n.$$

Thus, the actual difference scheme used to get the $u_{i,j}^{n+1}$ becomes

$$\frac{u_{i,j}^{n+1}-u_{i,j}^{n}}{\Delta t}=D_{xx}^{n+1/2}\left(\frac{1}{2}\left(u_{i,j}^{n}+u_{i,j}^{n+1}\right)+\frac{\Delta t}{4}\left(D_{yy}^{n}u_{i,j}^{n}-D_{yy}^{n+1}u_{i,j}^{n+1}\right)\right)+\frac{1}{2}\left(D_{yy}^{n}u_{i,j}^{n}+D_{yy}^{n+1}u_{i,j}^{n+1}\right)$$

$$-\frac{1}{2}\left(f_{i.j}^{n}+f_{i,j}^{n+1}\right)-(C_x)_{i,j}^{n}-\left(C_y\right)_{i,j}^{n}.$$

We use $(C_x)_{i,j}^{n}$ to correct the error in (D)(III)1.

Choose

$$(C_x)_{i,j}^{n}=\frac{1}{2}\frac{1}{h^2}\left(2J_p^{n+1/2}-J_p^{n}-J_p^{n+1}\right),$$

then

$$D_{xx}^{n+1/2}\left(\frac{1}{2}\left(u_{i,j}^{n}+u_{i,j}^{n+1}\right)\right)-(C_x)_{i,j}^{n}=(u_{xx})_{i,j}^{n+1/2}+O(h).$$

Next, we use $\left(C_y\right)_{i,j}^{n}$ to correct the error in (D)(III)2(a).

Since

$$L_x^{(s_x),n+1/2}\left(u_{yyt}\right)_{i,j}^{n+1/2}$$

$$=a_{p,-s_x}\left(u_{yyt}\right)_{i-s_x,j}^{n+1/2}+a_{p,0}\left(u_{yyt}\right)_{i,j}^{n+1/2}+a_{p,s_x}\left(u_{yyt}\right)_{i+s_x,j}^{n+1/2}+a_{p,2s_x}\left(u_{yyt}\right)_{i+2s_x,j}^{n+1/2}$$

$$=\left(\left(\beta_p+\beta_p^2\right)+\alpha_p\left(1+2\beta_p\right)\right)\rho_p\left(\left(u_{yyt}\right)_{i-s_x,j}^{n+1/2}-\left(u_{yyt}\right)_{i,j}^{n+1/2}\right)$$

$$+\left(1+2\beta_p\right)\rho_p\left(\left(u_{yyt}\right)_{i+s_x,j}^{n+1/2}-\left(u_{yyt}\right)_{i,j}^{n+1/2}\right)+\beta_p^2\rho_p\left(\left(u_{yyt}\right)_{i+s_x,j}^{n+1/2}-\left(u_{yyt}\right)_{i+2s_x,j}^{n+1/2}\right)$$

$$=\left(1+2\beta_p\right)\rho_p\left[u_{yyt}\right]_p^{n+1/2}+O(h)$$

$$=\left(1+2\beta_p\right)\rho_p\left(\frac{\left[u_{yy}\right]_p^{n+1}-\left[u_{yy}\right]_p^{n}}{\Delta t}\right)+O(h).$$

So if we choose

$$\left(C_y\right)_{i,j}^{n}=\frac{-1}{4}\rho_p\left(1+2\beta_p\right)\left(\frac{\left[u_{yy}\right]_p^{n+1}-\left[u_{yy}\right]_p^{n}}{\Delta t}\right),$$

then

$$\frac{\Delta t}{4}\frac{1}{h^2}L_x^{(s_x),n+1/2}\left(D_{yy}^{n}u_{i,j}^{n}-D_{yy}^{n+1}u_{i,j}^{n+1}\right)-\left(C_y\right)_{i,j}^{n}=O(h).$$

Finally, in order to compute $\left(C_y\right)_{i,j}^{n}$, we need to know how to compute $\left[u_{yy}\right]_p^t$. Suppose $p=\left(x_p,y_p\right)$. We use the local coordinate transformation at p:

$$\begin{bmatrix}\xi\\ \eta\end{bmatrix}=\begin{bmatrix}n_p^x & n_p^y\\ -n_p^y & n_p^x\end{bmatrix}\begin{bmatrix}x-x_p\\ y-y_p\end{bmatrix}\quad\left(\Leftrightarrow\begin{bmatrix}x\\ y\end{bmatrix}=\begin{bmatrix}x_p\\ y_p\end{bmatrix}+\begin{bmatrix}n_p^x & -n_p^y\\ n_p^y & n_p^x\end{bmatrix}\begin{bmatrix}\xi\\ \eta\end{bmatrix}\right).$$

Define

$$\tilde{u}(\xi,\eta,t)\triangleq u(x(\xi,\eta),y(\xi,\eta),t),$$

then

$$u(x,y,t)=\tilde{u}(\xi(x,y),\eta(x,y),t),$$

and

$$u_y(x,y,t)=\tilde{u}_\xi(\xi,\eta,t)\xi_y+\tilde{u}_\eta(\xi,\eta,t)\eta_y,$$

$$u_{yy}(x,y,t)=\tilde{u}_{\xi\xi}(\xi,\eta,t){\xi_y}^2+2\tilde{u}_{\xi\eta}(\xi,\eta,t)\xi_y\eta_y+\tilde{u}_{\eta\eta}(\xi,\eta,t){\eta_y}^2.$$

Let $(x,y)=p$ we get

$$\left(u_{yy}^{\pm}\right)_p^t=\tilde{u}_{\xi\xi}^{\pm}(0,0,t)\left(n_p^y\right)^2+2\tilde{u}_{\xi\eta}^{\pm}(0,0,t)n_p^x n_p^y+\tilde{u}_{\eta\eta}^{\pm}(0,0,t)\left(n_p^x\right)^2$$

$$\Rightarrow\left[u_{yy}\right]_p^t=\left[\tilde{u}_{\xi\xi}\right]_{(0,0)}^t\left(n_p^y\right)^2+2\left[\tilde{u}_{\xi\eta}\right]_{(0,0)}^t n_p^x n_p^y+\left[\tilde{u}_{\eta\eta}\right]_{(0,0)}^t\left(n_p^x\right)^2.$$

It remains to compute $\left[\tilde{u}_{\xi\xi}\right]_{(0,0)}^t$, , $\left[\tilde{u}_{\xi\eta}\right]_{(0,0)}^t$ and $\left[\tilde{u}_{\eta\eta}\right]_{(0,0)}^t$.

Since in a neighborhood of p, the interface lies in the η direction, we can parameterize the interface locally by $\xi=\chi(\eta)$ with $\chi'(0)=0$ and write the jumps as

$$\tilde{\tau}(\eta,t) \triangleq \tau(x(\chi(\eta),\eta),y(\chi(\eta),\eta),t)$$

$$= [u](x(\chi(\eta),\eta),y(\chi(\eta),\eta),t)$$

$$= [\tilde{u}](\chi(\eta),\eta,t). \tag{95}$$

$$\tilde{\sigma}(\eta,t) \triangleq \sigma(x(\chi(\eta),\eta),y(\chi(\eta),\eta),t)$$

$$= [u_n](x(\chi(\eta),\eta),y(\chi(\eta),\eta),t)$$

$$= [\tilde{u}_n](\chi(\eta),\eta,t). \tag{96}$$

(A)

$$[\tilde{u}_\xi](\xi,\eta,t) = [u_x](x(\xi,\eta),y(\xi,\eta),t)n_p^x + [u_y](x(\xi,\eta),y(\xi,\eta),t)n_p^y.$$

$$\therefore [\tilde{u}_\xi]_{(0,0)}^t = [u_x]_p^t n_p^x + [u_y]_p^t n_p^y$$

$$= [u_n]_p^t. \tag{97}$$

(B)

Define $\tilde{f}(\xi,\eta,t) \triangleq f(x(\xi,\eta),y(\xi,\eta),t)$.

