The Immersed Interface Method – A Numerical Approach for Partial Differential Equations with Interfaces

by

Zhilin Li

A dissertation submitted in partial fulfillment of
the requirements for the degree of

Doctor of Philosophy

University of Washington

1994

Approved by _____
(Chairperson of Supervisory Committee)

_____

_____

Program Authorized
to Offer Degree _____

Date _____

Signature_____

Date_____

University of Washington

Abstract

The Immersed Interface Method – A Numerical Approach for Partial Differential Equations with Interfaces

by Zhilin Li

Chairperson of Supervisory Committee: *Professor Randall J. LeVeque*
Department of Applied Mathematics

This thesis describes the Immersed Interface Method (IIM) for interface problems, in which the partial differential equations have discontinuities and singularities in the coefficients and the solutions. A typical example of such problems is heat conduction in different materials (discontinuous heat conductivity), or fluid interface problems where the surface tension gives a singular force that is supported only on the interface. The complexity of the interfaces makes it more difficult to develop efficient numerical methods.

Our immersed interface method is motivated by and related to Peskin's immersed boundary method (IBM) for solving incompressible Navier-Stokes equations with complicated boundaries. Our method, however, can apply to more general interface problems and usually attains second order accuracy.

We use uniform Cartesian grids so that we can take advantages of many conventional difference schemes for the regular grid points which are away from the interface. Hence attention is focused on developing difference schemes for the irregular grid points near the interface, which can cut through the grid in an arbitrary manner. Assuming a knowledge of jump conditions on the solution across the interface, which usually can be obtained either from the differential equation itself or by physical reasoning, we carefully choose the stencil and the coefficients of the difference scheme. By using local coordinate transformations and a modified undetermined coefficients method, we force the local truncation error to be $O(h^2)$ at regular grid points and $O(h)$ at irregular ones. For many interface problems, this leads to a second order accurate solution globally even if the solution is discontinuous. Cubic splines are used to represent and update the complicated interfaces.

We have implemented the immersed interface method for a number of interface problems including: general elliptic equations in one, two and three dimensions, alternating direction implicit methods for heat equations with singular sources or dipoles, the Stokes equations with a moving interface, and Stefan-like 1D moving interface problems in which the interface is determined by a nonlinear differential equation. Theoretical analysis and numerical examples are presented to show the efficiency of the immersed interface method. We believe that this methodology can be successfully applied to many other interface problems as well.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I am very grateful for my thesis advisor, Professor Randall LeVeque for his invaluable assistance, guidance, encouragement and friendship throughout my Ph.D study and thesis project. It is an enjoyable experience working with him. He not only brought me to the frontier of numerical analysis in the research but also taught me a great deal about applied mathematics and other experiences in general. Our innumerable times discussing and questioning the problems, and cheering any achievements together will leave a lasting impression. I look forward to collaborating with him on future research and expect constant support from him.

I would also like to thank the other members of my supervisory committee: Professor Loyce Adams, Professor Kenneth Bube, and Professor Ka-Kit Tung. In particular Professor Loyce Adams and Professor Kenneth Bube have painstakingly read through my thesis. I really appreciate their time, comments and suggestions, and other support and encouragement for my success.

I would also like to thank the faculty, students and staff at the Department of Applied Mathematics, University of Washington for providing a stimulating environments. Finally I would also like to thank my family for their constant support. This research was supported in part by NFS Grants DMS-8657319 and DMS-9204329, and DOE Grant DE-FG06-93ER25181.

Chapter 1

INTRODUCTION.

Interface problems have attracted a lot of attention from numerical analysts over the years. Mathematically, interface problems usually lead to differential equations whose input data and solutions have discontinuities or non-smoothness across some interface. Many numerical methods designed for smooth solutions do not work efficiently for interface problems. This thesis is concerned with developing, improving and analyzing numerical methods for various interface problems using *uniform Cartesian grids*. We will take advantage of the classic central difference scheme at regular grid points which are away from the interface and derive modified difference schemes at irregular grid points. These are much fewer in number than the regular ones so this will not increase the computational cost too much. Generally we get second order accurate solutions globally in the infinity norm for most of the interface problems discussed in this thesis. Combining these methods with the spline interpolation techniques, we are also able to deal with complicated geometries and moving interface problems.

Interface problems occur in many physical applications. We present a model problem below to show the importance and characteristics of interface problems.

1.1 A model problem.

The heat equation

$$u_t = \nabla \cdot (\beta \nabla u) + f \tag{1.1}$$

describes many physical phenomena. For instance, u may represent the temperature distribution in a material with heat conductivity β. If there are two or more materials present, then the coefficient β may be discontinuous across the interfaces between different materials. Physically, the temperature should be continuous, which means $[u] = 0$ across the interfaces, where $[\,\cdot\,]$ denotes the jump in a quantity. The heat flux βu_n across any interface should also be continuous if no heat source is present there. If β is discontinuous, then there must be a jump in the normal derivative u_n.

In the one-dimensional case, the heat equation is a mathematical model for heat conduction in a rod and we are looking at the case where the heat conduction coefficient changes abruptly at some point $x = \alpha$. The governing equation for the temperature $u(x)$ in dimensionless form is

$$u_t = (\beta u_x)_x + f(x,t)$$

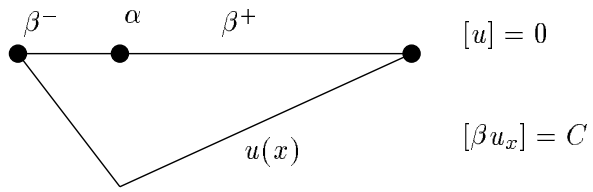
$$u(0,t) = u(1,t) = 0, \quad u(x,0) = g(x),$$

where $g(x)$ is the initial temperature and f is a heat source. We assume $0 < \alpha < 1$ and distinguish the following special cases:

- The source term $f(x,t)$ is continuous, but β is not. Then the heat flux is continuous, *i.e.*, $[\beta u_x] = 0$, but u_x is not.

- $f(x,t) = C(t)\,\delta(x-\alpha)$, in other words there is a singular heat source at a point α. So the heat flux at α now has a jump given by the source strength $C(t)$, *i.e.*, $[\beta\, u_x] = -C(t)$. In this case u_x is discontinuous even if β is continuous.

- The steady state problem: $(\beta u_x)_x = f(x)$. In particular suppose $f(x) = C\,\delta(x-\alpha)$ and β is constant on each side of the interface α. Then the solution is a piecewise linear function; see the following diagram. Even for this simplest example, special care has to be taken to deal with the discontinuity in β and the delta function singularity when we want to solve the problem numerically.



For a two-dimensional model, we consider two materials with different heat conductivity contacting on an interface, for example a circle as shown in Fig. 1.1 (a). Initially we assume the temperature is zero everywhere and that a heat source is applied at two boundaries. The mathematical description of the problem is the following:

$$
\begin{aligned}
&u_t = \nabla\cdot(\beta\nabla u), \qquad -1 \le x, y \le 1,\\
&\beta = \begin{cases} 1 & \text{if } x^2+y^2 \le \frac{1}{4}\\ 100 & \text{otherwise,}\end{cases}\\
&\text{BC: } u(-1,y,t) = u(x,-1,t) = 0;\\
&\qquad u(x,1,t) = \sin\left((x+1)\frac{\pi}{4}\right);\quad u(1,y,t) = \sin\left((y+1)\frac{\pi}{4}\right).\\
&\text{IC: } u(x,y,0) = 0.
\end{aligned}
$$

The heat propagates as time evolves and travels faster in the region with larger heat conductivity than the region with small one. Figure 1.1 (a) shows the contour plot of the temperature distribution at a short time while (b) is the mesh plot of the solution.

1.2 Overview of the interface problems considered in this thesis.

Interface problems are those problems in which *the input data (such as the coefficients of differential equations, source terms etc.) may be discontinuous or even singular across one or several interfaces which have lower dimension than that of the space where the problem is defined. The solution to the problem, therefore, may also be non-smooth or even discontinuous across those interfaces.* In this thesis we only consider those problems with smooth interfaces and bounded solutions. In the future we hope also to study problems

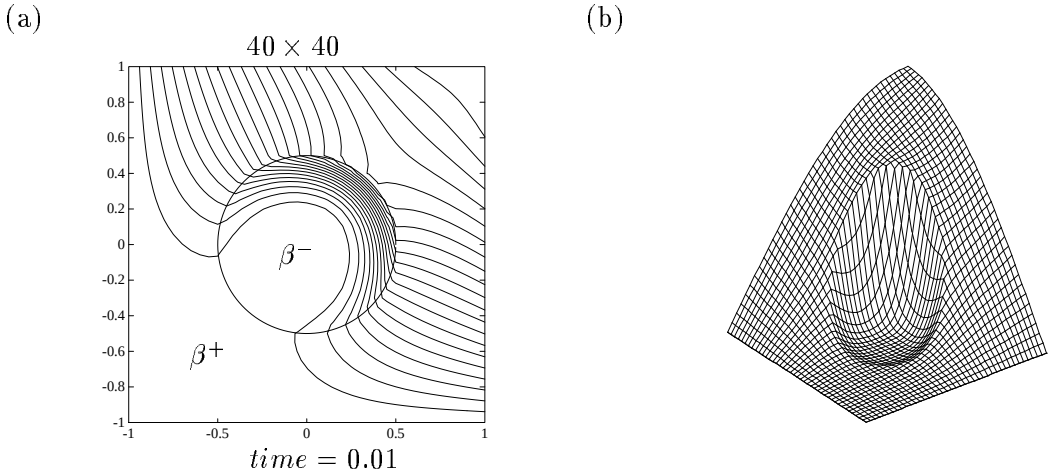


Figure 1.1: Heat propagation in different materials. (a) Contour plot of the temperature. (b) Mesh plot of the solution.

where the interface has corners or kinks and the solution has singular behavior near these points.

This thesis is concerned with the development of the immersed interface methods for the following classes of interface problems:

- *Elliptic equations corresponding to the steady state heat equations*

  $$\nabla \cdot (\beta \nabla u) + \kappa u = f$$

  in one, two and three (mostly two) space dimensions, where β, κ, and f may be discontinuous across some arbitrary interface(s), f can also be singular as in the form

  $$f = \int_\Gamma F(s)\delta(x - X(s))\, ds, \tag{1.2}$$

  where Γ is the interface. We will analyze some typical model problems to show how the immersed interface method works in Chapter 2. Various examples are also presented there to demonstrate the efficiency of the method.

- *Heat equations in 2D with fixed interfaces.* One class of problems is where the coefficients of the differential equation are continuous but the source term $f(x, y, t)$ corresponds to singular sources or dipoles along the interface. Then the solution may be non-smooth or discontinuous. We want not only to develop a difference methods for such problems but also to use the *Alternating Direction Implicit* (ADI) technique for better efficiency. In Chapter 3 we discuss how to modify the ADI scheme for such problems.

- *1D moving interface problems.* In many applications, the interface is moving with time [13], [15], [16], [17], [18], [19], [24]. An interesting class of such problems arises from modeling phase transitions, *e.g.*, solidification or melting. A simple example is

the classic Stefan problem modeling the interface between water and ice by a heat equation coupled with an equation for the motion of the interface. In this case there is a discontinuous heat conduction coefficient and also a singular heat source with support on the interface, due to latent heat released in the solidification process. In the one dimensional case, the equation takes the form

$$u_t = (\beta u_x)_x - c(t)\,\delta\,(x - \alpha(t))\,,$$

where the interface $\alpha(t)$ is determined by some non-linear ordinary differential equation

$$\frac{d\alpha}{dt} = w(t, \alpha, u, u_x, \cdots).$$

Various approaches have been used to solve Stefan problems numerically [13], [20], [24], [28]. Our immersed interface method seems to be a very promising approach for these problems. We discuss some one-dimensional problems with different interface conditions in Chapter 5.

- *The Stokes equations with a moving interface in 2D.* Consider a two-dimensional viscous incompressible fluid containing an immersed boundary. If the effects of inertia are negligible then the governing equations are the Stokes equations:

$$\begin{aligned} \nabla p &= \nu \Delta \vec{u} + \vec{F}(\vec{x}, t), \\ \nabla \cdot \vec{u} &= 0, \end{aligned}$$

where $\vec{u}$ is the fluid velocity, p is the fluid pressure, ν is the constant fluid viscosity, and $\vec{F}$ is the boundary force

$$\vec{F}(\vec{x}, t) = \int_{\Gamma(s,t)} \vec{f}(s, t)\delta_2(\vec{x} - \vec{X}(s, t))ds, \tag{1.3}$$

where $\vec{X}(s, t)$ is the Lagrangian representation of the immersed moving boundary, parameterized by s and $\vec{f}(s, t)$ is the force strength along the interface. For such problems the pressure will have a jump discontinuity across the interface and so will the normal derivative of the velocity. The interface moves with the fluid so it satisfies the equation

$$\frac{\partial \vec{X}}{\partial t}(s, t) = \vec{u}(\vec{X}(s, t), t). \tag{1.4}$$

We apply the immersed interface method for such problems in Chapter 4. Detailed analysis to handle the delta function and its derivative are given to derive the jump conditions needed for our immersed interface method. A quasi-Newton method with rank-2 updating is used to solve the nonlinear system of equations when the Crank-Nicholson implicit scheme is used for the stiff differential equation (1.4) determining the motion of the the elastic interface.

1.3 Our strategy and other approaches.

The Immersed Interface Method (IIM) takes various different forms for different problems. But there are some common characteristics which we list below.

1.3.1 The skeleton of the immersed interface method.

- Solve the PDE on a uniform Cartesian grid.
- Use finite difference methods.
- Immerse the interface(s) in the Cartesian grids.
- Apply the standard difference scheme at regular grid points which are away from interface(s).
- Derive the jump relations across the interface(s).
- Modify the difference scheme at irregular grid points which are near the interface by using the jump conditions.
- Solve the resulting system of difference equations to get the approximate solution of the differential equation.
- Estimate the global error through truncation error analysis.

We will discuss the details of each step in the coming chapters.

1.3.2 Why uniform Cartesian grid?

One of the most obvious advantages of using uniform Cartesian grids is that there is almost no cost for grid generation, and the conventional difference schemes can be used at most grid points (regular) which are away from the interface(s) since there are no irregularities there. Only those points near the interface(s), which are usually much fewer than the regular grid points, need special attention.

Certainly there are many other ways to discretize interface problems. Using a grid that conforms to the interface is an obvious alternative, for example a structured grid that is deformed in the neighborhood of the interface (e.g. [7]) or an unstructured triangulation. The finite element method on such a grid would be a natural choice for elliptic equations, and can be used very successfully (e.g., [3]). However, in many contexts the use of a uniform grid may be preferable.