Since

$$\tilde{u}_{\xi\xi}(\xi,\eta,t) = u_{xx}(x,y,t)\left(n_p^x\right)^2 + 2u_{xy}(x,y,t)n_p^x n_p^y + u_{yy}(x,y,t)\left(n_p^y\right)^2,$$

$$\tilde{u}_{\eta\eta}(\xi,\eta,t) = u_{xx}(x,y,t)\left(n_p^y\right)^2 - 2u_{xy}(x,y,t)n_p^x n_p^y + u_{yy}(x,y,t)\left(n_p^x\right)^2,$$

we have

$$\frac{\partial \tilde{u}}{\partial t} - \left(\tilde{u}_{\xi\xi} + \tilde{u}_{\eta\eta}\right) + \tilde{f} = \frac{\partial u}{\partial t} - \left(u_{xx} + u_{yy}\right) + f = 0, \tag{98}$$

1.compute $[\tilde{\mathbf{u}}_{\eta\eta}]_{(\mathbf{0},\mathbf{0})}^{\mathbf{t}}$

differentiating (95) with respect to η we get

$$\tilde{\tau}_\eta(\eta,t) = [\tilde{u}_\xi](\chi(\eta),\eta,t)\,\chi'(\eta) + [\tilde{u}_\eta](\chi(\eta),\eta,t).$$

$$\left(\text{note that if we let } \eta = 0 \text{, we get } [\tilde{u}_\eta]^t_{(0,0)} = \tilde{\tau}_\eta(0,t)\right) \tag{99}$$

Differentiating this with respect to η again we get

$$\tilde{\tau}_{\eta\eta}(\eta,t) = \left(\frac{\partial}{\partial\eta}[\tilde{u}_\xi](\chi(\eta),\eta,t)\right)\chi'(\eta) + [\tilde{u}_\xi](\chi(\eta),\eta,t)\chi''(\eta)$$
$$+[\tilde{u}_{\xi\eta}](\chi(\eta),\eta,t)\chi'(\eta) + [\tilde{u}_{\eta\eta}](\chi(\eta),\eta,t).$$

Let $\eta = 0$ and use (97) and $\chi'(0) = 0$. We get

$$[\tilde{u}_{\eta\eta}]^t_{(0,0)} = -\chi''(0)[u_n]^t_p + \tilde{\tau}_{\eta\eta}(0,t). \tag{100}$$

2.compute $[\tilde{u}_{\xi\xi}]^t_{(0,0)}$

From (98) we have

$$\left[\frac{\partial\tilde{u}}{\partial t}\right](\xi,\eta,t) - \left([\tilde{u}_{\xi\xi}](\xi,\eta,t) + [\tilde{u}_{\eta\eta}](\xi,\eta,t)\right) + [\tilde{f}](\xi,\eta,t) = 0.$$

Combining this with (100), we can get

$$[\tilde{u}_{\xi\xi}]^t_{(0,0)} = -[\tilde{u}_{\eta\eta}]^t_{(0,0)} + \left[\frac{\partial\tilde{u}}{\partial t}\right]^t_{(0,0)} + [\tilde{f}]^t_{(0,0)}$$
$$= \chi''(0)[u_n]^t_p - \tilde{\tau}_{\eta\eta}(0,t) + \left[\frac{\partial u}{\partial t}\right]^t_p + [f]^t_p.$$

3.compute $[\tilde{u}_{\xi\eta}]^t_{(0,0)}$

Since

$$\tilde{u}^{\pm}_n(\chi(\eta),\eta,t) = u^{\pm}_x(x(\chi(\eta),\eta),y(\chi(\eta),\eta),t)n^x_{(x(\chi(\eta),\eta),y(\chi(\eta),\eta))}$$
$$+ u^{\pm}_y(x(\chi(\eta),\eta),y(\chi(\eta),\eta),t)n^y_{(x(\chi(\eta),\eta),y(\chi(\eta),\eta))},$$

and

$$\begin{bmatrix} n^{x}_{(x(\chi(\eta),\eta),y(\chi(\eta),\eta))} \\ n^{y}_{(x(\chi(\eta),\eta),y(\chi(\eta),\eta))} \end{bmatrix} = \begin{bmatrix} n^{x}_{p} & -n^{y}_{p} \\ n^{y}_{p} & n^{x}_{p} \end{bmatrix} \begin{bmatrix} \dfrac{1}{\sqrt{1+(\chi')^2}} \\ \dfrac{-\chi'(\eta)}{\sqrt{1+(\chi')^2}} \end{bmatrix}.$$

We can rewrite $\tilde{u}^{\pm}_{n}(\chi(\eta),\eta,t)$ as

$$\begin{aligned} \tilde{u}^{\pm}_{n}(\chi(\eta),\eta,t) &= u^{\pm}_{x}(x(\chi(\eta),\eta),y(\chi(\eta),\eta),t)\left(\frac{n^{x}_{p}+n^{y}_{p}\chi'(\eta)}{\sqrt{1+(\chi')^2}}\right) \\ &\quad + u^{\pm}_{y}(x(\chi(\eta),\eta),y(\chi(\eta),\eta),t)\left(\frac{n^{y}_{p}-n^{x}_{p}\chi'(\eta)}{\sqrt{1+(\chi')^2}}\right) \\ &= \left(\tilde{u}^{\pm}_{\xi}(\chi(\eta),\eta,t)-\tilde{u}^{\pm}_{\eta}(\chi(\eta),\eta,t)\chi'(\eta)\right)\frac{1}{\sqrt{1+(\chi')^2}}. \end{aligned}$$

Therefore,

$$\begin{aligned} \tilde{\sigma}(\eta,t) &= [\tilde{u}_{n}](\chi(\eta),\eta,t) \ (\text{by}(96)) \\ &= \left([\tilde{u}_{\xi}](\chi(\eta),\eta,t)-[\tilde{u}_{\eta}](\chi(\eta),\eta,t)\chi'(\eta)\right)\frac{1}{\sqrt{1+(\chi')^2}}. \end{aligned}$$

Differentiating this with respect to η we have

$$\begin{aligned} \tilde{\sigma}_{\eta}(\eta,t) &= \left([\tilde{u}_{\xi\xi}](\chi(\eta),\eta,t)\chi'(\eta)+[\tilde{u}_{\xi\eta}](\chi(\eta),\eta,t)-\left(\frac{\partial}{\partial\eta}[\tilde{u}_{\eta}](\chi(\eta),\eta,t)\right)\chi'(\eta)-[\tilde{u}_{\eta}](\chi(\eta),\eta,t)\chi''(\eta)\right)\frac{1}{\sqrt{1+(\chi')^2}} \\ &\quad + \left([\tilde{u}_{\xi}](\chi(\eta),\eta,t)-[\tilde{u}_{\eta}](\chi(\eta),\eta,t)\chi'(\eta)\right)\frac{-\chi'\chi''}{(1+(\chi')^2)^{\frac{3}{2}}}. \end{aligned}$$

Let $\eta = 0$ and $\chi'(0) = 0$. We get

$$\begin{aligned} \tilde{\sigma}_{\eta}(0,t) &= [\tilde{u}_{\xi\eta}]^{t}_{(0,0)} - [\tilde{u}_{\eta}]^{t}_{(0,0)}\chi''(\eta) \\ &= [\tilde{u}_{\xi\eta}]^{t}_{(0,0)} - \tilde{\tau}_{\eta}(0,t)\chi''(\eta) \quad (\text{by } (99)). \end{aligned}$$

Hence

$$[\tilde{u}_{\xi\eta}]^{t}_{(0,0)} = \tilde{\sigma}_{\eta}(0,t) + \tilde{\tau}_{\eta}(0,t)\chi''(\eta).$$

JC 2.

We define $D_{xx}u_{i,j}$ and $D_{yy}u_{i,j}$ to be the approximations of $(u_{xx})_{i,j}$ and $(u_{yy})_{i,j}$ by CIM, respectively. Same as JC 1, assuming $\Delta t = h$, the difference scheme is

$$\frac{u_{i,j}^{n+1}-u_{i,j}^{n}}{\Delta t} = D_{xx}^{n+1/2}\left(\frac{1}{2}\left(u_{i,j}^{n}+u_{i,j}^{n+1}\right)+\frac{\Delta t}{4}\left(D_{yy}^{n}u_{i,j}^{n}-D_{yy}^{n+1}u_{i,j}^{n+1}\right)\right)+\frac{1}{2}\left(D_{yy}^{n}u_{i,j}^{n}+D_{yy}^{n+1}u_{i,j}^{n+1}\right)-\frac{1}{2}\left(f_{i,j}^{n}+f_{i,j}^{n+1}\right).$$

Now, at interface point, following the idea discussed in JC 1, we only need to consider the following term $D_{xx}^{n+1/2}\left(\frac{1}{2}\left(u_{i,j}^{n}+u_{i,j}^{n+1}\right)+\frac{\Delta t}{4}\left(D_{yy}^{n}u_{i,j}^{n}-D_{yy}^{n+1}u_{i,j}^{n+1}\right)\right)$ under the condition $\gamma_{i+1/2,j}+\gamma_{i-1/2,j}=1$.

$$\begin{aligned}
&D_{xx}^{n+1/2}\left(\frac{1}{2}\left(u_{i,j}^{n}+u_{i,j}^{n+1}\right)+\frac{\Delta t}{4}\left(D_{yy}^{n}u_{i,j}^{n}-D_{yy}^{n+1}u_{i,j}^{n+1}\right)\right)\\
&= d_{1,x}\left(\frac{1}{2}\left(u_{i-s_x,j}^{n}+u_{i-s_x,j}^{n+1}\right)+\frac{\Delta t}{4}\left(D_{yy}^{n}u_{i-s_x,j}^{n}-D_{yy}^{n+1}u_{i-s_x,j}^{n+1}\right)\right)\\
&\quad +d_{2,x}\left(\frac{1}{2}\left(u_{i,j}^{n}+u_{i,j}^{n+1}\right)+\frac{\Delta t}{4}\left(D_{yy}^{n}u_{i,j}^{n}-D_{yy}^{n+1}u_{i,j}^{n+1}\right)\right)+d_{3,x}u_{p}^{n+1/2},
\end{aligned}$$

where

$$d_{1,x}=\frac{2}{(x_{i-s_x}-x_i)(x_{i-s_x}-x_p)}\ ,d_{2,x}=\frac{2}{(x_i-x_{i-s_x})(x_i-x_p)}\ ,d_{3,x}=\frac{2}{(x_p-x_{i-s_x})(x_p-x_i)}.$$

We separate it into two parts.

1.

$$\begin{aligned}
&d_{1,x}\frac{1}{2}\left(u_{i-s_x,j}^{n}+u_{i-s_x,j}^{n+1}\right)+d_{2,x}\frac{1}{2}\left(u_{i,j}^{n}+u_{i,j}^{n+1}\right)+d_{3,x}u_{p}^{n+1/2}\\
&=\frac{1}{2}\left(d_{1,x}u_{i-s_x,j}^{n}+d_{2,x}u_{i,j}^{n}+d_{3,x}u\left(x_p,y_j,t^n\right)\right)-\frac{1}{2}d_{3,x}u_p^n\\
&\quad+\frac{1}{2}\left(d_{1,x}u_{i-s_x,j}^{n+1}+d_{2,x}u_{i,j}^{n+1}+d_{3,x}u\left(x_p,y_j,t^{n+1}\right)\right)-\frac{1}{2}d_{3,x}u_p^{n+1}\ \ +d_{3,x}u_p^{n+1/2}\\
&=\frac{1}{2}\left((u_{xx})_{i,j}^{n}+(u_{xx})_{i,j}^{n+1}\right)+O(h)+d_{3,x}\left(u_p^{n+1/2}-\frac{1}{2}\left(u_p^n+u_p^{n+1}\right)\right)\\
&=(u_{xx})_{i,j}^{n+1/2}+O(1).
\end{aligned}$$

So we may choose

$$(C_x)_{i,j}^n = d_{3,x}\left(u_p^{n+1/2} - \frac{1}{2}\left(u_p^n + u_p^{n+1}\right)\right).$$

then

$$d_{1,x}\frac{1}{2}\left(u_{i-s_x,j}^n + u_{i-s_x,j}^{n+1}\right) + d_{2,x}\frac{1}{2}\left(u_{i,j}^n + u_{i,j}^{n+1}\right) + d_{3,x}u_p^{n+1/2} - (C_x)_{i,j}^n$$

$$= (u_{xx})_{i,j}^{n+1/2} + O(h).$$

2.

We only need to consider the case when both $\gamma_{i,j+1/2} + \gamma_{i,j-1/2} \le 1$ and $\gamma_{i\pm1,j+1/2} + \gamma_{i\pm1,j-1/2} \le 1$. From (94), we have

$$D_{yy}^n u_{i,j}^n - D_{yy}^{n+1}u_{i,j}^{n+1} = \left(u_{yy}\right)_{i,j}^n - \left(u_{yy}\right)_{i,j}^{n+1} + O(h^2),$$

so

$$\begin{aligned}
&d_{1,x}\frac{\Delta t}{4}\left(D_{yy}^n u_{i-s_x,j}^n - D_{yy}^{n+1}u_{i-s_x,j}^{n+1}\right) + d_{2,x}\frac{\Delta t}{4}\left(D_{yy}^n u_{i,j}^n - D_{yy}^{n+1}u_{i,j}^{n+1}\right)\\
&= d_{1,x}\frac{\Delta t}{4}\left(\left(u_{yy}\right)_{i-s_x,j}^n - \left(u_{yy}\right)_{i-s_x,j}^{n+1}\right) + d_{2,x}\frac{\Delta t}{4}\left(\left(u_{yy}\right)_{i,j}^n - \left(u_{yy}\right)_{i,j}^{n+1}\right) + O(h)\\
&= \frac{\Delta t}{4}\left(d_{1,x}\left(u_{yy}\right)_{i-s_x,j}^n + d_{2,x}\left(u_{yy}\right)_{i,j}^n + d_{3,x}\left(u_{yy}\right)_p^{-,n}\right) - \frac{\Delta t}{4}d_{3,x}\left(u_{yy}\right)_p^{-,n}\\
&\quad - \frac{\Delta t}{4}\left(d_{1,x}\left(u_{yy}\right)_{i-s_x,j}^{n+1} + d_{2,x}\left(u_{yy}\right)_{i,j}^{n+1} + d_{3,x}\left(u_{yy}\right)_p^{-,n+1}\right) + \frac{\Delta t}{4}d_{3,x}\left(u_{yy}\right)_p^{-,n+1} + O(h)\\
&= \frac{\Delta t}{4}\left(\left(u_{yyxx}\right)_{i,j}^n - \left(u_{yyxx}\right)_{i,j}^{n+1}\right) + \frac{\Delta t}{4}d_{3,x}\left(\left(u_{yy}\right)_p^{-,n+1} - \left(u_{yy}\right)_p^{-,n}\right) + O(h)\\
&= -\frac{\Delta t^2}{4}\left(u_{yyxxt}\right)_{i,j}^{n+1/2} + \frac{\Delta t^2}{4}d_{3,x}\left(\left(u_{yyt}\right)_p^{-,n+1/2}\right) + O(h)\\
&= O(1).
\end{aligned}$$

So we may choose

$$\left(C_y\right)_{i,j}^n = \frac{\Delta t}{4}d_{3,x}\left(\left(u_{yy}\right)_p^{-,n+1} - \left(u_{yy}\right)_p^{-,n}\right),$$

then

$$d_{1,x}\frac{\Delta t}{4}\left(D_{yy}^{n}u_{i-s_x,j}^{n}-D_{yy}^{n+1}u_{i-s_x,j}^{n+1}\right)+d_{2,x}\frac{\Delta t}{4}\left(D_{yy}^{n}u_{i,j}^{n}-D_{yy}^{n+1}u_{i,j}^{n+1}\right)-\left(C_y\right)_{i,j}^{n}=O(h).$$

However, it's impossible to know $\left(u_{yy}\right)_{p}^{(-s_x),n+1}$ in advance, so we can't compute $\left(C_y\right)_{i,j}^{n}$. $\left(\text{Consider}\begin{cases}u_1(x,y,t)=y-x\\u_2(x,y,t)=y^2-x^2\end{cases}\text{, both functions satisfy } u_t=\Delta u \text{ and equal to } 0 \text{ on the line } y=x, \text{ but } (u_1)_{yy}=0 \text{ on } y=x \text{ and } (u_2)_{yy}=2 \text{ on } y=x.\right)$

Numerical examples

JC 1.

Example 1.[12]

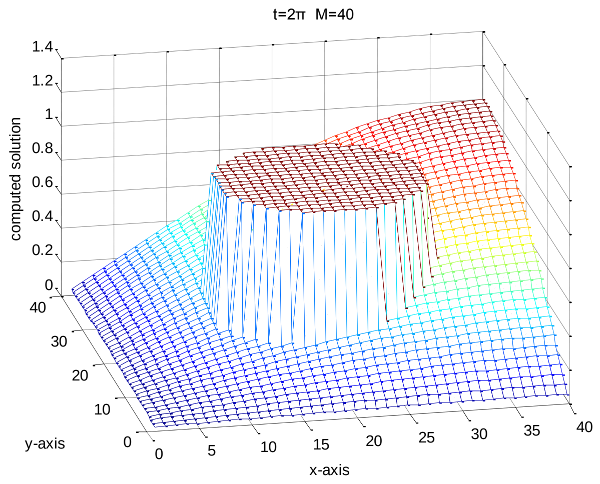

Figure 34. The computed solution $u(x,y,t)$ at $t=2\pi$ and $M=40$ with correction term.