In particular, for the equation

$$(\beta u_x)_x + (\beta u_y)_y = \int_\Gamma F(s)\delta\left(x - X(s)\right)\,\delta\left(y - Y(s)\right)\,ds \tag{1.5}$$

where Γ is an arbitrary interface, if β is constant then we will see that our modified difference equation uses the standard 5–point difference operator and only the right hand side of the

linear system is modified (see Chapter 2). This means that fast Poisson solvers can still be used to solve the system on a uniform grid, an advantage that would be lost on an irregular grid. Even if β is discontinuous so that the coefficients in the linear system must be modified, the system maintains the same block structure as in the continuous case. One can then use available software designed to accept a user–specified stencil on a uniform rectangular grid.

More importantly, we are interested primarily in time-dependent problems, and the interfaces are typically moving. Although it is possible to develop moving mesh methods that conform to the interfaces in each time step, this is generally much more complicated than simply allowing the interface to move relative to a fixed underlying uniform grid. For example, the immersed boundary method has been very successful in modeling flow in very complicated time-dependent geometries such as the beating heart with valves opening and closing (see the next section). This would be difficult if not impossible to do with grids that conform to the boundary.

1.3.3 Peskin's immersed boundary method.

A more complicated interface problem arises in using the *immersed boundary method* (IBM) to solve the incompressible Navier–Stokes equations in a region with complicated geometry. This method was originally developed by Peskin[41], [42] to model blood flow in the heart, and has since been used for many other problems, particularly in biophysics [5], [22], [25], [26]. The idea is to solve the Navier-Stokes equation on a uniform grid in a rectangular region in spite of the complicated time-varying geometry (see Fig 1.2), e.g., the heart wall. Without worrying about the boundary conditions, the Navier-Stokes equations then take the form

$$\begin{aligned} &\vec{u}_t + (\vec{u}\cdot\nabla)\,\vec{u} + \nabla p = \mu\Delta\vec{u} + \vec{F}, \\ &\nabla\cdot\vec{u} = 0, \\ &\frac{\partial\vec{X}}{dt} = \vec{u}, \end{aligned} \tag{1.6}$$

where $\vec{F}$ has the same form as (1.3). Now these equations are defined on the rectangular region. As in the Stokes equations we mentioned earlier, the boundary is viewed as being immersed in the fluid and moves with the local fluid velocity. Since the force is singular, the pressure is discontinuous and the velocity is not smooth across the interface. The force strength $\vec{f}(s)$ usually can be determined by physical reasoning, say by Hooke's law for the elastic interfaces.

This approach leads to an interface problem with a singular source. Peskin uses a discrete approach to solve such interface problem numerically. First he discretizes the immersed boundary by a set of Lagrangian points $(X(s_k), Y(s_k)),\ \ k = 1, 2, \cdots, m$ (see Fig 1.2), and replaces the integral in (1.3) by a discrete sum, also replacing the delta function by some discrete approximation $d_h(x)$ with support related to the mesh width h. Simple examples are the hat function

$$d_h(x) = \begin{cases} (h - |x|)/h^2 & \text{if } |x| < h \\ 0 & \text{if } |x| \geq h \end{cases} \tag{1.7}$$

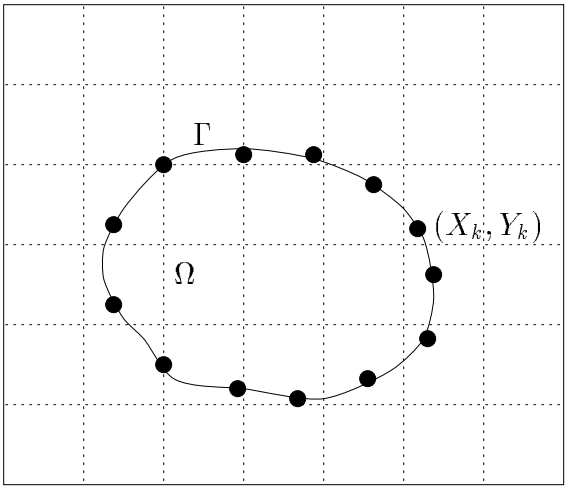


Figure 1.2: Peskin's approach: embedding the region in a rectangle with a uniform Cartesian grid.

and Peskin's discrete delta function

$$d_h(x) = \begin{cases} \frac{1}{4h}\left(1 + \cos\left(\pi x/2h\right)\right) & \text{if } |x| < 2h \\ 0 & \text{if } |x| \geq 2h. \end{cases} \tag{1.8}$$

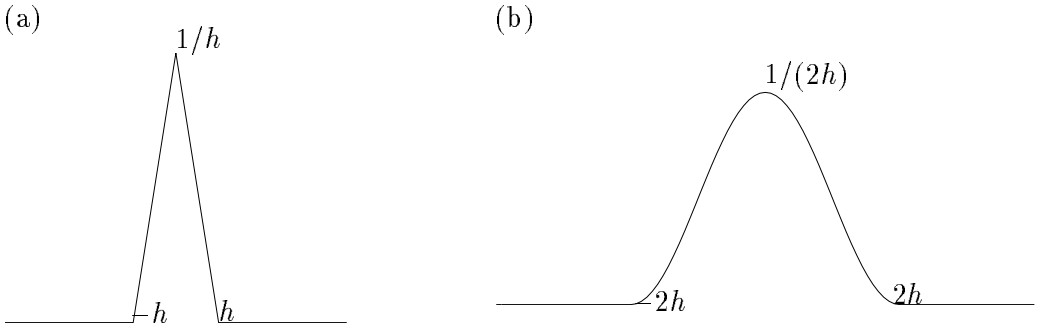


Figure 1.3: Two typical discrete delta functions. (a) Hat delta function. (b) Peskin's delta function.

Figure 1.3 is the plot of these two discrete delta functions. They are both continuous. The first one is not smooth but gives second order accuracy for many one dimensional problems [6]. Peskin's discrete delta function is smooth and usually is only first order accurate because it will also smooth the kinks in the solution if there are any.

With Peskin's discrete delta function approach the discrete form of (1.3) is

$$f_{ij} = \sum_{k=1}^{m} C(s_k)\, d_h(x_i - X_k)\, d_h(y_j - Y_k)\Delta s. \tag{1.9}$$

In one space dimension this approach is easy to analyze. In this case the interface reduces to a single point. For example, for the one-dimensional equation $u_{xx} = C\,\delta(x-\alpha)$, $x = \alpha \in$

$(0,1)$, the finite difference method

$$(u_{j+1} - 2u_j + u_{j-1})/h^2 = C\, d_h(x_j - \alpha) \tag{1.10}$$

with d_h given by (1.7) turns out to be very accurate; in fact it produces the exact solution $u = u(x_j)$ at all grid points in spite of the non-smoothness of u (see [6]). However if we use (1.8) to define d_h, we only end up with a first order accurate result.

Beyer and LeVeque[6] have also analyzed time-dependent versions of the problem and show that second order accuracy can still be obtained with an appropriate choice of the discrete delta function.

Our original motivation is to analyze and improve the results in [6] to two space dimensions and try to improve Peskin's method. However, it seems very unlikely that the discrete delta function approach can achieve second order accuracy except in a few special situations, *e.g.*, when the interface is aligned with the grid. An intuitive explanation why it can not be second order accurate in two space dimensions is that the expression of (1.9) as the discrete form of right hand side of (1.5) is independent of the derivative of $C(s)$ and the curvature of the interface, which seem to be crucial in obtaining second order accuracy (see Chapter 2).

1.3.4 Harmonic averaging.

Peskin's approach is designed for interface problems with singular delta functions. For problems with discontinuous coefficients, another approach to deriving the proper coefficients on a uniform grid stencil is the method of *harmonic averaging*, see [4], [46] and [50]. The one dimensional expression $(\beta u_x)_x$, for example, can be approximated by

$$\frac{1}{h^2}\left[\beta_{i+\frac{1}{2}}(u_{i+1} - u_i) - \beta_{i-\frac{1}{2}}(u_i - u_{i-1})\right].$$

If β is smooth then we can take $\beta_{i+\frac{1}{2}} = \beta(x_{i+\frac{1}{2}})$ (where $x_{i+\frac{1}{2}} = x_i + h/2$) and achieve second order accuracy. If β is discontinuous in $[x_{i-1}, x_{i+1}]$, then the coefficients can be chosen as harmonic averages of $\beta(x)$, e.g.,

$$\beta_{i+\frac{1}{2}} = \left[\frac{1}{h}\int_{x_i}^{x_{i+1}} \beta^{-1}(x)\, dx\right]^{-1}$$

This can be justified by homogenization theory for problems where $\beta(x)$ varies rapidly on the scale of the grid cells, and to some extent also for the case where β is simply discontinuous as we are considering, but the fact that this yields second order accurate results seems to be primarily the result of fortuitous cancellation. To see this, suppose that β has a discontinuity at α which lies between x_j and x_{j+1}. So x_j and x_{j+1} are two irregular grid points. The local truncation error defined as

$$T_i = (\beta u_x)_x - \frac{1}{h^2}\left[\beta_{i+\frac{1}{2}}(u_{i+1} - u_i) - \beta_{i-\frac{1}{2}}(u_i - u_{i-1})\right]$$

is only $O(1)$ at x_j and x_{j+1} which usually means the solution to the differential equation would only be first order accurate. But more careful investigation reveals that $T_j =$

$-T_{j+1} + O(h)$ and the cancellation to the order $O(h)$ is the key to the second order accuracy. However, we need to calculate the integrals to $O(h^3)$ at $i = j$ and $i = j + 1$, to guarantee the second order accuracy, which is not so easy especially in the interval where the discontinuity takes place.

In two space dimensions, harmonic averaging is also commonly used to deal with discontinuous coefficients [4], [46], now integrating over squares to obtain the harmonic average of $\beta(x, y)$. In this case, however, the method does not appear to give second order accurate results because the cancellations are very unlikely to take place for arbitrary interfaces. It is also not practical to compute the integrals to $O(h^3)$ at irregular grid points in two dimensions especially when β is discontinuous across the interface. We find that our approach is greatly superior.

1.3.5 Other approaches.

Because of the many advantages of the uniform Cartesian grid, there has been a lot of research done using uniform Cartesian grids to solve interface or non-interface problems. Tikhonov and Samarskii[50] discuss one-dimensional elliptic interface problems and derived second order accurate methods on uniform grids using jump conditions at a point of discontinuity in the coefficients. In two dimensions, Mayo [35] has considered similar problems and shown how standard difference formulas can be modified to obtain second order accuracy in the context of solving Poisson or biharmonic equations on irregular regions by solving some integral equation. The region is embedded in a regular region where a fast solver can be used on a uniform grid and the right hand side is appropriately modified near the original boundary. Mayo and Greenbaum[38] consider an interface problem in magneto-statics of the form (1.5) with a piecewise constant coefficient β. The possibility of extension to variable β is mentioned in [36]. Li and Mayo [32] discuss difference methods to solve the heat equations with fixed interfaces. Mayo [37] studied the stationary Stokes equations with immersed interfaces in terms of Cauchy integrals.

MacKinnon and Carey[33] also use a similar approach in one dimension and make some extensions to two-dimensional problems in which the interface lies along a coordinate direction. Fornberg and Meyer-Spasche[27] have considered elliptic equations with free boundaries which are solved on a uniform grid by adding correction terms near the interface to improve the accuracy.

1.4 Mathematical aspects of the interface problems.

1.4.1 Interface conditions.

Generally, the interface problems with bounded solutions can be decomposed into one or several regions whose size and shape may change with time. The solutions in different regions are continuously differentiable to a certain degree and they are coupled by some interface conditions, usually jump relations across the interfaces. It is crucial for our approach that we be able to derive the interface conditions directly from the differential equations or other known data. Sometimes this turns out to be a challenge.

Sometimes we can derive the interface conditions from the differential equations themselves. In the equation $(\beta u_x)_x = C\,\delta(x - \alpha)$, for example, the jump relations $[u] = 0$

and $[\beta\, u_x] = C$ can easily be obtained from the equation itself. With a little effort we can prove that the jump relations for equation (1.5) are $[u] = 0$ and $[u_n] = F(s)$ at each point $(X(s), Y(s))$ on the $\partial\Omega$. More complicated examples such as the Stokes equations can be found in § 4.2.

Very often we deal with a physical application in which we have enough information to determine the interface relations. For instance, in the example of heat propagation, we know the temperature is continuous and the heat flux has to be zero across the interface, so we have the interface relations $[u] = 0$ and $[\beta u_n] = 0$ at every point of the interface. For the ice melting problem, as another example, the value of the temperature on the interface is known to be zero.

1.4.2 The classification of interface problems.

There are many kinds of interface problems. In this thesis, we roughly distinguish them by the following classifications.

A: Classification by the structure of the differential equations.

- The coefficients of the differential equations are continuous, but there are singular source terms as in equations (1.5) and (1.6).
- The coefficients are discontinuous along the interface but no singular source terms are present. The heat propagation is a typical example.
- A combination of the cases above.

B: Classification by the structure of the interface.

- The interface is fixed. The problem may or may not be time dependent.
- The interface is moving in a time-dependent problem.
- There is more than one interface.

1.4.3 Numerical difficulties.

The most noticeable characteristic of an interface problem is the discontinuity or non-smoothness in the solution which is the result of discontinuities of the coefficients or singularities of the sources in the corresponding differential equations. This brings up several substantial difficulties in the numerical analysis process for interface problems.

- *Discretization.* Special care has to be taken to discretize the discontinuous coefficients. Many current techniques such as *harmonic averaging* or *coefficient smoothing* [48] fail to give high order accuracy in two or higher space dimensions.

  If singular sources are present such as delta functions or dipoles, it is not clear what the best way is to discretize them to achieve the desired accuracy in two or higher space dimensions.

- *Arbitrary interfaces.* Generally the interfaces can be arbitrary and complicated, and analytical expressions for them are rarely available. Moreover, there are situations in which the interface may develop cusps and spikes, change topology, and break or merge.

- *Error analysis.* Because of the discontinuity and non-smoothness in the solution and the complexity of the interface, it is difficult to perform convergence analysis in the conventional way.

- *Solving the system of discrete equations.* Due to the presence of the interface and the discontinuity or non-smoothness in the solution, the system of discrete equations may lose many nice properties such as symmetry, positive definiteness, and diagonal dominance etc. The structure of the linear system may be very different from regular problems making it hard to use multi-grid or other efficient solvers.

1.4.4 Interface expressions.

To solve interface problems numerically, we need the information about the interface such as the position, tangential and normal directions, and sometimes curvature as well. Some common approaches to express the interface are the following.

- *Analytic expression.* If the interface is fixed, we may have an analytic expression for the interface. However, it can still be difficult to calculate other information needed such as the first derivative to determine the tangential and normal directions, and second derivative to determine the curvature, etc., if the analytic expression is too complicated. Then a discrete method to calculate those quantities to a certain accuracy is needed.