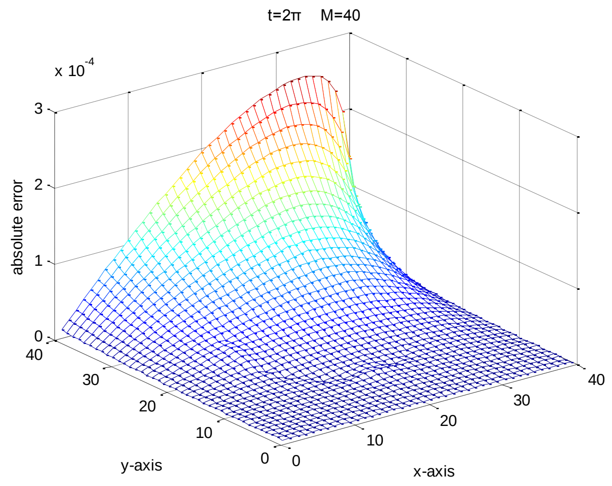

Figure 35. The absolute error at $t=2\pi$ and $M=40$ with correction term.

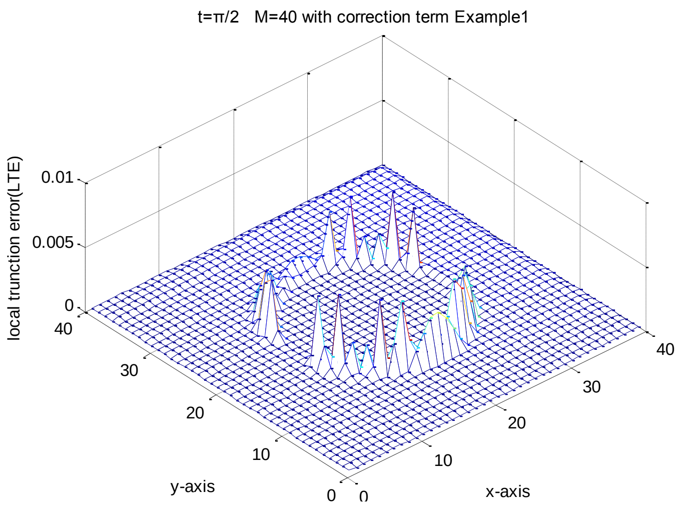

Figure 36. Local truncation error at $t = \pi/2$ and $M = 40$ with correction term.

Table 20. Numerical results for example 1.

	$t=2\pi$		$t=\pi/2$	
M	Max-norm error	order	Max-norm LTE error	order
20	8.698×10^{-4}		1.351×10^{-2}	
40	2.575×10^{-4}	1.756	6.012×10^{-3}	1.168
80	6.845×10^{-5}	1.911	5.269×10^{-3}	0.190
160	1.830×10^{-5}	1.903	2.617×10^{-3}	1.010
320	4.673×10^{-6}	1.969	1.340×10^{-3}	0.966

Example 2.

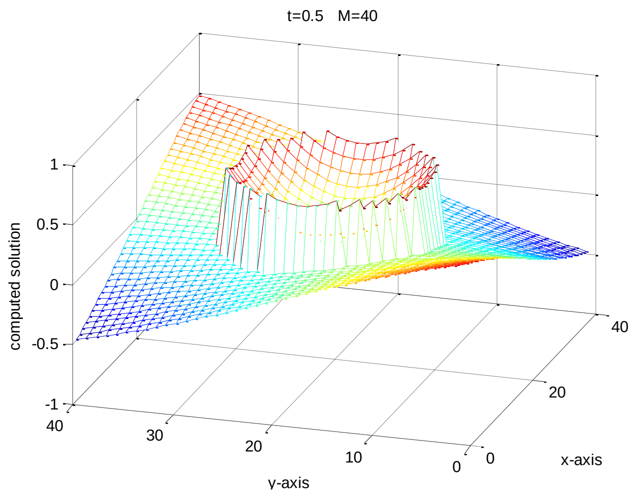

Figure 37. The computed solution $u(x, y, t)$ at $t = 0.5$ and $M = 40$ with correction term.

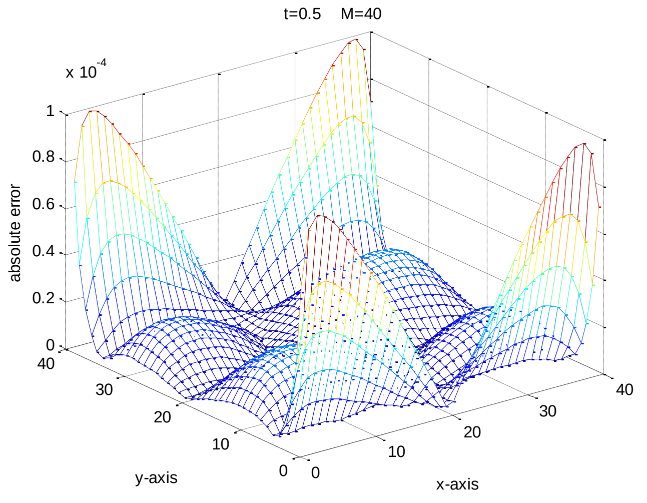

Figure 38. The absolute error at $t = 0.5$ and $M = 40$ with correction term.

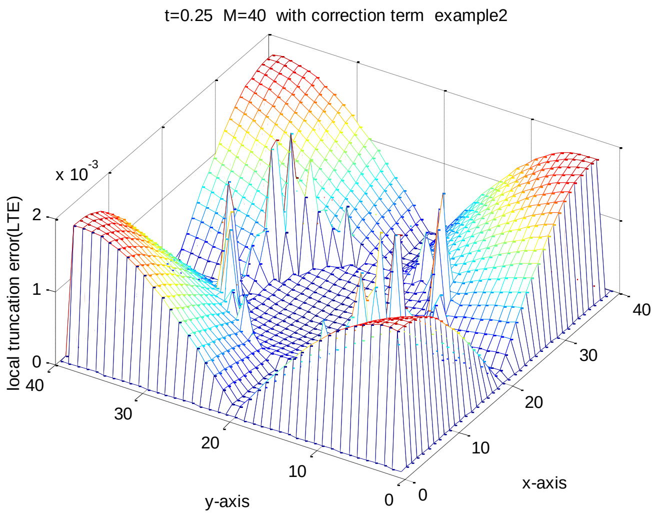

Figure 39. Local truncation error at $t = 0.25$ and $M = 40$ with correction term.

Table 21. Numerical results for example 2.

	t=0.5		t=0.25	
M	Max-norm error	order	Max-norm LTE error	order
20	2.775×10^{-4}		6.603×10^{-3}	
40	9.911×10^{-5}	1.485	1.954×10^{-3}	1.757
80	2.974×10^{-5}	1.737	1.237×10^{-3}	0.660
160	8.280×10^{-6}	1.845	8.064×10^{-4}	0.617
320	2.209×10^{-6}	1.906	3.928×10^{-4}	1.038

As you can see, the local truncation error is much smaller.

JC 2.

Although we don't know how to compute $\left(C_y\right)_{i,j}^{n}$, we still test an example to see what will happen if we know $\left(C_y\right)_{i,j}^{n}$.

Example 2. (Assuming $\left(C_y\right)_{i,j}^{n}$ is known)

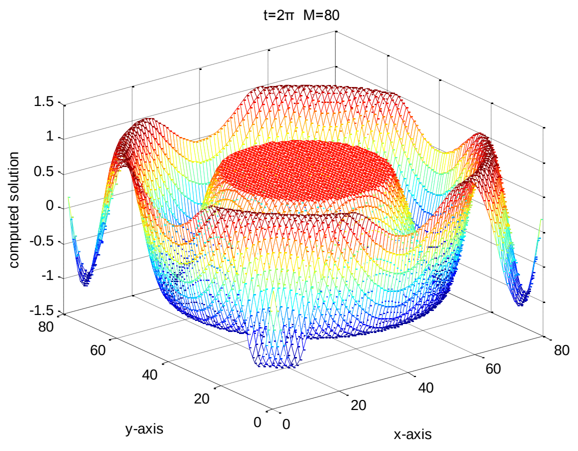

Figure 40. The computed solution $u(x, y, t)$ at $t = 2\pi$ and $M = 80$ with correction term.