- *Discrete parameterization and interpolation.* Very often we only know the coordinates of a number of control points on the interface, say (X_k, Y_k), $k = 1, 2, \cdots$, in two space dimensions. There are two ways to get derivative information on the interface.

  The first approach is to use discrete difference formulas such as the central difference to get the required derivatives. This approach has been widely used in implementing the immersed boundary method for many problems. However we must balance the needs of accuracy and stability in this approach. Usually higher order accurate difference formula, or too many control points, will destabilize the algorithm and worsen the condition of the resulting linear system of equations. This approach seems to be unable to handle the situations when the interface develops cusps and spikes or when the interface breaks or merges.

  We have used a different approach in our numerical method. First we use piecewise interpolation, mostly cubic splines, to get an analytic expression of the interface. Then we calculate all the information about the interface from the analytic expression of the interpolated interface. This approach works very well for many test problems including Stokes flow with a moving interface. One advantage is that we can take relatively few control points on the interface if the interface is smooth. Moreover,

other quantities such as the force strength, jumps etc., can be calculated with the same parameter as used in the interpolation formula (see Chapter 4). Although it may be difficult to implement, this approach can handle cusps and spikes and even situations when the interface breaks or merges.

- *Level set approach.* In this approach, the interface is modeled as the zero set of a smooth function ϕ defined on the entire physical domain. The boundary is then moved by solving a nonlinear equation of Hamilton-Jacobi type on the whole domain. This approach was introduced by Osher and Sethian in [40] and has been used for many moving interface problems (*e.g.*, [11], [12], [48], [53]) since then.

  This approach does not rely on a discrete parameterization of the interface and can be used for complicated moving interfaces in two and three dimensions. It can handle cusps and spikes and situations in which the interfaces break or merge. The disadvantage of this approach is that it requires solving the Hamilton-Jacobi equation on all grid points and comparing the signs of the level set function to determine the interface while the interpolation technique only requires tracking the control points, which are much fewer than the number of Cartesian grid points. Another difficulty of this method is to extend certain quantities only defined on the interface to the whole domain.

  So depending on the knowledge of the physical problem, we can choose a suitable method to express the interface. We have not tried to implement the level set approach for our immersed interface method and that is something we are going to do in the near future. David Chopp in the Mathematics Department, at the University of Washington, is currently trying to solve the potential flow problems with free fluid interfaces using the immersed interface method described in this thesis and level set techniques.

1.5 Other applications.

In this section we mention a few more interesting applications of interface problems.

A Poisson problem with discontinuous coefficients is a fundamental problem in various important applications, for example at the interface between two materials with different diffusion parameters in steady state heat diffusion or electrostatic problems. Such problems also arise in multicomponent flow problems, *e.g.*, the porous media equations used to model the interface between oil and injected fluid in simulations of secondary recovery in oil reservoirs [2], [4], [46]. The immersed interface method is derived for such elliptic problems in Chapter 2.

Discontinuous coefficient problems also can be found in hyperbolic equations, for example in wave propagation through non-homogeneous media with discontinuities in the propagation speed. For example, solving inverse problems in oil exploration seismology requires a good technique for solving the forward problem, which is typically a hyperbolic equation with discontinuous coefficients, or wave speeds, at geological interfaces. Progress has been made for solving such problems in one or two space dimensions with fixed interfaces by Chaoming Zhang, a Ph.D. student of Randy LeVeque, using the immersed interface method described in [31] and this thesis.

Domain embedding. Sometimes problems on irregular regions can be handled more easily as interface problems by embedding the region into a rectangular domain and then solving the equation on a Cartesian grid in the rectangle. The original boundary then becomes an interface. The original application of Peskin's immersed boundary method used this approach; the fluid dynamics problem within the heart was extended to a flow problem over a rectangle.

As another example, suppose we want to solve an elliptic equation on an irregular region Ω. We can embed the region in a larger rectangular domain R. For example, we could solve the Dirichlet problem

$$\begin{aligned} u_{xx} + u_{yy} &= 0 \quad \text{in} \quad \Omega, \\ u &= g \quad \text{on} \quad \partial\Omega, \end{aligned}$$

by extending it to the problem

$$\begin{aligned} u_{xx} + u_{yy} &= \int_{\partial\Omega} F(s)\delta\left(x - X(s)\right)\delta\left(y - Y(s)\right) ds \quad \text{in} \quad R, \\ u &= 0, \qquad\qquad\qquad\qquad \text{on} \quad \partial R. \end{aligned} \tag{1.11}$$

The problem is then to determine $F(s)$ so that the condition $u = g$ on $\partial\Omega$ is satisfied. The solution is still continuous on the enlarged region R, but not smooth across the interface $\partial\Omega$.

This particular problem has been extensively studied in the past and a number of domain embedding procedures have been developed, *e.g.*, capacitance methods [10], [21], [35], [45] and methods based on solving integral equations along $\partial\Omega$. Of particular note is the method of Mayo [35] since, after solving an integral equation for the source strengths $F(s)$, she then uses the resulting jumps in derivatives across $\partial\Omega$ to determine the right hand side in the Poisson problem (1.11) using a technique that is very closely related to our method in this case.

With the immersed interface idea, we can also develop an embedding technique to solve elliptic equations on complicated regions with Dirichlet boundary conditions. This is described in §2.5, see also Appendix B.

Chapter 2

THE IMMERSED INTERFACE METHOD FOR ELLIPTIC EQUATIONS WITH DISCONTINUOUS COEFFICIENTS AND SINGULAR SOURCES

In this chapter we develop the immersed interface method for elliptic equations of the form

$$\nabla \cdot (\beta \nabla u) + \kappa u = f \tag{2.1}$$

in a domain Ω in one, two, or three space dimensions. Within the region Ω, suppose there is an irregular surface of codimension 1 (hereafter called an interface) across which the function u or some of its derivatives are known to be discontinuous. For simplicity we assume that Ω is a simple domain, such as a square in two dimensions or a solid rectangle in three dimensions. We wish to solve the equation using a finite difference method on a regular grid, e.g., a uniform Cartesian grid. The interface is typically not aligned with the grid but rather cuts between grid points so that for grid points near the interface the stencil of a standard finite difference method will contain points from both sides of the interface. Because of the nonsmoothness of u, differencing u across the interface using standard difference formulas will not produce accurate approximations to derivatives of u, and hence a naive discretization will produce results with low accuracy.

In order for discontinuities to arise in the solution or its derivatives, there must be discontinuities or singularities present in the coefficients of the equation. Suppose, for example, that the function β is discontinuous across the interface, while κ and f are continuous. Then u and $\beta \partial u/\partial n$ will be continuous while the normal derivative $\partial u/\partial n$ will be discontinuous. Such problems arise frequently in practical applications as we mentioned in Chapter 1.

Here we try to derive modified difference equations for a quite general problem of the form (2.1), which produce second order accurate results on a uniform grid in one, two, or three dimensions. Taking the two dimensional case as an example, we derive appropriate coefficients at the grid points on a stencil that contains at most six points: the points of the standard five-point stencil plus a sixth point if we are near the interface which is chosen from the set of diagonally adjacent grid points. The coefficients at these points can be determined by solving a system of six linear equations.

Instead of discontinuities in β, another possibility is that β is continuous but that the source term f has a delta function singularity along the interface Γ, e.g., in two dimensions

$$f(x,y) = \int_\Gamma C(s)\, \delta(x - X(s))\, \delta(y - Y(s))\, ds, \tag{2.2}$$

where $(X(s), Y(s))$ is the arc length parameterization of Γ and $C(s)$ is the source strength. By this we mean that $f(x,y)$ is a distribution with the property that

$$\iint f(x,y)\, \phi(x,y)\, dxdy = \int_\Gamma C(s)\, \phi(X(s), Y(s))\, ds$$

for any smooth test function $\phi(x, y)$. Again the solution u will be continuous but the normal derivative will have a discontinuity of magnitude $C(s)$. As a model problem, consider the heat conduction problem in which a heat source is applied only along Γ. The temperature u will be highest along Γ, falling off on either side resulting in a jump discontinuity in the normal derivative. In this case the standard five-point stencil can be used, but we must derive an appropriate term on the right hand side to model the singular source. A dipole source may also occur, in which f contains the derivative of the delta function, and as a result the solution u itself is discontinuous across Γ. Again we can derive the appropriate right hand side f_{ij} at each grid point so that the solution to the finite difference equations is second order accurate in spite of the discontinuities.

More generally we can handle discontinuities in β, κ, and f simultaneously with delta function or dipole sources. A general procedure for deriving the coefficients in the stencil and the right hand side is presented below. All that is required is *a priori* knowledge about the jumps in derivatives of u across Γ. For the above examples, sufficient information can be derived from the equation itself, without *a priori* knowledge of the solution.

2.1 One-dimensional problems

We begin by considering the one-dimensional problem

$$(\beta u_x)_x + \kappa u = f + C\delta(x - \alpha) \tag{2.3}$$

on the interval $[0, 1]$ with specified boundary conditions on u at $x = 0$ and $x = 1$. The function $\beta(x)$ is allowed to be discontinuous at $x = \alpha$. For simplicity we will assume that $\kappa(x)$ and $f(x)$ are smooth functions, although discontinuities in these functions could also be handled with a minor modification of what follows.

We also allow an additional constraint to be imposed on the solution, namely that the function u should have a jump discontinuity at $x = \alpha$ of specified strength $\hat{C}$,

$$[u] = u^+ - u^- = \hat{C}. \tag{2.4}$$

This could be incorporated into the equation (2.3) by including a dipole source term proportional to the derivative of the delta function, changing (2.3) to

$$(\beta u_x)_x + \kappa u = f + C\delta(x - \alpha) + \frac{1}{2}(\beta^- + \beta^+)\hat{C}\delta'(x - \alpha). \tag{2.5}$$

For simplicity, however, we leave this as an external constraint.

By integrating (2.3) across the discontinuity, we find that βu_x has a jump of magnitude C,

$$[\beta u_x] = \beta^+ u_x^+ - \beta^- u_x^- = C. \tag{2.6}$$

An alternative way to state the problem is to require that u satisfy the equation

$$(\beta u_x)_x + \kappa u = f \tag{2.7}$$

in each of the intervals $(0, \alpha)$ and $(\alpha, 1)$, together with the two internal boundary conditions (2.4) and (2.6) at $x = \alpha$.

We now wish to approximate the solution $u(x)$ on a uniform grid in the interval $[0,1]$, with

$$x_i = ih, \quad i = 1, 2, \cdots, n$$

where $h = 1/n$. The point α will typically fall between grid points, say $x_j \le \alpha < x_{j+1}$, Our goal is to develop finite difference equations of the form

$$\gamma_{i,1} u_{i-1} + \gamma_{i,2} u_i + \gamma_{i,3} u_{i+1} + \kappa_i u_i = f_i + C_i \quad i = 1, 2, \cdots, n-1 \tag{2.8}$$

that can be used together with the boundary data u_0 and u_n to obtain a second order accurate approximation to $u(x)$ at the uniform grid points.

For $i \neq j, j+1$ the solution u is smooth in the interval $[x_{i-1}, x_{i+1}]$ and we can use the standard approximation

$$\frac{1}{h^2}\left(\beta_{i+\frac{1}{2}}(u_{i+1} - u_i) - \beta_{i-\frac{1}{2}}(u_i - u_{i-1})\right) + \kappa_i\, u_i \;= f_i, \tag{2.9}$$

where

$$\beta_{i+\frac{1}{2}} = \beta(x_{i+\frac{1}{2}}),\ \kappa_i = \kappa(x_i), \quad f_i = f(x_i).$$

In this case we can take

$$\begin{aligned} &\gamma_{i,1} = \beta_{i-\frac{1}{2}}/h^2, \quad \gamma_{i,2} = -(\beta_{i-\frac{1}{2}} + \beta_{i+\frac{1}{2}})/h^2, \\ &\gamma_{i,3} = \beta_{i+\frac{1}{2}}/h^2 \quad \text{and} \quad C_i = 0. \end{aligned} \tag{2.10}$$

This gives a local truncation error that is $O(h^2)$:

$$T_i = \gamma_{i,1} u(x_{i-1}) + \gamma_{i,2} u(x_i) + \gamma_{i,3} u(x_{i+1}) + \kappa_i u(x_i) - f_i = O(h^2). \tag{2.11}$$

We wish to determine formulas of the form (2.8) for $i = j$ and $i = j+1$ so that second order global accuracy is obtained. Since only two grid points are involved (independent of h), it is sufficient to have an $O(h)$ local truncation error at those points.