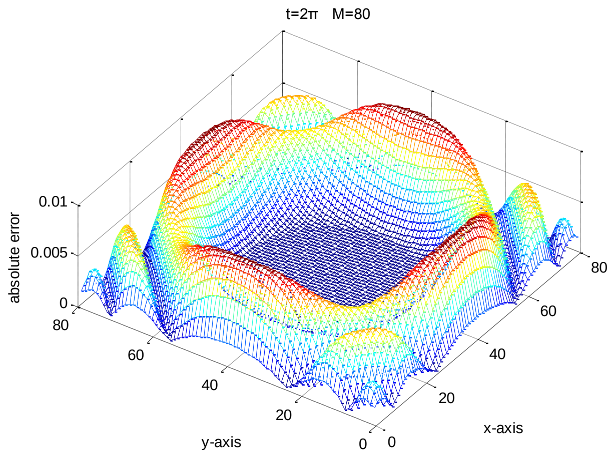

Figure 41. The absolute error at $t = 2\pi$ and $M = 80$ with correction term.

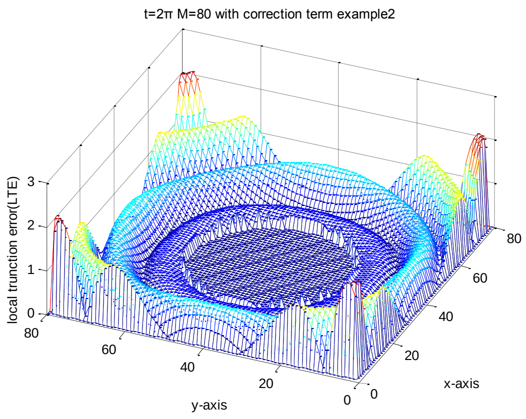

Figure 42. Local truncation error at $t = \pi$ and $M = 80$ with correction term.

Table 22. Numerical results for example 2 .

	t=2π		t=π	
M	Max-norm error	order	Max-norm LTE error	order
20	1.380×10^{-1}		16.529	
40	3.523×10^{-2}	1.970	6.644	1.315
80	9.058×10^{-3}	1.960	2.119	1.649
160	2.293×10^{-3}	1.982	0.538	1.978
320	5.766×10^{-4}	1.992	0.184	1.548

In summary, we apply the CIM to the two dimensional diffusion equations with fixed interface using Crank-Nicholson scheme and confirm its second-order accuracy. We also combine the CIM with ADI method and add some correction terms to make sure it is second-order as well.

Chapter 5
Application to Two Dimensional Melting Problems

In this chapter we apply the coupling interface method to the two dimensional melting problems. Let $[a, b] \times [a, b]$ be our domain Ω of consideration. Denote the phase transition interface in Ω by Γ. By conservation of energy, we have

$$\frac{\partial T}{\partial t} = \frac{\kappa}{C_p \rho} \Delta T = D \Delta T \qquad (x, y) \in \Omega \setminus \Gamma, \tag{101}$$

where C_p is the specific heat at constant pressure, ρ is the density of the material, κ is the thermal conductivity and $D \triangleq \frac{\kappa}{C_p \rho}$ the diffusion coefficient. We denote the region, temperature, thermal conductivity, specific heat with constant pressure, density and diffusion coefficient of water (ice) by Ω_l, T_l, κ_l, C_p^l, ρ_l and D_l (replace l by s), respectively. See figure 43 for the relative position of water and ice. The rate of net heat deposited at the interface is

$$H = -[C_p \kappa T_n] \triangleq -\left(C_p^l \kappa_l \frac{\partial T_l}{\partial \vec{n}} - C_p^s \kappa_s \frac{\partial T_s}{\partial \vec{n}}\right).$$

Here the jump is taken from water to ice and $\vec{n}$ is the outward normal direction of ice. This net heat will be the latent heat to either melt or freeze (depending on the sigh of H) the material at the interface. The energy balance at the ice-water interface can be written as $\rho_s L \upsilon_n = H$, (102)

where L is the latent heat and υ_n is the normal velocity of the moving front. Finally, the temperature at the interface is a constant (melting temperature)

$$T = T_M \ \text{ on } \Gamma, \tag{103}$$

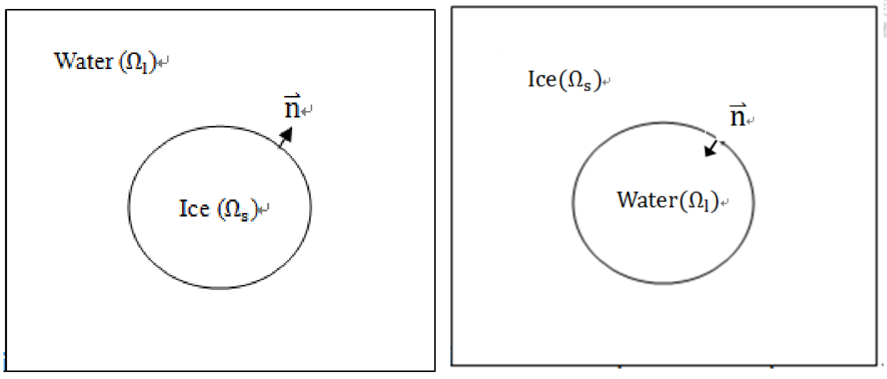

Figure 43. The relative position of water and ice.

Numerical method

We partition $[a, b]$ into $M + 1$ subintervals evenly like we did in previous chapters. As usual, we denote the mesh size and the temporal step size in time by h and Δt, respectively. The Crank-Nicholson scheme for (101) is

$$\frac{T_{i,j}^{n+1} - T_{i,j}^{n}}{\Delta t} = \frac{1}{2} D_{i,j} \left((T_{xx})_{i,j}^{n} + \left(T_{yy}\right)_{i,j}^{n} \right) + \frac{1}{2} D_{i,j} \left((T_{xx})_{i,j}^{n+1} + \left(T_{yy}\right)_{i,j}^{n+1} \right). \tag{104}$$

Spatial discretization

Since the discussion here doesn't concern with time, we will drop t or the superscript n for simplicity. At an interior point $\left(x_i, y_j\right)$, a standard central finite difference scheme is adopted. Namely,

$$(T_{xx})_{i,j} = \frac{T_{i-1,j} - 2T_{i,j} + T_{i+1,j}}{h^2} + O(h^2),$$

$$\left(T_{yy}\right)_{i,j} = \frac{T_{i,j-1} - 2T_{i,j} + T_{i,j+1}}{h^2} + O(h^2).$$

At an on-front grid point (x_i, y_j), as in JC 2 in chapter 4, we consider the x-direction first. We separate it into three parts.

(A) $\gamma_{i+1/2,j} + \gamma_{i-1/2,j} = 2$

Denote the interface point in $[x_{i-1}, x_i) \times \{y_j\}$ and $[x_i, x_{i+1}) \times \{y_j\}$ by $p_- = (x_{p_-}, y_{p_-})$ and $p_+ = (x_{p_+}, y_{p_+})$, respectively. Then

$$(T_x)_{i\pm1/2,j} = \frac{T_{p_\pm} - T_{i,j}}{x_{p_\pm} - x_i} + O(h),$$

and we approximate $(T_{xx})_{i,j}$ by

$$(T_{xx})_{i,j} = \frac{1}{h}\left((T_x)_{i+1/2,j} - (T_x)_{i-1/2,j}\right) + O(1).$$

(B) $\gamma_{i+1/2,j} + \gamma_{i-1/2,j} = 1$

Given (x_1, y), (x_2, y), (x_3, y) and (x, y) four points. Define

$$D_x^2 T\left(x_1, x_2, x_3, (x, y)\right) \triangleq d_{1,x} T(x_1, y) + d_{2,x} T(x_2, y) + d_{3,x} T(x_3, y),$$

where $\left(d_{1,x}, d_{2,x}, d_{3,x}\right)$ is defined in (90).

Then

$$(T_{xx})_{i,j} = D_x^2 T\left(x_{i-s_x}, x_i, x_p, (x_i, y_j)\right) + O(h),$$

where x_p is the x-component of the interface point p

(C) $\gamma_{i+1/2,j} + \gamma_{i-1/2,j} = 0$

We approximate $(T_{xx})_{i,j}$ by

$$(T_{xx})_{i,j} = \frac{T_{i-1,j} - 2T_{i,j} + T_{i+1,j}}{h^2} + O(h^2).$$

A parallel argument applies to y-direction.

Next we discuss how to compute T_x and T_y at interface points. We need to compute these quantities in order to approximate υ_n and $\left[\frac{\partial T}{\partial \vec{n}}\right]$. We give a first order and second order method to compute these quantities.

Method 1 (first order method)

Given an interface point p=(x_p, y_p), Suppose we want to compute $\frac{\partial T}{\partial \vec{n}}$ at p in liquid state (the solid state is the same), consider the following point :

$p_l = p + h\vec{n}$.

Choose the grid points in liquid state which is the closest to p_l. We call this point (x_i, y_j). By Taylor expansion we have

$$T_{p,l} = T_{i,j} + (T_x)_{i,j}(x_{p_l} - x_i) + (T_y)_{i,j}(y_{p_l} - y_j) + O(h^2).$$

We can easily get a first order approximation of $(T_x)_{i,j}$ and $(T_y)_{i,j}$. Finally, we approximate $\frac{\partial T_l}{\partial \vec{n}}$ by

$$\frac{\partial T_l}{\partial \vec{n}} = \frac{T_{p_l} - T_M}{h} + O(h).$$

Method 2 (second order method)

Given an interface point p=(x_p, y_p), Suppose we want to compute T_x, T_y at p in liquid state, choose the grid points in liquid state which is the closest to p (the solid state is the same). We call this point (x_i, y_j). By Taylor expansion we have

$$T_x = (T_x)_{i,j} + (T_{xx})_{i,j}(x_p - x_i) + (T_{xy})_{i,j}(y_p - y_j) + O(h^2),$$

$$T_y = (T_y)_{i,j} + (T_{yx})_{i,j}(x_p - x_i) + (T_{yy})_{i,j}(y_p - y_j) + O(h^2).$$

So it suffices to find a second order approximation of $(T_x)_{i,j}$ and $\left(T_y\right)_{i,j}$, and a first order approximation of $\left(T_{xy}\right)_{i,j}$ and $\left(T_{yx}\right)_{i,j}$. We follow the idea in the CIM.

(I) approximation of $(T_x)_{i,j}$

(A) $\gamma_{i+1/2,j} + \gamma_{i-1/2,j} = 2$

1. $x_p > x_i$

We approximate $(T_x)_{i,j}$ by

$$(T_x)_{i,j} = \frac{T\left(\hat{x}^n, y_j, t^n\right) - T_{i,j}^n}{\hat{x}^n - x_i} + O(h),$$

where $\left(\hat{x}^n, y_j\right)$ is an interface point in $[x_i, x_{i+1}) \times \{y_j\}$

2. $x_p < x_i$

We approximate $(T_x)_{i,j}$ by

$$(T_x)_{i,j} = \frac{T_{i,j}^n - T\left(\hat{x}^n, y_j, t^n\right)}{x_i - \hat{x}^n} + O(h),$$

where $\left(\hat{x}^n, y_j\right)$ is an interface point in $[x_{i-1}, x_i) \times \{y_j\}$

(B) $\gamma_{i+1/2,j} + \gamma_{i-1/2,j} \leq 1$

We approximate $(T_x)_{i,j}$ by

$$(T_x)_{i,j} = \frac{1}{h} D_{x,(s_x)} T(x,y) + s_x \frac{1}{2} (T_{xx})_{i,j} h + O(h^2).$$

(II) approximation of $\left(T_{xy}\right)_{i,j}$

(A) $\gamma_{i,j+1/2} + \gamma_{i,j-1/2} = 2$

$$\left(T_{xy}\right)_{i,j} = 0 + O(1).$$

(B) $\gamma_{i,j+1/2} + \gamma_{i,j-1/2} = 1$

If $\gamma_{i-1/2,j} + \gamma_{i+1/2,j} + \gamma_{i-1/2,j-s_y} + \gamma_{i+1/2,j-s_y} = 0$, then

$$(T_{xy})_{i,j} = s_y\left(\left(\frac{T_{i+1,j} - T_{i-1,j}}{2h^2}\right) - \left(\frac{T_{i+1,j-s_y} - T_{i-1,j-s_y}}{2h^2}\right)\right) + O(h).$$

Else if $\gamma_{i-1/2,j} + \gamma_{i-1/2,j-s_y} = 0$, then

$$(T_{xy})_{i,j} = s_y\left(\left(\frac{T_{i,j} - T_{i-1,j}}{h^2}\right) - \left(\frac{T_{i,j-s_y} - T_{i-1,j-s_y}}{h^2}\right)\right) + O(h).$$

Else if $\gamma_{i+1/2,j} + \gamma_{i+1/2,j-s_y} = 0$, then

$$(T_{xy})_{i,j} = s_y\left(\left(\frac{T_{i+1,j} - T_{i,j}}{h^2}\right) - \left(\frac{T_{i+1,j-s_y} - T_{i,j-s_y}}{h^2}\right)\right) + O(h).$$

Else

$$(T_{xy})_{i,j} = 0 + O(1).$$

(C) $\gamma_{i,j+1/2} + \gamma_{i,j-1/2} = 0$

If $y_p < y_j$, follow (B) and set $s_y = 1$. If $y_p > y_j$, follow (B) and set $s_y = -1$.

A parallel argument applies to y-direction.

Representation of the interface location

We use the front tracking method (FronTier library[6]) to simulate the melting problem. Front tracking is a numerical method in which the interface is explicitly represented in the discrete form of curves in two dimension. The discrete solution is based on a composite grid that consists of a spatial grid, together with a co-dimension one grid that represents the tracked fronts. In two space dimensions the front consists of curves with piecewise linear segments called bonds.

When using front tracking method, a robust and stable calculation of curvature on a discretized interface mesh is required to give an accurate growth rate of the ice-water interface. Several methods have been proposed to estimate the normal vector at each vertex of a discrete mesh. FronTier library use the method of local least square fitting: first they construct a local coordinate system by defining a height function and then perform polynomial fitting to obtain derivatives, and finally convert the derivatives to normal or curvature. For the details, refer to [9]. In 2D, the curve is defined by a height function $f: \mathbb{R} \longrightarrow \mathbb{R}$. Given a height function f, let f' denote its first derivative. The function f defines a curve composed of points $(x, f(x)) \in \mathbb{R}^2$ and the unit normal vector to the curve is

$$\vec{n} = \frac{(-f'(x), 1)}{\sqrt{f'(x)^2 + 1}}.$$

Although the CIM is a second order method, the FronTier library only provide explicit Euler method to compute the interface in melting problem, so the overall method is only first-order.

Numerical examples

Example 1.

$$\frac{\partial T}{\partial t} = \Delta T - g(x,y,t) \qquad (x,y) \in \Omega \setminus \Gamma,\ \ \Omega = [-3,3] \times [-3,3],\ \ t > 0\,,$$

$$\upsilon_n = -[T_n],$$

$$T = 0 \text{ on } \Gamma,$$

with

$$g(x,y,t) = \begin{cases} 0 & \text{if } \sqrt{x^2+y^2} \le -t+2, \\ -\exp\left(t+\sqrt{x^2+y^2}-2\right)\dfrac{1}{\sqrt{x^2+y^2}} & \text{if } \sqrt{x^2+y^2} > -t+2. \end{cases}$$

The exact solution is

$$T(x,y,t) = \begin{cases} 0 & \text{if } \sqrt{x^2+y^2} \le -t+2, \\ \exp\left(t+\sqrt{x^2+y^2}-2\right)-1 & \text{if } \sqrt{x^2+y^2} > -t+2. \end{cases}$$

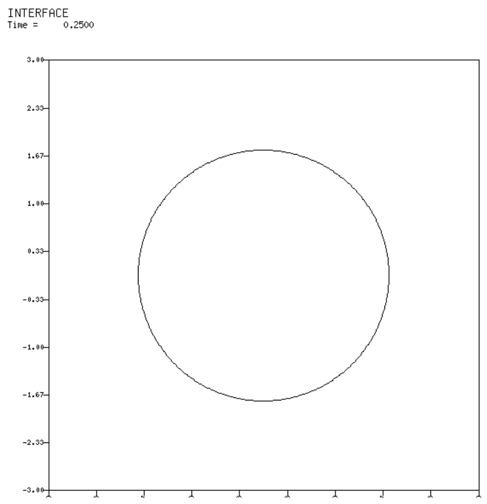

Figure 44. The computed interface at $t = 0.25$ with $M = 128$.