To compute the local truncation error at the point x_j, we expand u_{j-1}, u_j, and u_{j+1} in Taylor series about the point $x = \alpha$. Since we expect the γ coefficients to be $O(1/h^2)$ we must expand out through $O(h^3)$ in order to ensure an $O(h)$ truncation error. We use the notation

$$u^- = \lim_{x \to \alpha -} u(x), \qquad u^+ = \lim_{x \to \alpha +} u(x),$$

and expand to obtain

$$u(x_{j-1}) = u^- + (x_{j-1} - \alpha)\, u_x^- + \frac{1}{2}(x_{j-1} - \alpha)^2\, u_{xx}^- + O(h^3) \tag{2.12}$$

$$u(x_j) = u^- + (x_j - \alpha)\, u_x^- + \frac{1}{2}(x_j - \alpha)^2\, u_{xx}^- + O(h^3) \tag{2.13}$$

$$u(x_{j+1}) = u^+ + (x_{j+1} - \alpha)\, u_x^+ + \frac{1}{2}(x_{j+1} - \alpha)^2\, u_{xx}^+ + O(h^3). \tag{2.14}$$

Note if $x_j = \alpha$, then $u(x_j)$ is defined as the limit of the $u(x)$ approaching from the left. The corresponding u_j then is the approximation to this specific limit. We also use

$$\kappa_j u(x_j) = \kappa(\alpha)\, u^-(\alpha) + O(h) \quad \text{and} \quad f_j = f(\alpha) + O(h) \tag{2.15}$$

The expression for $u(x_{j+1})$ involves u^+, u_x^+ and u_{xx}^+ at α^+. However, using the known jump relations we can replace these by values at α^-. This will allow us to use the PDE (2.3) to determine the γ coefficients. From (2.4) and (2.6) we have

$$\begin{aligned} u^+ &= u^- + \hat{C}, \\ u_x^+ &= (\beta^- u_x^- + C)/\beta^+. \end{aligned}$$

From the equation (2.7) we also see that $(\beta u_x)_x + \kappa u$ is continuous at $x = \alpha$, since f is, and so

$$\beta_x^+ u_x^+ + \beta^+ u_{xx}^+ + \kappa u^+ = \beta_x^- u_x^- + \beta^- u_{xx}^- + \kappa u^-$$

and hence

$$u_{xx}^+ = \frac{1}{\beta^+} \left(\beta^- u_{xx}^- + \left(\beta_x^- - \frac{\beta_x^+ \beta^-}{\beta^+} \right) u_x^- - \frac{\beta_x^+}{\beta^+} C - \kappa \hat{C} \right) \tag{2.16}$$

Using these expressions in (2.14) gives

$$\begin{aligned} u(x_{j+1}) &= u^- + \left[\frac{\beta^-}{\beta^+} (x_{j+1} - \alpha) + \left(\frac{\beta_x^-}{\beta^+} - \frac{\beta_x^+ \beta^-}{(\beta^+)^2} \right) \frac{(x_{j+1} - \alpha)^2}{2} \right] u_x^- \\ &\quad + \frac{(x_{j+1} - \alpha)^2 \beta^-}{2\beta^+} u_{xx}^- + \hat{C} \\ &\quad + (x_{j+1} - \alpha) \frac{C}{\beta^+} - \frac{(x_{j+1} - \alpha)^2}{2} \left(\frac{\beta_x^+}{\beta^+} C + \kappa \hat{C} \right). \end{aligned} \tag{2.17}$$

In computing the local truncation error we also use the PDE (2.3), which in approaching α from the left gives

$$\beta_x^- u_x^- + \beta^- u_{xx}^- + \kappa(\alpha) u^- = f(\alpha). \tag{2.18}$$

We use this to replace the $f(\alpha)$ term in the local truncation error, obtaining

$$\begin{aligned} T_j &= \gamma_{j,1} u(x_{j-1}) + \gamma_{j,2} u(x_j) + \gamma_{j,3} u(x_{j+1}) + \kappa(\alpha) u^- \\ &\quad - [\beta_x^- u_x^- + \beta^- u_{xx}^- + \kappa u^-] - C_j + O(h) \end{aligned} \tag{2.19}$$

Replacing $u(x_{j-1}), u(x_j)$ and $u(x_{j+1})$ by the expressions (2.12)–(2.14) and collecting together terms then gives

$$\begin{aligned} T_j &= (\gamma_{j,1} + \gamma_{j,2} + \gamma_{j,3}) u^- + \left\{ (x_{j-1} - \alpha)\gamma_{j,1} + (x_j - \alpha)\gamma_{j,2} \right. \\ &\quad \left. + \left(\frac{\beta^-}{\beta^+} (x_{j+1} - \alpha) + \left(\frac{\beta_x^+}{\beta^+} - \frac{\beta_x^+ \beta^-}{(\beta^+)^2} \right) \frac{(x_{j+1} - \alpha)^2}{2} \right) \gamma_{j3} - \beta_x^- \right\} u_x^- \\ &\quad + \frac{1}{2} \left\{ (x_{j-1} - \alpha)^2 \gamma_{j,1} + (x_j - \alpha)^2 \gamma_{j,2} + (x_{j+1} - \alpha)^2 \frac{\beta^-}{\beta^+} \gamma_{j3} - 2\beta^- \right\} u_{xx}^- \\ &\quad + \gamma_{j,3} \left\{ \hat{C} + (x_{j+1} - \alpha) \frac{C}{\beta^+} - \frac{(x_{j+1} - \alpha)^2}{2} \left(\frac{\beta_x^+ C}{(\beta^+)^2} + \kappa \frac{\hat{C}}{\beta^+} \right) \right\} - C_j + O(h) \end{aligned} \tag{2.20}$$

We can ensure that $T_j = O(h)$ by requiring that each coefficient of u^-, u_x^-, u_{xx}^- vanishes, as well as the constant term. This gives four equations for the four unknowns $\gamma_{j,1}, \gamma_{j,2}, \gamma_{j,3}$ and C_j. The first three equations gives a linear system for the γ's:

$$\begin{aligned}
\gamma_{j,1} + \gamma_{j,2} + \gamma_{j,3} &= 0 \\
(x_{j-1} - \alpha)\gamma_{j,1} + (x_j - \alpha)\gamma_{j,2} & \\
+ \left\{ \frac{\beta^-}{\beta^+}(x_{j+1} - \alpha) + \left(\frac{\beta_x^-}{\beta^+} - \frac{\beta^- \beta_x^+}{(\beta^+)^2} \right) \frac{(x_{j+1} - \alpha)^2}{2} \right\} \gamma_{j,3} &= \beta_x^- \qquad (2.21) \\
\frac{(x_{j-1} - \alpha)^2}{2}\gamma_{j,1} + \frac{(x_j - \alpha)^2}{2}\gamma_{j,2} + \frac{(x_{j+1} - \alpha)^2 \beta^-}{2\beta^+}\gamma_{j,3} &= \beta^-
\end{aligned}$$

Once these γ's have been computed, we then set

$$C_j = \gamma_{j,3} \left\{ \hat{C} + (x_{j+1} - \alpha) \frac{C}{\beta^+} - \frac{1}{2}(x_{j+1} - \alpha)^2 \left(\frac{\beta_x^+ C}{\beta^{+2}} + \kappa \frac{\hat{C}}{\beta^+} \right) \right\}. \qquad (2.22)$$

In a similar way, we can compute the coefficients in the equation at x_{j+1} from the system

$$\begin{aligned}
\gamma_{j+1,1} + \gamma_{j+1,2} + \gamma_{j+1,3} &= 0 \\
\left\{ \frac{\beta^+}{\beta^-}(x_j - \alpha) + \left(\frac{\beta_x^+}{\beta^-} - \frac{\beta_x^- \beta^+}{(\beta^-)^2} \right) \frac{(x_j - \alpha)^2}{2}) \right\} \gamma_{j+1,1} & \\
+ (x_{j+1} - \alpha)\gamma_{j+1,2} + (x_{j+2} - \alpha)\gamma_{j+1,3} &= \beta_x^+ \qquad (2.23) \\
\frac{(x_j - \alpha)^2}{2}\frac{\beta^+}{\beta^-}\gamma_{j+1,1} + \frac{(x_{j+1} - \alpha)^2}{2}\gamma_{j+1,2} + \frac{(x_{j+2} - \alpha)^2}{2}\gamma_{j+1,3} &= \beta^+
\end{aligned}$$

and then

$$C_{j+1} = \gamma_{j+1,1} \left\{ -\hat{C} + (\alpha - x_j)\frac{C}{\beta^-} - \frac{1}{2}(\alpha - x_j)^2 \left(\frac{\beta_x^- C}{\beta^{-2}} - \kappa \frac{\hat{C}}{\beta^-} \right) \right\}. \qquad (2.24)$$

In the particular case when $\beta_x^- = 0$ and $\beta_x^+ = 0$ (in particular if β is piecewise constant), we can easily get explicit expressions for the γ_j's. Setting

$$\begin{aligned}
D_j &= h^2 + [\beta](x_{j-1} - \alpha)(x_j - \alpha)/2\beta^-, \\
D_{j+1} &= h^2 - [\beta](x_{j+2} - \alpha)(x_{j+1} - \alpha)/2\beta^+,
\end{aligned}$$

these can be written as:

$$\begin{aligned}
\gamma_{j,1} &= (\beta^- - [\beta](x_j - \alpha)/h)/D_j & \gamma_{j+1,1} &= \beta^-/D_{j+1} \\
\gamma_{j,2} &= (-2\beta^- + [\beta](x_{j-1} - \alpha)/h)/D_j & \gamma_{j+1,2} &= (-2\beta^+ + [\beta](x_{j+2} - \alpha)/h)/D_{j+1} \\
\gamma_{j,3} &= \beta^+/D_j & \gamma_{j+1,3} &= (\beta^+ - [\beta](x_{j+1} - \alpha)/h)/D_{j+1}
\end{aligned}$$

provided that $D_j,\ D_{j+1} \neq 0$. In practical problems β often represents a physical quantity such as conductivity, permeability, or density and so $\beta > 0$ everywhere. In this case we have the following theorem on the solvability of the linear systems for the γ coefficients.

Theorem 2.1 *Suppose $\beta^-\beta^+ > 0$. If $\beta_x^- = 0$ and $\beta_x^+ = 0$ (which includes the special case in which $\beta(x)$ is piecewise constant), then equations (2.21) and (2.23) have unique solutions. More generally, these systems are guaranteed to have unique solutions for h sufficiently small.*

Proof: It is enough to prove that equations (2.21) have a unique solution. The proof for equations (2.23) is identical. Let A be the determinant of the coefficient matrix of the system (2.21). With the help of *Mathematica*, a symbolic software package, we calculate that:

$$A = \begin{cases} C_1\, D_j & \text{if } \beta_x^-(\alpha) = 0 \text{ and } \beta_x^+(\alpha) = 0, \\ \\ C_1\ (D_j + O(h^3)) & \text{otherwise,} \end{cases}$$

where C_1 is a nonzero constant. Without loss of generality, we supporse $\beta^- > 0$ and $\beta^+ > 0$. Notice that $(x_{j-1} - \alpha)(x_j - \alpha) \geq 0$, and h is small, so if $[\beta] \geq 0$, then $|A| > |C_1|h^2 > 0$, hence the theorem is true. If $[\beta] < 0$, *i.e.*, $\beta^- > \beta^+$, then since $\beta^-\beta^+ > 0$ we have:

$$\left| \frac{\beta^- - \beta^+}{\beta^-} \right| < 1.$$

Hence:

$$\begin{aligned} h^2 + \frac{[\beta]}{2\beta^-}(x_{j-1} - \alpha)(x_j - \alpha) &> h^2 - \frac{1}{2}(x_{j-1} - \alpha)(x_j - \alpha) \\ &> h^2 - \frac{1}{2}(2h)h = 0, \end{aligned}$$

where we have used $x_j \leq \alpha < x_{j+1}$. The rest of the proof is trivial. □

If $\beta^+\beta^- < 0$ then the systems may be singular, although generically they are still nonsingular. Note that in this case it would be possible to multiply the equation by -1 on one side of α, yielding a problem with $\beta^-\beta^+ > 0$ at the possible expense of introducing discontinuities in κ and f. These discontinuities can easily be handled as described below. In this case one must be careful with the jump conditions — the jump conditions for the original equation must be imposed and not the jump conditions for the modified β.

Note also the following properties and special cases of the γ coefficients that result from solving these systems:

- The γ coefficients depends only on the function $\beta(x)$ and the position of α relative to the grid, and not on C or $\hat{C}$.

- If β is constant, then solving the systems (2.21) and (2.23) we recover the standard coefficients $\gamma_{i1} = \gamma_{i3} = \beta/h^2$ and $\gamma_{i2} = -2\beta/h^2$ for $i = j,\ j+1$.

- If β is continuous, then the standard coefficients (2.10) satisfy the system (2.21) to $O(h)$.

- In the case when β is piecewise constant, the harmonic averaging coefficients satisfy the first two equations of (2.21) but not the third, indicating that the truncation error of this method at x_j and x_{j+1} is $O(1)$. But one can prove that due to cancellation of errors this method is still second order accurate (see §1.3.4).

- If $C = \hat{C} = 0$, then $C_j = C_{j+1} = 0$ and the inhomogeneous term in the difference equation is simply f_i. In this case a discontinuity in β affects only the coefficients and not the right hand side.

- If β is constant and $\kappa = 0$, then

$$C_j = \frac{1}{h^2}(x_{j+1} - \alpha)C + \frac{\beta}{h^2}\hat{C} = C d_h(x_j - \alpha) + \hat{C}\beta d_h'(x_j - \alpha) \tag{2.25}$$

  where d_h is the hat function (1.7). In this case we can view the difference scheme as a direct discretization of the equation

$$\beta u''(x) = f(x) + C\delta(x - \alpha) + \hat{C}\beta\delta'(x - \alpha).$$

The general one-dimensional problem. Now we suppose that f and κ may also have discontinuities at α. We only need a slight change in the linear systems for the γ_{jk}s and $\gamma_{j+1,k}$s and the corrections C_j and C_{j+1} to get the correct difference schemes at the grid points x_j, x_{j+1}.

At the grid point x_j, the first equation of the linear system (2.21) becomes

$$\gamma_{j,1} + \gamma_{j,2} + \left(1 - \frac{(x_{j+1} - \alpha)^2}{2\beta^+}[\kappa]\right)\gamma_{j,3} = 0 \tag{2.26}$$

and the correction term now is

$$C_j = \gamma_{j,3}\left\{\hat{C} + (x_{j+1} - \alpha)\frac{C}{\beta^+} - \frac{(x_{j+1} - \alpha)^2}{2}\left(\frac{\beta_x^+ C}{(\beta^+)^2} + \kappa^+\frac{\hat{C}}{\beta^+} - \frac{[f]}{\beta^+}\right)\right\}. \tag{2.27}$$

At the grid x_{j+1}, the first equation of the linear system (2.23) becomes

$$\left(1 + \frac{(x_j - \alpha)^2}{2\beta^-}[\kappa]\right)\gamma_{j+1,1} + \gamma_{j+1,2} + \gamma_{j+1,3} = 0 \tag{2.28}$$

and the correction term is

$$C_{j+1} = \gamma_{j+1,1}\left\{-\hat{C} + (\alpha - x_j)\frac{C}{\beta^-} - \frac{(\alpha - x_j)^2}{2}\left(\frac{\beta_x^- C}{(\beta^-)^2} - \kappa^-\frac{\hat{C}}{\beta^-} + \frac{[f]}{\beta^-}\right)\right\}. \tag{2.29}$$

2.2 A simple two-dimensional problem

In order to introduce the ideas used in two dimensions in a simple framework, we begin by considering the equation

$$(\beta u_x)_x + (\beta u_y)_y + \kappa(x, y)\, u = f(x, y) \qquad (x, y) \in \Omega \tag{2.30}$$

in the case where β is piecewise constant and has a jump discontinuity across some curve Γ in Ω, while κ and f are assumed to be smooth. Formulas for the more general case, in which κ and f may be discontinuous, f may contain singular forces, and we may also require a discontinuity in the solution u, will be presented in §2.3.

The interface Γ can be an arbitrary piecewise smooth curve lying in Ω. We need not assume that Γ is closed or even connected. It may consist of several segments.

We assume the domain Ω is a square, say $[a,b] \times [a,b]$. We take a uniform grid with

$$x_i = a + ih, \; y_j = a + jh, \;\; i,j = 0, 1, \cdots, n$$

where $h = (b-a)/n$. Figure 2.1 gives an example of the uniform grid and the immersed interface.

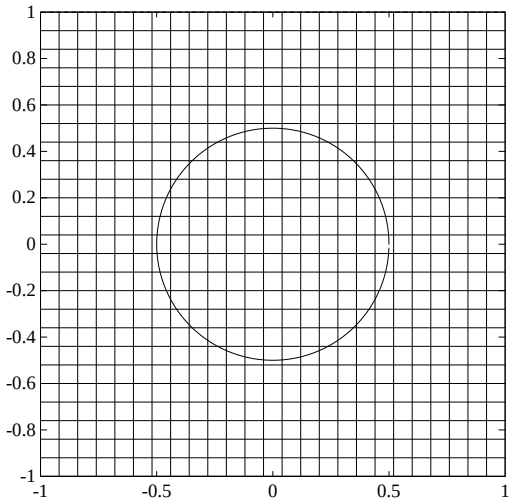


Figure 2.1: A circular interface Γ in a 26×26 uniform grid. This geometry is used for the test problems presented in Section 2.4.

Our goal is to develop a finite difference equation of the form

$$\sum_k \gamma_k \, u_{i+i_k, j+j_k} + \kappa_{ij} u_{ij} = f_{ij} + C_{ij} \tag{2.31}$$

for use at the point (x_i, y_j). The sum over k involves a finite number of points neighboring (x_i, y_j) (at most six in the formula we derive). So each i_k, j_k will take values in the set $\{-1, 0, 1\}$. The coefficients γ_k and indices i_k, j_k will depend on (i,j), so these should really be labeled γ_{ijk}, etc., but for simplicity of notation we will concentrate on a single point (i,j) and drop these indices.

We say (i,j) is a *regular point* if the interface does not come between any points in the standard 5-point stencil centered at (i,j). At these points we obtain an $O(h^2)$ truncation error using the standard 5-point $(k=5)$ formula

$$\frac{1}{h}\left(\left(\beta_{i+1/2,j}\frac{(u_{i+1,j} - u_{ij})}{h} - \beta_{i-1/2,j}\frac{(u_{i,j} - u_{i-1,j})}{h}\right)\right. \tag{2.32}$$

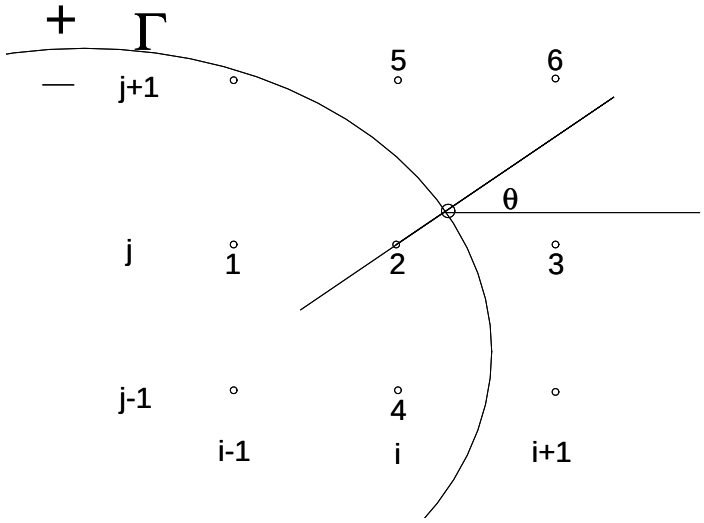


Figure 2.2: The geometry at an irregular grid point (i,j). The coefficients γ_1 through γ_6 will be determined for the stencil points labeled 1 through 6. The circled point on Γ is the point (x_i^*, y_j^*).

$$+ \left(\beta_{i,j+1/2}\frac{(u_{i,j+1}-u_{i,j})}{h} - \beta_{i,j-1/2}\frac{(u_{i,j}-u_{i,j-1})}{h}\right)\Bigg) + \kappa_{ij}u_{ij} = f_{ij},$$

with

$$C_{ij} = 0. \tag{2.33}$$

We wish to determine formulas of the form (2.31) for the irregular points also. Since these points are adjacent to the curve Γ, and form a lower dimensional set, it turns out to be sufficient to require an $O(h)$ truncation error at these points, just as in one dimension. We follow the same approach as in one dimension, and expand all $u_{i+i_k,j+j_k}$ about some point (x_i^*, y_j^*) on the interface Γ. In one dimension there was only one such point, α. In two dimensions we have flexibility in choosing (x_i^*, y_j^*). We might take, for example, the point closest to (x_i, y_j) as illustrated in Figure 2.2. We then expand each $u_{i+i_k,j+j_k}$ about (x_i^*, y_j^*), being careful to use the limiting values of derivatives of u from the correct side of the interface. We use the superscripts $-$ or $+$ to denote the limiting values of a function from one side or the other. As an example, in the configuration shown in Figure 2.2, we would expand

$$\begin{aligned} u(x_i,y_j) &= u^- + u_x^-(x_i-x_i^*) + u_y^-(y_j-y_j^*) + \frac{1}{2}u_{xx}^-(x_i-x_i^*)^2 \\ &\quad + \frac{1}{2}u_{yy}^-(y_j-y_j^*)^2 + u_{xy}^-(x_i-x_i^*)(y_j-y_j^*) + O(h^3) \end{aligned} \tag{2.34}$$

and

$$\begin{aligned} u(x_{i+1},y_j) &= u^+ + u_x^+(x_{i+1}-x_i^*) + u_y^+(y_j-y_j^*) + \frac{1}{2}u_{xx}^+(x_{i+1}-x_i^*)^2 \\ &\quad + \frac{1}{2}u_{yy}^+(y_j-y_j^*)^2 + u_{xy}^+(x_{i+1}-x_i^*)(y_j-y_j^*) + O(h^3). \end{aligned} \tag{2.35}$$

If we do this expansion at each point used in the difference equation (2.31) then the local truncation error T_{ij} can be expressed as a linear combination of the values $u^{\pm}$, $u^{\pm}_x$, $u^{\pm}_y$, $u^{\pm}_{xx}$, $u^{\pm}_{xy}$, $u^{\pm}_{yy}$. Following the one-dimensional derivation in Section 2.1, we now wish to eliminate all values on one side of the interface, say the values u^+, u^+_x, u^+_y, u^+_{xx}, u^+_{xy}, u^+_{yy}, in terms of the values on the other side, u^-, u^-_x, u^-_y, u^-_{xx}, u^-_{xy}, u^-_{yy}. We must do this using the jump conditions across Γ,

$$u^- = u^+ \tag{2.36}$$

and

$$\beta^- \frac{\partial u^-}{\partial n} = \beta^+ \frac{\partial u^+}{\partial n} \tag{2.37}$$

where $\partial/\partial n$ represents differentiation in the normal direction. From (2.36) we have that tangential derivatives are continuous, while (2.37) gives information about the jump in the normal direction. Differentiating these and manipulating the results allows us to perform the desired elimination, as detailed below. In order to do this, it turns out to be very convenient to first perform a local coordinate transformation into directions ξ, normal to Γ, and η, tangential to Γ.

Once T_{ij} is expressed as a linear combination of the values $u^-, u^-_x, u^-_y, u^-_{xx}, u^-_{xy}$, and u^-_{yy}, we must require that the coefficient of each of these terms vanishes in order to achieve an $O(h)$ truncation error. This gives a linear system of six equations to determine the coefficients γ_k. To obtain a solvable system we require six points in the stencil. We use the standard 5-point stencil together with one additional point.

To summarize, in order to determine the difference scheme at an irregular grid point we need to do the following:

- Select a point $(x_i^*, y_j^*) \in \Gamma$ near (x_i, y_j).
- Apply a local coordinate transformation in directions normal and tangential to Γ at (x_i^*, y_j^*).
- Derive the jump conditions relating $+$ and $-$ values at (x_i^*, y_j^*) in the local coordinates.
- Choose an additional point to form a six–point stencil.
- Set up and solve a linear system of six equations for the coefficients γ_k. The value C_{ij} is also obtained.

Below we give a detailed analysis of each step.

For each irregular grid point (x_i, y_j) we need to find a point (x_i^*, y_j^*) on the interface. We usually take this point as the projection of (x_i, y_j) on the interface if the interface is smooth at this point. Otherwise we can take any smooth point on the interface in the neighborhood of (x_i, y_j). In some contexts it may be more convenient to choose a nearby point that lies on a coordinate line between (x_i, y_j) and one of its neighbors.

After choosing (x_i^*, y_j^*) we are ready to apply a local coordinate transformation (shift + rotation) near this grid point. Let θ be the angle between the x-axis and the normal direction, pointing in the direction of the + side. The transformation is as follows:

$$\xi = (x - x_i^*)\cos\theta + (y - y_j^*)\sin\theta, \tag{2.38}$$

$$\eta = -(x - x_i^*)\sin\theta + (y - y_j^*)\cos\theta. \tag{2.39}$$

Notice that under this local coordinate transformation the PDE (2.30) remains unchanged. In fact, this is true more generally when β, κ, and f depend on x and y, as is shown in §2.3. We should have a new notation for $u(x,y), \kappa(x,y), f(x,y)$ in the local coordinates, say, $\bar{u}(\xi,\eta) = u(x,y)$, $\bar{\kappa}(\xi,\eta) = \kappa(x,y)$, and $\bar{f}(\xi,\eta) = f(x,y)$. But for simplicity we drop the bars and use the same notation in the local coordinates as in the old ones. With these local coordinates we are able to derive the interface conditions as we did in §2.1.

The interface relations in the local coordinates for 2D problems.
We consider a fixed point (x_i^*, y_j^*) and define a new ξ–η coordinate system based on the directions normal and tangential to Γ at this point using the formulas (2.38) and (2.39). In a neighborhood of this point, the interface lies roughly in the η–direction, so we can parameterize Γ locally by $\xi = \chi(\eta)$, $\eta = \eta$. Note that $\chi(0) = 0$ and, provided the boundary is smooth at (x_i^*, y_j^*), $\chi'(0) = 0$ as well.

The continuity condition (2.36) holds at each point on Γ. In our local coordinates, we can write this as

$$u^-(\chi(\eta), \eta) = u^+(\chi(\eta), \eta) \tag{2.40}$$

for all η in a neighborhood of $\eta = 0$. Differentiating this with respect to η gives

$$u_\xi^- \chi' + u_\eta^- = u_\xi^+ \chi' + u_\eta^+, \tag{2.41}$$

or, in compact form,

$$[u_\xi]\chi' + [u_\eta] = 0. \tag{2.42}$$

Differentiating again with respect to η gives

$$[u_{\xi\xi}]\chi'^2 + 2[u_{\xi\eta}]\chi' + [u_\xi]\chi'' + [u_{\eta\eta}] = 0. \tag{2.43}$$

Evaluating (2.42) and (2.43) at $\eta = 0$, where $\chi' = 0$, gives two of the desired jump conditions:

$$[u_\eta] = 0, \quad \text{i.e.} \quad u_\eta^- = u_\eta^+ = u_\eta, \tag{2.44}$$

$$[u_\xi]\,\chi'' + [u_{\eta\eta}] = 0. \tag{2.45}$$

We also have the jump condition (2.37) at each point on Γ. At a point $(\chi(\eta), \eta) \in \Gamma$ we can express the normal derivative in terms of ξ- and η- derivatives as

$$\frac{\partial u}{\partial n} = \frac{1}{\sqrt{1 + \chi'^2}}\,(u_\xi - u_\eta \chi')$$

so that we can write (2.37) as

$$\beta^-(u^-_\xi - u^-_\eta \chi') = \beta^+(u^+_\xi - u^+_\eta \chi'). \tag{2.46}$$

Differentiating this with respect to η gives

$$[\,\beta\,(u_{\xi\xi}\chi' + u_{\xi\eta} - u_\eta \chi'' - u_{\xi\eta}\chi'^2 - u_{\eta\eta}\,\chi')\,] = 0. \tag{2.47}$$

Evaluating (2.46) and (2.47) at $\eta = 0$ gives more jump conditions:

$$[\,\beta\, u_\xi\,] = 0, \tag{2.48}$$

$$[\,\beta\,(u_{\xi\eta} - u_\eta \chi'')\,] = 0. \tag{2.49}$$

We can use the relations (2.46)–(2.49) to derive the following expressions for values on the $(+)$ side of Γ in terms of values in the $(-)$ side. Setting

$$\rho = \beta^-/\beta^+,$$

we can write these relations as

$$\begin{aligned} u^+ &= u^- \\ u^+_\eta &= u^-_\eta \\ u^+_\xi &= \rho\, u^-_\xi \\ u^+_{\xi\,\eta} &= \rho u^-_{\xi\,\eta} + (1-\rho)\;\, u^-_\eta \chi'' \\ u^+_{\eta\,\eta} &= u^-_{\eta\,\eta} + (1-\rho)\;\, u^-_\xi\;\chi''. \end{aligned} \tag{2.50}$$

To obtain an expression for $u^+_{\xi\,\xi}$, we note that the PDE (2.30) gives

$$\beta^+ u^+_{\xi\,\xi} = \beta^- u^-_{\xi\,\xi} + \beta^- u^-_{\eta\,\eta} - \beta^+ u^+_{\eta\,\eta}$$

so that

$$u^+_{\xi\,\xi} = \rho u^-_{\xi\,\xi} + (\rho - 1)\, u^-_{\eta\,\eta} + (\rho - 1)\, u^-_\xi \chi''. \tag{2.51}$$

Now we have expressed all the quantities with $(+)$ superscripts in terms of the quantities with $(-)$ superscripts for the case $\chi'(0) = 0$. In this simple case they are homogeneous. The next thing to do is to choose an additional point from $(i-1, j-1)$, $(i-1, j+1)$, $(i+1, j-1)$, $(i+1, j+1)$ besides the standard five point stencil. It seems that the best choice is the point which has the shortest distance from (x^*, y^*). The additional point can be written as (x_{i+i_0}, y_{j+j_0}), where i_0 and j_0 are either -1 or 1 depending on the position of the additional point.

The derivation of the difference scheme for an irregular point. We are now ready to derive the difference schemes at irregular grid points. Denote the ξ-η coordinates of the six points in the difference stencil,

$$(x_{i-1}, y_j), (x_i, y_j), (x_{i+1}, y_j), (x_i, y_{j-1}), (x_i, y_{j+1}), (x_{i+i_0}, y_{j+j_0}),$$

as

$$(\xi_1, \eta_1), (\xi_2, \eta_2), (\xi_3, \eta_3), (\xi_4, \eta_4), (\xi_5, \eta_5), (\xi_6, \eta_6),$$

respectively. The local truncation error T_{ij} of the difference scheme (2.31) at (x_i, y_j) is then

$$\begin{aligned} T_{ij} = \gamma_1 u(\xi_1,\eta_1) + \gamma_2 u(\xi_2,\eta_2) + \gamma_3 u(\xi_3,\eta_3) + \gamma_4 u(\xi_4,\eta_4) \\ + \gamma_5 u(\xi_5,\eta_5) + \gamma_6 u(\xi_6,\eta_6) + \kappa_{ij} u(\xi_2,\eta_2) - f_{ij} - C_{ij}. \end{aligned} \tag{2.52}$$

We now expand all the terms about $(0,0)$ in the local coordinates from each side of the interface, as we did in (2.34) and (2.35), obtaining

$$u(\xi_k,\eta_k) \;=\; u^{\pm} + \xi_k\, u_\xi^{\pm} + \eta_k\, u_\eta^{\pm} + \frac{1}{2}\,\xi_k^2\, u_{\xi\xi}^{\pm} + \xi_k\,\eta_k\, u_{\xi\eta}^{\pm} + \frac{1}{2}\,\eta_k^{\pm}\, u_{\eta\eta}^{\pm} + O(h^3),$$

where the + or − sign is chosen depending on whether (ξ_k,η_k) lies on the + or − side of Γ.