Table 23. Numerical results for example 1.

	$t = 0.25$			
M	max-norm error	order	error in interface location	order
32	8.089×10^{-2}		3.008×10^{-3}	
64	2.231×10^{-2}	1.858	8.000×10^{-4}	1.911
128	5.860×10^{-3}	1.929	2.820×10^{-4}	1.504
256	1.502×10^{-3}	1.964	1.540×10^{-4}	0.873

Example 2.

$$\frac{\partial \mathrm{T}}{\partial \mathrm{t}} = \Delta \mathrm{T} - \mathrm{g}(\mathrm{x},\mathrm{y},\mathrm{t}) \qquad (\mathrm{x},\mathrm{y}) \in \Omega \setminus \Gamma,\ \ \Omega = [-2,2] \times [-2,2],\ \ \mathrm{t} > 0,$$

$$\upsilon_{\mathrm{n}} = -[\mathrm{T}_{\mathrm{n}}],$$

$$\mathrm{T} = 0 \text{ on } \Gamma,$$

with

$$\mathrm{g}(\mathrm{x},\mathrm{y},\mathrm{t}) = \begin{cases} \frac{1}{3}\exp\left(\frac{1}{9}\mathrm{t} - \frac{1}{3}\sqrt{\mathrm{x}^2+\mathrm{y}^2} + \frac{1}{3}\right)\frac{1}{\sqrt{\mathrm{x}^2+\mathrm{y}^2}} & \text{if } \sqrt{\mathrm{x}^2+\mathrm{y}^2} \le \frac{1}{3}\mathrm{t}+1, \\ 0 & \text{if } \sqrt{\mathrm{x}^2+\mathrm{y}^2} > \frac{1}{3}\mathrm{t}+1. \end{cases}$$

The exact solution is

$$\mathrm{T}(\mathrm{x},\mathrm{y},\mathrm{t}) = \begin{cases} \exp\left(\frac{1}{9}\mathrm{t} - \frac{1}{3}\sqrt{\mathrm{x}^2+\mathrm{y}^2} + \frac{1}{3}\right)\frac{1}{\sqrt{\mathrm{x}^2+\mathrm{y}^2}} - 1 & \text{if } \sqrt{\mathrm{x}^2+\mathrm{y}^2} \le \frac{1}{3}\mathrm{t}+1, \\ 0 & \text{if } \sqrt{\mathrm{x}^2+\mathrm{y}^2} > \frac{1}{3}\mathrm{t}+1. \end{cases}$$

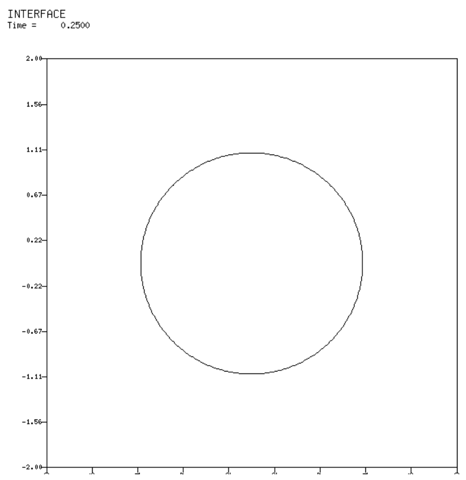

Figure 45. The computed interface at $\mathrm{t} = 0.25$ with $\mathrm{M} = 128$.

Table 24. Numerical results for example 2.

	$\mathrm{t} = 0.25$			
M	max-norm error	order	error in interface location	order
32	2.890×10^{-2}		3.398×10^{-3}	
64	1.946×10^{-2}	0.571	8.270×10^{-4}	2.039
128	1.232×10^{-2}	0.660	3.370×10^{-4}	1.295
256	7.474×10^{-3}	0.721	9.300×10^{-5}	1.857

Remark :

1. If we can get a second order approximation of the interface location, then in order to get an overall second order accuracy we need to consider the grid crossing and the time correction term (like in the 1-D case):

$$\frac{T_{i,j}^{n+1} - T_{i,j}^{n}}{\Delta t} - C_{i,j}^{n} = \frac{1}{2} D_{i,j}\left((T_{xx})_{i,j}^{n} + (T_{yy})_{i,j}^{n}\right) + \frac{1}{2} D_{i,j}\left((T_{xx})_{i,j}^{n+1} + (T_{yy})_{i,j}^{n+1}\right),$$

where $C_{i,j}^{n}$ is a correction term for time. Suppose (x_i, y_j, t^n) and (x_i, y_j, t^{n+1}) are on the same region of the interface, then there is no need for any correction and we set $C_{i,j}^{n} = 0$. However, if there is a grid crossing at (x_i, y_j) from time t^n to time t^{n+1}, say, at time $\tilde{t}$, $t^n < \tilde{t} < t^{n+1}$ (see figure 46), then like what we did in chapter 1 we need to add a correction term $C_{i,j}^{n}$ to make sure the time derivative of T has at least first-order accuracy. The following theorem tells us how to choose $C_{i,j}^{n}$.

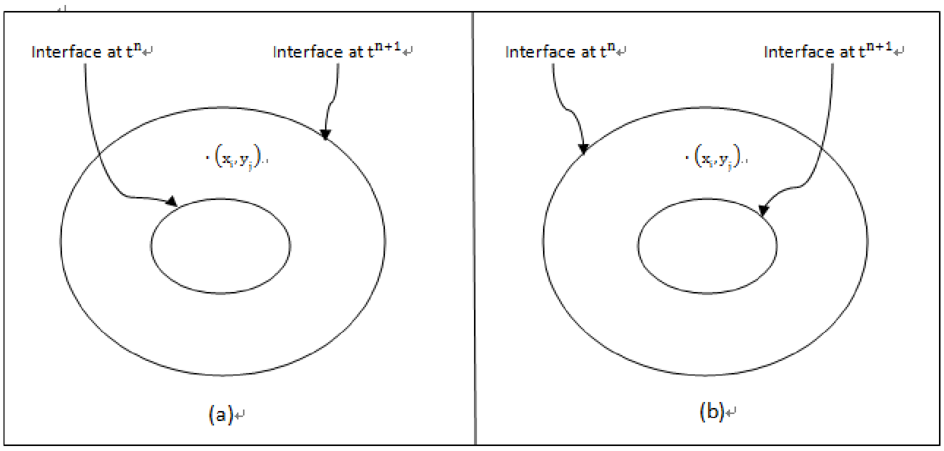

Figure 46.

Theorem If there is a grid crossing at (x_i, y_j) from time t^n to time t^{n+1} at time $\tilde{t} \in (t^n, t^{n+1})$, then if we choose

$C_{i,j}^{n} = \frac{1}{\Delta t}\left(t^{n} + \frac{1}{2}\Delta t - \tilde{t}\right)[T_t]_{;\tilde{t}}$,

then we can get a first order approximation of the time derivative of T

$$\frac{T_{i,j}^{n+1} - T_{i,j}^{n}}{\Delta t} - C_{i,j}^{n} = \frac{1}{2}\left((T_t)_{i,j}^{n+1} + (T_t)_{i,j}^{n}\right) + O(\Delta t),$$

where $[T]_{;\tilde{t}} \triangleq T\left(x_i, y_j, \tilde{t}^{+}\right) - T\left(x_i, y_j, \tilde{t}^{-}\right)$.