We also use

$$\kappa_{ij}\, u(\xi_2,\eta_2) = \kappa^-\, u^- + O(h) \quad \text{and} \quad f_{ij} = f^- + O(h), \tag{2.53}$$

where $\kappa^- = \kappa(0,0)$ and so forth (recall that κ, u, and f are continuous). Using these expansions in (2.52) and collecting terms gives an expression of the form

$$\begin{aligned} T_{ij} \;=\; & a_1\, u^- + a_2\, u^+ + a_3\, u_\xi^- + a_4\, u_\xi^+ + a_5\, u_\eta^- + a_6\, u_\eta^+ \\ & + a_7\, u_{\xi\xi}^- + a_8\, u_{\xi\xi}^+ + a_9\, u_{\eta\eta}^- + a_{10}\, u_{\eta\eta}^+ + a_{11}\, u_{\xi\eta}^- \\ & + a_{12}\, u_{\xi\eta}^+ + \kappa^-\, u^- - f^- - C_{ij} + O(h). \end{aligned} \tag{2.54}$$

The coefficients a_j depend only on the position of the stencil relative to the interface. They are independent of the functions u, κ and f. If we define the index sets K^+ and K^- by

$$K^{\pm} = \{k :\ (\xi_k,\eta_k) \text{ is on the } \pm \text{ side of } \Gamma\},$$

then the a_j are given by

$$\begin{aligned} a_1 &= \sum_{k\in K^-} \gamma_k & a_2 &= \sum_{k\in K^+} \gamma_k \\ a_3 &= \sum_{k\in K^-} \xi_k\gamma_k & a_4 &= \sum_{k\in K^+} \xi_k\gamma_k \\ a_5 &= \sum_{k\in K^-} \eta_k\gamma_k & a_6 &= \sum_{k\in K^+} \eta_k\gamma_k \\ a_7 &= \frac{1}{2}\sum_{k\in K^-} \xi_k^2\gamma_k & a_8 &= \frac{1}{2}\sum_{k\in K^+} \xi_k^2\gamma_k \\ a_9 &= \frac{1}{2}\sum_{k\in K^-} \eta_k^2\gamma_k & a_{10} &= \frac{1}{2}\sum_{k\in K^+} \eta_k^2\gamma_k \\ a_{11} &= \sum_{k\in K^-} \xi_k\eta_k\gamma_k & a_{12} &= \sum_{k\in K^+} \xi_k\eta_k\gamma_k. \end{aligned} \tag{2.55}$$

Using the interface relations (2.50) and (2.51) in (2.54) and rearranging it we obtain

$$\begin{aligned} T_{ij} \;=\; & (a_1+a_2)\, u^- + \{a_3 + a_4\,\rho + a_8\,(\rho-1)\,\chi'' + a_{10}\,(1-\rho)\,\chi''\}\; u_\xi^- \\ & + \{a_5 + a_6 + a_{12}\,(1-\rho)\,\chi''\}\; u_\eta^- + \{a_7 + a_8\rho - \beta^-\}\; u_{\xi\,\xi}^- \\ & + \{a_9 + a_{10} + a_8\,(\rho-1) - \beta^-\}\; u_{\eta\,\eta}^- + \{a_{11} + a_{12}\rho\}\; u_{\xi\,\eta}^- \\ & + \{\beta^-\,(u_{\xi\,\xi}^- + u_{\eta\,\eta}^-) + k^-\, u^- - f^- + C_{ij}\} + O(h), \end{aligned} \tag{2.56}$$

where again $\rho = \beta^-/\beta^+$. From the PDE (2.30) we know that

$$\beta^- \left(u^-_{\xi\,\xi} + u^-_{\eta\,\eta}\right) + k^-\, u^- - f^- = 0.$$

and so this term drops out of (2.56) by taking $C_{ij} = 0$. We can ensure that $T_{ij} = O(h)$ by requiring that each coefficient of $u^-, u^-_\xi, u^-_\eta, u^-_{\xi\,\xi}, u^-_{\xi\,\eta}$, and $u^-_{\eta\,\eta}$ vanish. This gives six equations for the six unknowns $\gamma_1, \cdots, \gamma_6$:

$$\begin{aligned} a_1 + a_2 &= 0, \\ a_3 + a_4\,\rho + a_8\,(\rho - 1)\,\chi'' + a_{10}\,(1-\rho)\,\chi'' &= 0, \\ a_5 + a_6 + a_{12}\,(1-\rho)\,\chi'' &= 0, \\ a_7 + a_8\,\rho &= \beta^-, \\ a_9 + a_{10} + a_8\,(\rho - 1) &= \beta^-, \\ a_{11} + a_{12}\,\rho &= 0. \end{aligned} \tag{2.57}$$

As in one dimension, if $\beta^-\beta^+ > 0$, then the linear system has a unique solution. To prove this is not very complicated but rather tedious. We need to consider all the possible cases for the formation of the new stencil (*i.e.*, the position of the points relative to the interface). We omit the detailed analysis here. If $\beta^-\beta^+ < 0$, then it turns out that only for some specific value of $[\beta]$ the coefficient matrix for the unknown γ_js is singular, so the algorithm is typically successful even in this case. Moreover, by negating the equation on one side of the interface, it is possible to insure that $\beta^-\beta^+ > 0$ at the expense of perhaps introducing discontinuities into κ and f.

Note that since the interface relations (2.50) and (2.51) are homogenous, we have $C_{ij} = 0$ and there is no contribution to the right hand side resulting from the discontinuous coefficients. If $\beta^+ = \beta^-$ then solving (2.57) we recover the standard 5-point coefficients

$$\gamma_1 = \gamma_3 = \gamma_4 = \gamma_5 = \frac{\beta}{h^2},\ \gamma_2 = -\frac{4\,\beta}{h^2},\ \text{ and } \gamma_6 = 0.$$

In general, however, the resulting γ_js are different from those in the standard five point stencil. Figure 2.3 shows some representative stencils for a problem in which β has the value 1 on one side of Γ and 3 on the other side.

The exact nature of the coefficients depends on how large the jump in β is. We have not investigated these coefficients in general, but at least for reasonably mild discontinuities it seems that:

- The contributions to the difference schemes at irregular points are mainly from the standard five point stencil. These coefficients are $O(1/h^2)$ while the contributions from the 'additional points' are typically much smaller. The magnitude depends on the jump in β and the geometry of the grid.

- All the coefficients except occasionally γ_6 have the same sign ($-$ for the diagonal and $+$ for the off-diagonal) as in the classic five point difference formula. Since the contribution from the sixth point is much smaller than from the standard five points, we expect the classical theoretical analysis to still be applicable for the resulting linear system with slight modifications. In particular, the system is nearly diagonally dominant, and strictly so if γ_6 is always positive.

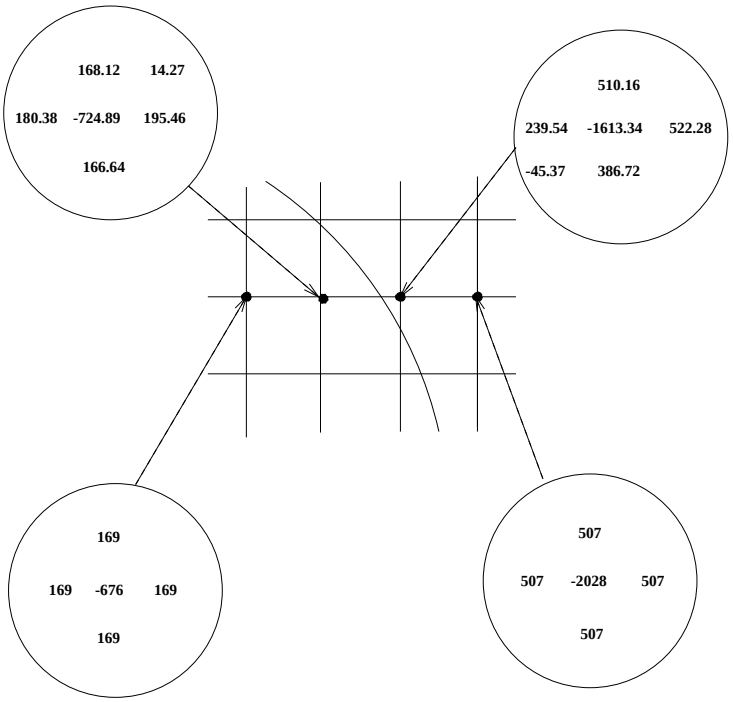


Figure 2.3: The γ_j coefficients at four grid points near the interface. The coefficient β is piecewise constant with the value $\beta = 1$ to the left and $\beta = 3$ to the right. The standard 5-point stencil is used at regular grid points while special 6-point stencils are used near the interface. The grid is a section of Figure 2.1, with $h = 1/13$.

We use an iterative method to solve the resulting linear system, which is block tridiagonal. In most of our numerical experiments we have used an LSOR iteration. If $\beta^-\beta^+ > 0$, the relaxation parameter is chosen as the optimal parameter for the Poisson problem on a square. The convergence speed is almost the same as that if we use the LSOR method to solve the Poisson problem with constant β on a square. This confirms the conclusions above. But if $\beta^-\beta^+ < 0$, it is difficult to determine a suitable relaxation parameter and we simply use the line Gauss-Seidel iteration. Since this case is less interesting physically, we have not investigated other approaches.

In the future, we plan to study the use of multigrid methods to achieve faster convergence. It is not clear how the multigrid convergence rate will be affected by the discontinuity in the coefficients. Multigrid methods for problems like (2.30) with discontinuous coefficients have been previously studied (*e.g.*, [1], [8]), but mainly for problems where the interfaces are aligned with the coordinate directions.

2.3 The general two-dimensional problem

In this section we present the analysis for the more complicated two-dimensional problem

$$(\beta\, u_x)_x + (\beta\, u_y)_y + \kappa(x,y)\, u = f(x,y) \qquad (x,y) \in \Omega, \tag{2.58}$$

Now β, κ, and f may all have discontinuities along a general interface Γ, and so do u, u_x, u_y, u_{xx}, u_{xy}, and u_{yy}. The process basically is the same as in the discussion of §2.2. We use the same notations and assumptions about the region Ω, uniform grid and arbitrary interface Γ. Again we want to use the difference scheme (2.31). For regular grid points, we still apply the standard 5-point stencil (2.32) and (2.33) giving a local truncation error of $O(h^2)$. We will concentrate on the derivation of the difference scheme at a typical irregular point (x_i, y_j).

We first demonstrate that the PDE (2.58) remains unchanged if the coordinate transformation is composed of a shift and rotation. In fact, taking an arbitrary function $w(x, y)$, under the transformations (2.38) and (2.39), we have:

$$\begin{aligned} w_x &= \bar{w}_\xi \cos\theta + \bar{w}_\eta \sin\theta, \\ w_y &= -\bar{w}_\xi \sin\theta + \bar{w}_\eta \cos\theta, \end{aligned}$$

where $\bar{w}(\xi, \eta) = w(x, y)$ and so forth, so we have

$$\begin{aligned} (\beta\, u_x)_x + (\beta\, u_y)_y + \kappa u &= \beta\,(u_{xx} + u_{yy}) + \beta_x\, u_x + \beta_y\, u_y + \kappa u \\ &= \bar{\beta}\,(\bar{u}_{\xi\,\xi} + \bar{u}_{\eta\,\eta}) + (\bar{\beta}_\xi \cos\theta - \bar{\beta}_\eta \sin\theta)\,(\bar{u}_\xi \cos\theta - \bar{u}_\eta \sin\theta) \\ &\quad + (\bar{\beta}_\xi \sin\theta + \bar{\beta}_\eta \cos\theta)\,(\bar{u}_\xi \sin\theta + \bar{u}_\eta \cos\theta) + \bar{\kappa}\bar{u} \\ &= \bar{\beta}\,(\bar{u}_{\xi\,\xi} + \bar{u}_{\eta\,\eta}) + \bar{\beta}_\xi\, \bar{u}_\xi + \bar{\beta}_\eta\, \bar{u}_\eta + \bar{\kappa}\bar{u} \\ &= (\bar{\beta}\, \bar{u}_\xi)_\xi + (\bar{\beta}\, \bar{u}_\eta)_\eta + \bar{\kappa}\bar{u}. \end{aligned}$$

For simplicity, we will drop the bars again. If some grid point $u(x_i, y_j)$ happens to fall on the interface, then $u(x_i, y_j)$ is defined as the limiting value of $u(x, y)$ from one side of the interface or the other. The same argument applies to all other functions such as β, κ, f and the derivatives of $u(x, y)$. The corresponding u_{ij} is the approximation to this specific limit. We again use the superscripts $-$ and $+$ to express the limiting values from one side of the interface or the other.

The essential difference now is that the interface relations are more complicated. Two interface conditions are needed in advance to make the problem well-posed. We assume locally that they are defined by

$$u^+ - u^- = w(\eta), \tag{2.59}$$

$$\beta^+ \frac{\partial u}{\partial n}^+ - \beta^- \frac{\partial u}{\partial n}^- = v(\eta), \tag{2.60}$$

where again $\xi = \chi(\eta)$, $\eta = \eta$ is the parametric representation of the interface in the neighborhood of the point (x_i^*, y_j^*). Here $v(\eta)$ and $w(\eta)$ are arbitrary (smooth) functions that are used to impose quite general jump conditions across Γ. (Often $v = w = 0$, but we may wish to impose other jumps as an external constraint. An example occurs in the incompressible Navier-Stokes equations with the immersed boundary method, where the known jump in pressure across the interface must be imposed in the solution of a Poisson problem; see Chapter 4.)