Note that $[T]_{;\tilde{t}} = -[T]\left(x_i, y_j, \tilde{t}\right)$ if $\left(x_i, y_j\right)$ is in the water region at time t^{n} and in the ice region at time t^{n+1} (see figure 46(a)), and $[T]_{;\tilde{t}} = [T]\left(x_i, y_j, \tilde{t}\right)$ if $\left(x_i, y_j\right)$ is in the ice region at time t^{n} and in the water region at time t^{n+1}(see figure 46(b))

The proof is exactly the same as in one-dimensional case, so we omit it. In order to find $C_{i,j}^{n}$, we need to compute $\tilde{t}$ and $[T_t]_{;\tilde{t}}$. First we discuss how to find $\tilde{t}$. If there is a grid crossing at (x_i, y_j) from time t^{n} to time t^{n+1} at time $\tilde{t}$, then at time t^{n} by front tracking we may find two under-tracking interface points p_1^{n} and p_2^{n} with corresponding normal vectors $\vec{n}_1, \vec{n}_2$ and normal velocities $\upsilon_{n,1}$, $\upsilon_{n,2}$,respectively such that $\left(x_i, y_j\right)$ lies in the region enclosed by four straight lines connecting p_1^{n}, p_1^{n+1}, p_2^{n+1} and p_2^{n} (see figure 47), where

$p_1^{n+1} = p_1^{n} + \Delta t \bar{\upsilon}_{n,1}\vec{n}_1$,

$p_2^{n+1} = p_2^{n} + \Delta t \bar{\upsilon}_{n,2}\vec{n}_2$,

are the estimated positions of the interface at time t^{n+1}. Here $\bar{\upsilon}_{n,1}$ and $\bar{\upsilon}_{n,2}$ depend on how we approximate the interface location (ex. $\bar{\upsilon}_{n,1} = \upsilon_{n,1}$ and $\bar{\upsilon}_{n,2} = \upsilon_{n,2}$ if we use the forward Euler method). Then we may find $\Delta\tilde{t} \in (0, \Delta t)$ and $s \in (0,1)$ such that

$$(1-s)\left(p_1^n + \Delta\tilde{t}\,\bar{\upsilon}_{n,1}\vec{n}_1\right) + s\left(p_2^n + \Delta\tilde{t}\,\bar{\upsilon}_{n,2}\vec{n}_2\right) = \left(x_i, y_j\right),$$

and we estimate $\tilde{t}$ by

$$\tilde{t} \approx t^n + \Delta\tilde{t}.$$

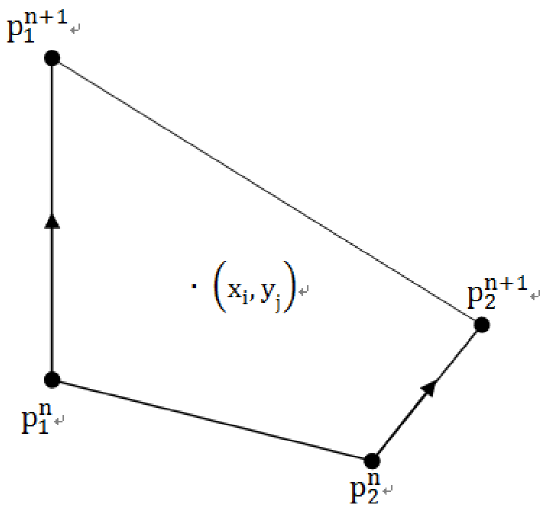

Figure 47.

Next, we discuss how to find $[T_t]_{;\tilde{t}}$. Like in the 1D case, we have the following formula:

$$[T_t]\left(x_i, y_j, \tilde{t}\right) = -v_n\left(x_i, y_j, \tilde{t}\right)\left[\frac{\partial T}{\partial \vec{n}}\right]\left(x_i, y_j, \tilde{t}\right).$$

So we may calculate $[T_t]_{;\tilde{t}}$ as

$$[T_t]_{;\tilde{t}} = -[T_t]\left(x_i, y_j, \tilde{t}\right) = v_n\left(x_i, y_j, \tilde{t}\right)\left[\frac{\partial T}{\partial \vec{n}}\right]\left(x_i, y_j, \tilde{t}\right)$$

$$\approx \begin{cases} \upsilon_{n,2}\left[\frac{\partial T}{\partial \vec{n}}\right](p_2^n, t^n) & \text{if } s > 0.5 \\ \upsilon_{n,1}\left[\frac{\partial T}{\partial \vec{n}}\right](p_1^n, t^n) & \text{if } s < 0.5 \end{cases} \quad \text{in figure 46(a),}$$

and

$$[T_t]_{;\tilde{t}} = [T_t]\left(x_i, y_j, \tilde{t}\right) = -v_n\left(x_i, y_j, \tilde{t}\right)\left[\frac{\partial T}{\partial \vec{n}}\right]\left(x_i, y_j, \tilde{t}\right)$$

$$\approx \begin{cases} -\upsilon_{n,2}\left[\frac{\partial T}{\partial \vec{n}}\right](p_2^n, t^n) & \text{if } s > 0.5 \\ -\upsilon_{n,1}\left[\frac{\partial T}{\partial \vec{n}}\right](p_1^n, t^n) & \text{if } s < 0.5 \end{cases} \quad \text{in figure 46(b),}$$

2. When we use the second order method to compute T_x and T_y at the interface points,

if the grid point (x_i, y_j) we choose lies in the following region (see figure 48), then the state we determine at (x_i, y_j) will be wrong. Consequently the error in computing T_x and T_y will become larger.

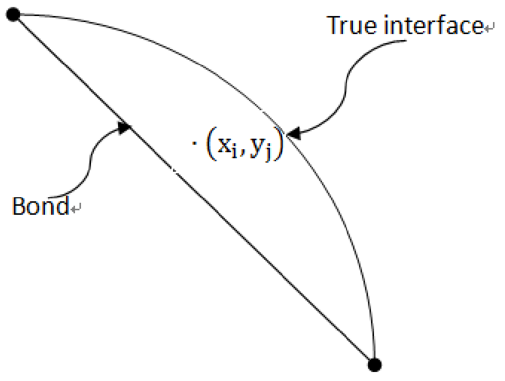

Figure 48.

In summary, we apply the CIM to the two dimensional melting problems using Crank-Nicholson scheme and front tracking method (explicit Euler method) to move the interface. The overall method is first-order. We also provide a second-order method to compute normal derivatives at the interface and a method to compute time correction term.

Chapter 6
Conclusions

In this thesis, we apply the coupling interface method (CIM) to various interface problems. In one dimensional moving interface problems, combining the Crank-Nicholson scheme and prediction correction approach (latter on linear multistep method), we verify the method is second-order accurate. We also apply the CIM to the two-dimensional diffusion problems with fixed interface combining Crank-Nicholson scheme and ADI method. The result is second-order also. Finally, we apply the CIM to two dimensional melting problems. Although the overall method is first-order, we provide a second-order method to compute normal derivatives at the interface and a method to compute time correction term for future reference.

REFERECE

[1] R. Almgren, Variational algorithms and pattern formation in dendritic solidification, Journal of Computational Physics 106 (1993) 337–354.

[2] N. Al-Rawahi, G. Tryggvason, Numerical simulation of dendritic solidification with convection: two-dimensional geometry, Journal of Computational Physics 180 (2002) 471–496.

[3] R.P. Beyer, R.J. LeVeque, Analysis of a one-dimensional model for the immersed boundary method, SIAM Journal in Numerical Analysis Vol. 29, No. 2 (Apr., 1992) 332-364.

[4] S. Chen, B. Merriman, S. Osher, P. Smereka, A simple level set method for solving Stefan problems, Journal of Computational Physics 135 (1997) 8–29.

[5] I.L. Chern, Y.C. Shu, A coupling interface method for elliptic interface problems, Journal of Computational Physics 225 (2007) 2138–2174.

[6] J. Glimm, M. J. Graham, J. Grove, X. L. Li, T.M. Smith, D. Tan, F. Tangerman, Q. Zhang, J, front tracking in two and three dimensions, Comp. Math. 7 (1998) 1-12.

[7] F. Gibou, R.P. Fedkiw, L.T. Cheng, M. Kang, A second-order- accurate symmetric discretization of the poisson equation on irregular domains, Journal of Computational Physics 176 (2002) 205–227.

[8] D. Juric, G. Tryggvason, A front-tracking method for dendritic solidification, Journal of Computational Physics 123 (1996) 127–148.

[9] X.M. Jiao, H.Y. Zha, Consistent computation of first- and second-order differential quantities for surface meshes, ACM Solid and Physical Modeling Symposium (2008) 159-170.

[10] X.L. Li, J. Glimm, X.M. Jiao, C. Peyser, Y.H. Zhao, Study of crystal growth and solute precipitation through front tracking method, Acta Mathematica Scientia 30B(2) (2010) 377–390.

[11] Z.L. Li, Immersed interface methods for moving interface Problems, Numerical Algorithms 14 (1997) 269–293.

[12] Z.L. Li, The immersed interface method – a numerical approach for partial differential equations with interfaces, Ph. D. Thesis, University of Washington (1994).

[13] Z.L. Li, The immersed interface method: numerical solutions of pdes involving interfaces and irregular domains, SIAM Frontiers in Applied mathematics 33 (2006).

[14] Z.L. Li, Fast and accurate numerical approaches for Stefan problems and crystal growth, Numerical Heat Transfer Part B (1999), in press.

[15] R.J. LeVeque, Finite difference methods for ordinary and partial differential equations: steady-state and time-dependent problems, SIAM (2007).