Differentiating (2.59) with respect to η along the interface we get

$$[u_\xi]\, \chi' + [u_\eta] = w'(\eta). \tag{2.61}$$

Differentiating this again with respect to η we obtain

$$[u_\xi]\, \chi'' + \chi' \frac{d}{d\eta}[u_\xi] + [u_{\xi\eta}]\, \chi' + [u_{\eta\eta}] = w''(\eta). \tag{2.62}$$

Notice that in the local coordinates, (2.60) can be written as

$$\beta^+(u_\xi^+ - u_\eta^+\chi') = \beta^-(u_\xi^- - u_\eta^-\chi') + v\sqrt{1+(\chi')^2}. \tag{2.63}$$

Differentiating this with respect to η along the interface we have

$$\begin{aligned}(\beta_\xi{}^+\chi' + \beta_\eta{}^+)\,(u_\xi^+ - u_\eta^+\chi') + \beta^+\left(u_{\xi\,\xi}^+\chi' + u_{\xi\,\eta}^+ - \frac{d}{d\eta}(u_\eta^+)\,\chi' - u_\eta^+\chi''\right)\\ = (\beta_\xi{}^-\chi' + \beta_\eta{}^-)\,(u_\xi^- - u_\eta^-\chi')\\ + \beta^-\left(u_{\xi\,\xi}^-x' + u_{\xi\,\eta}^- - \frac{d}{d\eta}(u_\eta^-)\,\chi' - u_\eta^-\chi''\right)\\ + v'(\eta)\left(\sqrt{1+(\chi')^2} + \frac{v(\eta)\chi'\chi''}{\sqrt{1+(\chi')^2}}\right).\end{aligned} \tag{2.64}$$

Also from the PDE we know that

$$\begin{aligned}u_{\xi\,\xi}^+ &= \frac{\beta^-}{\beta^+}\,u_{\xi\,\xi}^- + \frac{\beta^-}{\beta^+}\,u_{\eta\,\eta}^- - u_{\eta\,\eta}^+ + \frac{\beta_\xi{}^-}{\beta^+}\,u_\xi^- - \frac{\beta_\xi{}^+}{\beta^+}\,u_\xi^+\\ &\quad + \frac{\beta_\eta{}^-}{\beta^+}\,u_\eta^- - \frac{\beta_\eta{}^+}{\beta^+}\,u_\eta^+ + \frac{[f]}{\beta^+} + \frac{\kappa^-\,u^- - \kappa^+\,u^+}{\beta^+}.\end{aligned} \tag{2.65}$$

The numerator of the last term can be rewritten as

$$\kappa^-\,u^- - \kappa^+\,u^+ \;=\; -[\kappa]\,u^- - [u]\,\kappa^+. \tag{2.66}$$

Using these relations, we can express quantities with $(+)$ superscripts in terms of those with $(-)$ superscripts. The detailed analysis is similar to the process in Section 3 although it is more complicated due to the fact that $\beta(x,y)$ is not constant in the neighborhood of the interface and the presence of the source-like terms $w(\eta)$, $v(\eta)$. To save space here we omit the detailed analysis and simply present the results. Recall that the parameterization $\xi = \chi(\eta)$ is assumed to be smooth with $\chi'(0) = 0$ and that we are considering the jumps across Γ at a fixed point (x_i^*, y_j^*) corresponding to $\xi = \eta = 0$. In the expressions below, all functions are evaluated at this point. The jump relations are given by:

$$\begin{aligned}u^+ &= u^- + w,\\ u_\xi^+ &= \rho\,u_\xi^- + \frac{v}{\beta^+},\\ u_\eta^+ &= u_\eta^- + w',\\ u_{\xi\,\xi}^+ &= \left(\frac{\beta_\xi{}^-}{\beta^+} - \chi''\right)u_\xi^- + \left(\chi'' - \frac{\beta_\xi{}^+}{\beta^+}\right)u_\xi^+ + \frac{\beta_\eta{}^-}{\beta^+}\,u_\eta^- - \frac{\beta_\eta{}^+}{\beta^+}\,u_\eta^+\\ &\quad + (\rho-1)\,u_{\eta\,\eta}^- + \rho u_{\xi\,\xi}^- - w'' + \frac{[f]}{\beta^+} - \frac{[\kappa]\,u^- + \kappa^+\,[u]}{\beta^+},\\ u_{\eta\,\eta}^+ &= u_{\eta\,\eta}^- + (u_\xi^- - u_\xi^+)\,\chi'' + w'',\\ u_{\xi\,\eta}^+ &= \frac{\beta_\eta{}^-}{\beta^+}\,u_\xi^- - \frac{\beta_\eta{}^+}{\beta^+}u_\xi^+ + \left(u_\eta^+ - \rho u_\eta^-\right)\chi'' + \rho\,u_{\xi\,\eta}^- + \frac{v'}{\beta^+}.\end{aligned} \tag{2.67}$$

The local truncation error T_{ij} at (x_i, y_j) is again given by (2.54) with the coefficients a_k given by the expressions (2.55) in terms of the unknowns γ_k. We now replace all of the $(+)$ values by expressions involving $(-)$ values using (2.67). After combining common terms and eliminating some terms due to the relation

$$\beta^- (u^-_{\xi\,\xi} + u^-_{\eta\,\eta}) + \beta^-_\xi u^-_\xi + \beta^-_\eta u^-_\eta + k^- u^- - f^- = 0$$

(resulting from the PDE (2.58)), we obtain

$$\begin{aligned} T_{ij} \;=\; & (a_1 - \frac{a_8\,[\kappa]}{\beta^+} + a_2)\, u^- + \left\{ a_3 + a_8 \left(\frac{\beta_\xi{}^-}{\beta^+} - \chi'' \right) + a_{10}\chi'' + a_{12}\frac{\beta_\eta{}^-}{\beta^+} \right. \\ & \left. + \rho \left(a_4 + a_8 \left(\chi'' - \frac{\beta_\xi{}^+}{\beta^+} \right) - a_{10}\chi'' - a_{12}\frac{\beta_\eta{}^+}{\beta^+} \right) - \beta_\xi{}^- \right\} u^-_\xi \\ & + \left\{ a_5 + a_6 + a_8 \left(\frac{\beta_\eta{}^-}{\beta^+} - \frac{\beta_\eta{}^+}{\beta^+} \right) + a_{12}(1-\rho)\,\chi'' - \beta_\eta{}^- \right\} u^-_\eta \\ & + \{a_7 + a_8\rho - \beta^-\}\; u^-_{\xi\,\xi} + \{a_9 + a_{10} + a_8\,(\rho - 1) - \beta^-\}\; u^-_{\eta\,\eta} \\ & + \{a_{11} + a_{12}\rho\}\; u^-_{\xi\,\eta} + (\hat{T}_{ij} - C_{ij}) + O(h), \end{aligned}$$

where

$$\begin{aligned} \hat{T}_{ij} \;=\; & a_2\, w + a_{12}\,\frac{v'}{\beta^+} + \left(a_6 - \frac{a_8\,\beta_\xi{}^+}{\beta^+} + a_{12}\chi'' \right) w' + a_{10}\, w'' \\ & + \frac{1}{\beta^+} \left(a_4 + a_8(\chi'' - \frac{\beta_\xi{}^+}{\beta^+}) - a_{10}\chi'' - a_{12}\frac{\beta_\eta{}^+}{\beta^+} \right) v \\ & + a_8 \left\{ \frac{[f]}{\beta^+} - \frac{\kappa^+\, w}{\beta^+} - w'' \right\}. \end{aligned} \tag{2.68}$$

We can ensure that $T_{ij} = O(h)$ by requiring that each coefficient of $u^-, u^-_\xi,\, u^-_\eta,\, u^-_{\xi\,\xi}$, $u^-_{\xi\,\eta}$, and $u^-_{\eta\,\eta}$ vanish, as well as the term $(\hat{T}_{ij} - C_{ij})$. This gives seven equations for the unknowns $\gamma_1, \cdots, \gamma_6$ and C_{ij}. The first six equations give a linear system for the γ's (recall that each a_j is a linear combination of the γ's, given by (2.55), and that $\rho = \beta^-/\beta^+$):

$$\begin{aligned} a_1 + a_2 - a_8\,[\kappa]/\beta^+ &= 0 \\ a_3 + \rho a_4 + a_8(\beta_\xi{}^- - \rho\beta_\xi{}^+ - [\beta]\chi'')/\beta^+ & \\ + a_{10}[\beta]\chi''/\beta^+ + a_{12}(\beta_\eta{}^- - \rho\beta_\eta{}^+)/\beta^+ &= \beta_\xi{}^- \\ a_5 + a_6 - a_8[\beta_\eta]/\beta^+ + a_{12}(1-\rho)\,\chi'' &= \beta_\eta{}^- \\ a_7 + a_8\,\rho &= \beta^- \\ a_9 + a_{10} + a_8\,(\rho - 1) &= \beta^- \\ a_{11} + a_{12}\,\rho &= 0. \end{aligned} \tag{2.69}$$

Once the γ_j's are computed, we can easily obtain C_{ij} as

$$C_{ij} = \hat{T}_{ij}, \tag{2.70}$$

where $\hat{T}_{ij}$ is given by (2.68).

The remarks at the end of the §2.2 still hold. Moreover, in the case where β and κ are continuous but vary with x and y, we see that the set of equations (2.69) reduces to

$$\begin{aligned} a_1 + a_2 &= 0, \\ a_3 + a_4 &= \beta_\xi, \\ a_5 + a_6 &= \beta_\eta, \\ a_7 + a_8 &= \beta, \\ a_9 + a_{10} &= \beta, \\ a_{11} + a_{12} &= 0. \end{aligned}$$

This set of equations is satisfied to $O(h)$ by using the five-point stencil with

$$\gamma_1 = \beta_{i-1/2,j}/h^2, \qquad \gamma_2 = -(\beta_{i-1/2,j} + \beta_{i+1/2,j} + \beta_{i,j-1/2} + \beta_{i,j+1/2})/h^2,$$

$$\gamma_3 = \beta_{i+1/2,j}/h^2, \qquad \gamma_4 = \beta_{i,j-1/2}/h^2, \qquad \gamma_5 = \beta_{i,j+1/2}/h^2, \qquad \gamma_6 = 0.$$

These are the coefficients for the standard formula (2.32). So for elliptic equations with singular sources we still can use the standard central difference scheme and only need to add the correction terms at irregular grid points. Furthermore if $\beta(x, y)$ is a constant, we have the following theorem

Theorem 2.2 *If $\beta(x, y)$ is a constant and κ is continuous, then the solution of equations (2.69) are*

$$\gamma_1 = \gamma_3 = \gamma_4 = \gamma_5 = \frac{\beta}{h^2}, \quad \gamma_2 = -\frac{4\,\beta}{h^2}, \quad \gamma_6 = 0. \tag{2.71}$$

Proof: We only need to verify that these γ_is satisfy the system of equations (2.69). Without loss of generality let the irregular grid point (x_i, y_j) be the origin. The continuity condition in β and κ means $[\kappa] = 0$, $[\beta] = 0$, $\rho = 1$, $\beta_\xi^- = \beta_\xi^+$, and $\beta_\eta^- = \beta_\eta^+$. Therefore the first equation in (2.69) now becomes

$$a_1 + a_2 = 0, \qquad \textit{i.e.} \qquad \sum_{k=1}^{k=6} \gamma_k = 0,$$

which is obviously true. Under the transformations (2.38) and (2.39), the new coordinates (ξ_{i_k}, η_{j_k}), $k = 1, \ldots, 5$, corresponding to $(-h, 0)$, $(0, 0)$, $(h, 0)$, $(0, -h)$, and $(0, h)$ are

$$\begin{aligned} (\xi_{i_1}, \eta_{j_1}) &= (-(h + x^*)\cos\alpha - y^*\sin\alpha,\ (h + x^*)\sin\alpha - y^*\cos\alpha) \\ (\xi_{i_2}, \eta_{j_2}) &= (-x^*\cos\alpha - y^*\sin\alpha,\ x^*\sin\alpha - y^*\cos\alpha) \\ (\xi_{i_3}, \eta_{j_3}) &= ((h - x^*)\cos\alpha - y^*\sin\alpha,\ (-h + x^*)\sin\alpha, -y^*\cos\alpha) \\ (\xi_{i_4}, \eta_{j_4}) &= (-x^*\cos\alpha - (h + y^*)\sin\alpha,\ x^*\sin\alpha - (h + y^*)\cos\alpha)) \\ (\xi_{i_5}, \eta_{j_5}) &= (-x^*\cos\alpha + (h - y^*)\sin\alpha,\ x^*\sin\alpha + (h - y^*)\cos\alpha)\,. \end{aligned}$$

The second equation of the system (2.69) is

$$a_3 + a_4 = \sum_{k=1}^{k=6} \gamma_k \, \xi_{i_k} = \frac{\beta}{h^2} \, ((-h - x^*) + (h - x^*) - x^* - x^* + 4x^*) \cos\theta$$
$$+ \frac{\beta}{h^2} \, (-y^* - y^* - (h + y^*) + (h - y^*) + 4y^*) \sin\theta = 0.$$

For the fourth equation we have

$$a_7 + a_8 = \sum_{k=1}^{k=6} \gamma_k \, \frac{{\xi_{i_k}}^2}{2} =$$
$$\beta \, \Big(\, (h + x^*)^2 \cos^2\theta + {y^*}^2 \sin^2\theta + 2(h + x^*) y^* \cos\theta \sin\theta$$
$$+ (h - x^*)^2 \cos^2\theta + {y^*}^2 \sin^2\theta - 2(h - x^*) y^* \cos\theta \sin\theta +$$
$$ {x^*}^2 \cos^2 + (h + y^*)^2 \sin^2\theta + 2x^*(h + y^*) \cos\theta \sin\theta +$$
$$ {x^*}^2 \cos^2\theta + (h - y^*)^2 \sin^2\theta - 2x^*(h - y^*) \cos\theta \sin\theta -$$
$$ -4{x^*}^2 \cos^2\theta - 4{y^*}^2 \sin^2\theta - 8x^* y^* \cos\theta \sin\theta \, \Big) \, / (2\, h^2) = \beta.$$

By the same token we get

$$a_5 + a_6 = \sum_{k=1}^{k=6} \gamma_k \, \eta_{i_k} = 0,$$
$$a_9 + a_{10} = \sum_{k=1}^{k=6} \gamma_k \, \frac{{\eta_{i_k}}^2}{2} = \beta.$$

The last equation is verified below:

$$a_{11} + a_{12} = \sum_{k=1}^{k=6} \gamma_k \, \xi_{i_k} \eta_{i_k} =$$
$$-\beta \, (\, [\, (h + x^*) \cos\theta + y^* \sin\theta] \, [\, (h + x^*) \sin\theta - y^* \sin\theta]$$
$$- [\, (h - x^*) \cos\theta - y^* \sin\theta \,] \, [\, (h - x^*) \sin\theta + y^* \cos\theta]$$
$$- [\, x^* \cos\theta + (h + y^*) \sin\theta] \, [\, x^* \sin\theta - (h + y^*) \cos\theta]$$
$$- [\, x^* \cos\theta - (h - y^*) \sin\theta] \, [\, x^* \sin\theta + (h - y^*) \cos\theta]$$
$$+ 4 \, [\, x^* \cos\theta + y^* \sin\theta] \, [\, x^* \sin\theta - y^* \cos\theta] \,) \, / h^2 = 0. \qquad \square$$

2.4 Numerical Results

We have done many numerical tests which confirm the expected order of accuracy for the immersed interface approach. We will present a few examples in two dimensions. In all of these examples Γ is the circle $x^2 + y^2 = 1/4$ within the computaional domain $-1 \le x, \, y \le 1$. See Figure 2.1.

(a) (b)

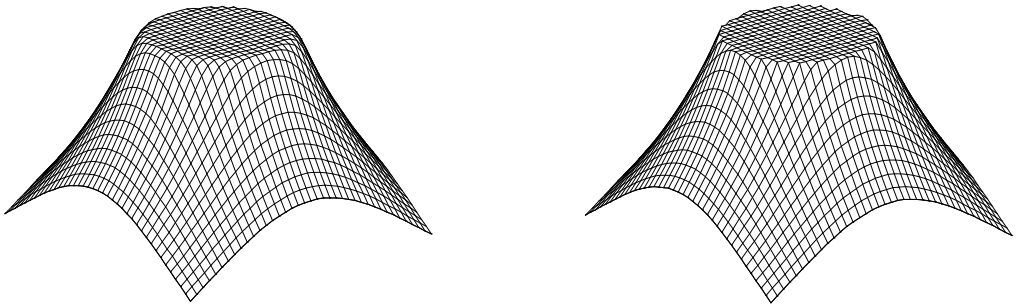


Figure 2.4: Comparison of two methods on Example 2.1. (a) The discrete delta function approach. (b) The immersed interface method.

Example 2.1 *In this example we compare our method with the discrete delta function approach for a problem where there is a singular source term along* Γ. *The differential equation is:*

$$u_{xx} + u_{yy} = \int_{\Gamma} 2\,\delta(x - X(s))\,\delta(y - Y(s))\,ds. \tag{2.72}$$

We use the Dirichlet boundary condition which is determined from the exact solution

$$u(x,y) = \begin{cases} 1 & \text{if } r \le \frac{1}{2} \\ 1 + \log(2r) & \text{if } r > \frac{1}{2}, \end{cases} \tag{2.73}$$

where $r = \sqrt{x^2 + y^2}$. *From the equation we know that* $[\partial u/\partial n] = 2$ *at all points on* Γ.

For the discrete delta function method we take m points on the interface Γ, where $m = n = 2/\Delta x = 2/\Delta y$ is the number of uniform grid points in each direction. In the numerical experiments we have found that beyond this point, increasing the number of points on the interface gives little improvement in the solution. We use Peskin's discrete delta function (1.8). We have also tested the hat delta function defined in (1.7) and the numerical results are almost the same.

Figure 2.4 shows the results of both methods. We see that our method accurately gives the jump in the normal direction while the discrete delta function approach smears the jump, resulting in first order accuracy.

Table 2.1 shows the results of a grid refinement study. The maximum error over all grid points,

$$\| E_n \|_\infty = \max_{i,j} \mid u(x_i, y_j) - u_{ij} \mid,$$

is presented, where u_{ij} is the computed approximation at the uniform grid points (x_i, y_j). For our method we also display $\| T_n \|_\infty$, the infinity norm of the local truncation error

Table 2.1: Numerical results for Example 2.1

	Discrete delta function		Immersed interface method			
n	$\| E_n \|_\infty$	ratio	$\| E_n \|_\infty$	ratio	$\| T_n \|_\infty$	ratio
20	3.6140×10^{-1}		2.3908×10^{-3}		2.8276×10^{-1}	
40	2.6467×10^{-2}	12.7939	8.3461×10^{-4}	2.8646	1.6922×10^{-1}	1.6710
80	1.3204×10^{-2}	2.0045	2.4451×10^{-4}	3.4134	8.3449×10^{-2}	2.0278
160	6.6847×10^{-3}	1.9753	6.6856×10^{-5}	3.6573	4.1892×10^{-2}	1.9920
320	3.3393×10^{-3}	2.0018	1.5672×10^{-5}	4.2658	2.3049×10^{-2}	1.8175

over all grid points. The local truncation errors are $O(h^2)$ except at those points which are close to the interface where they are $O(h)$. We also display the ratios of successive errors,

$$\text{ratio} = \| E_{2n} \|_\infty / \| E_n \|_\infty, \quad \text{or} \quad \| T_{2n} \|_\infty / \| T_n \|_\infty.$$

A ratio of 2 corresponds to first order accuracy, while a ratio of 4 indicates second order accuracy. We will use the same notation for other examples in this section.

Example 2.2 *We now consider a problem with discontinuous coefficients as well as a singular source term. The equation is*

$$(\beta u_x)_x + (\beta u_y)_y = f(x,y) + C \int_\Gamma \delta(\vec{x} - \vec{X}(s))\, ds \tag{2.74}$$

with

$$f(x,y) = 8\,(x^2+y^2) + 4,$$

$$\beta(x,y) = \begin{cases} x^2+y^2+1 & \text{if } x^2+y^2 \le \frac{1}{4}. \\ b & \text{if } x^2+y^2 > \frac{1}{4} \end{cases}$$

Dirichlet boundary conditions are determined from the exact solution

$$u(x,y) = \begin{cases} r^2 & \text{if } r \le \frac{1}{2} \\ \left(1 - \frac{1}{8\,b} - \frac{1}{b}\right)/4 + \left(\frac{r^4}{2} + r^2\right) b + C\,\log(2r)/b & \text{if } r > \frac{1}{2}. \end{cases} \tag{2.75}$$

It is easy to check that (2.75) satisfies (2.74). Table 2.2 gives numerical results for the case $b = 10$, $C = 0.1$. Again the local truncation error near Γ is only $O(h)$, but the resulting global error is seen to be $O(h^2)$. Figure 2.5 shows the computed solution for the case $b = 10$, $C = 0.1$ and $b = -3$, $C = 0.1$, respectively. In the first case $\beta^-\beta^+ > 0$. As we mentioned in Section 3 the resulting linear system is "almost" symmetric positive definite. We use the LSOR method with the optimal relaxation parameter for the Poisson equation on the square. In the second case $\beta^-\beta^+ < 0$. The computed solution has the same accuracy as in the first case. In this case we used the Gauss-Seidel iteration.

(a) (b)

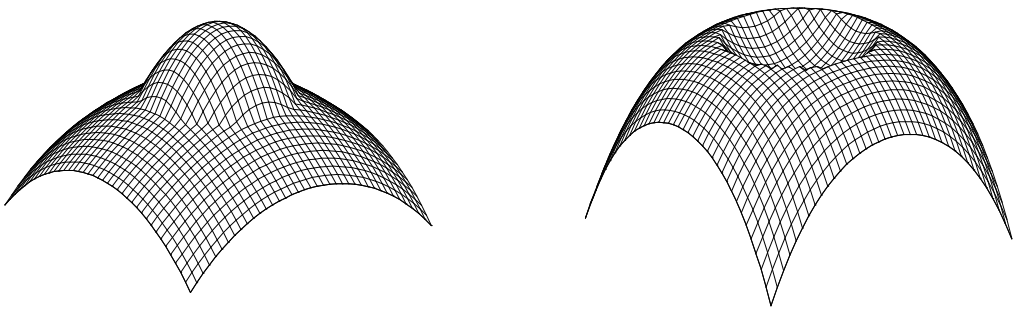


Figure 2.5: The solutions for Example 2.2. (a) The function u for the case $b = 10$, $C = 0.1$. (b) The function $-u$ in the case $b = -3$, $C = 0.1$.

Table 2.2: Numerical results for Example 2.2 with $b = 10$, $C = 0.1$

n	$\| E_n \|_\infty$	ratio	$\| T_n \|_\infty$	ratio
20	3.5195×10^{-3}		6.3843×10^{-1}	
40	7.5613×10^{-4}	4.6547	3.5988×10^{-1}	1.7740
80	1.6512×10^{-4}	4.5792	1.8999×10^{-1}	1.8942
160	3.6002×10^{-5}	4.5864	9.7499×10^{-2}	1.9487
320	8.4405×10^{-6}	4.2655	4.9374×10^{-2}	1.9747

Example 2.3 *In this example we impose a jump in the function u itself and also a jump in the normal derivative of u as external constraints. The differential equation on each side of the interface is simply the Laplace equation*

$$u_{xx} + u_{yy} = 0.$$

The jumps in u and $\partial u/\partial n$ are chosen so that the following function is the exact solution:

$$u(x,y) = \begin{cases} e^x \cos y & \text{if } r \le \frac{1}{2} \\ 0 & \text{if } r > \frac{1}{2}. \end{cases} \tag{2.76}$$

From this we can compute the functions v and w in (2.59) and (2.60). Since $\beta \equiv 1$, the standard five-point stencil is used at each grid point and equation (2.70) is used to determine the right hand side C_{ij}. Any fast Poisson solver can then be used to solve the resulting system, with Dirichlet boundary conditions $u = 0$ on $\partial\Omega$.

Figure 2.6a shows the computed results on a 40×40 grid. The discontinuity in u is captured sharply. Table 2.3 shows that we again obtain second order accuracy at all grid points, even in the neighborhood of the discontinuity.

(a) (b)

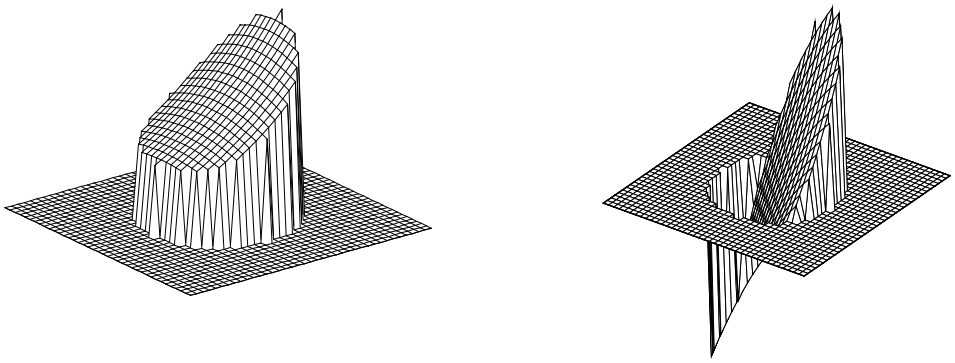


Figure 2.6: The solutions for Example 2.3, with jumps in u and its normal derivative specified along Γ. (a) Solution (2.76). (b) Solution (2.77).

Example 2.4 *As a final test, we repeated this experiment with the exact solution*

$$u(x,y) = \begin{cases} x^2 - y^2 & \text{if } r \le \frac{1}{2} \\ 0 & \text{if } r > \frac{1}{2} \end{cases} \tag{2.77}$$

shown in Figure 2.6b. In this case our method produced a computed solution with errors in the range $10^{-13} - 10^{-15}$ at all grid points (in double precision). This is expected since for the special case of a quadratic function the resulting truncation error should be identically zero, and only rounding errors appear in the computed solution (as amplified by the condition number of the matrix).

Table 2.3: Numerical results for Example 2.3 with true solution (2.76).

n	$\| E_n \|_\infty$	ratio	$\| T_n \|_\infty$	ratio
20	4.37883×10^{-4}		2.99215×10^{-2}	
40	1.07887×10^{-4}	4.0587	1.52546×10^{-2}	1.9615
80	2.77752×10^{-5}	3.8843	7.70114×10^{-3}	1.9808
160	7.49907×10^{-6}	3.7038	3.87481×10^{-3}	1.9875
320	1.74001×10^{-6}	4.3098	1.93917×10^{-3}	1.9982

In summary, we have developed second order accurate difference methods for elliptic equations in the following situations: (i) The differential equations have discontinuous coefficients along a general interface. (ii) The differential equations have singular sources along a general interface. (iii) The differential equations have externally imposed constraints on the jump in u or normal derivatives of u across an interface. In all cases we are able to derive an appropriate difference stencil involving at most six grid points and the correct right hand side so that the global error is $O(h^2)$ at all points on a uniform grid.

In the special case where the coefficients are continuous, the difference stencil reduces to the standard 5-point stencil (2.32) and only the correct right hand side must be derived to obtain second order accuracy. In particular, if the coefficients are constant then the standard 5-point Laplacian is used and a fast Poisson solver can be used to solve the resulting linear system.

The ideas presented here can be used on a wide variety of other problems with discontinuous coefficients or singular sources. All that is required is that we be able to predict jumps in the solution and its first derivatives across Γ from the equation. These jumps are used in conjunction with appropriate Taylor series expansions about the interface to derive the difference scheme and right hand side.

2.5 Some implementation details and a Fortran package for two-dimensional problems.

It seems to be difficult to implement the immersed interface methods even if we have the analytic expressions for the interfaces and jumps conditions because

- The interface may be arbitrary and complicated.
- We need to identify the irregular grid points and decide which side of the interface they are.
- We need first and second derivative information for the interface.
- We need to differentiate the jumps u and $[\beta u]$ along the interface.

We have written several Fortran subroutines to perform these complicated jobs. Although not optimized, they have been used successfully to deal with many interface problems including the Stokes equations with a moving interface (see Chapter 4).

The idea is to use cubic spline interpolation[1] with the arc-length of the interface as the parameter. All the quantities defined on the interface, such as the x- and y- coordinates, jumps in u and βu, etc., can be expressed by splines with the same parameter. The first and second derivatives of these quantities can be obtained by differentiating the splines exactly.

By using the splines, it is also relatively easy to identify those irregular grid points near the interface. Taking a grid point (x_i, y_j), how do we determine whether a grid point is regular and on which side of the interface it is located? First we find all the intersections between the interface and two straight lines $x = x_i$ and $y = y_j$. This only requires solving some cubic equations. Let these intersections be $(x_i, y^{(1)})$, $(x_i, y^{(2)})$, $\cdots$, $(x_i, y^{(s)})$, and $(x^{(1)}, y_j)$, $(x^{(2)}, y_j)$, $\cdots$, $(x^{(p)}, y_j)$, where $0 \le s, p \le N_1$, N_1 is the number of control points taken on the interface. Then we can find the point (x^*, y^*) which has the shortest distance from (x_i, y_j) among the intersections. If the distance is less than or equal to the space size h, then (x_i, y_j) is an irregular grid point. By the sign of the inner product of $(x_i - x^*, y_j - y^*)^T \cdot \vec{n}$, we can tell on which side of the interface this grid point lies. The point (x^*, y^*) is also saved along with other information and used later for the local coordinate transformation needed to derive the modified difference scheme at the irregular grid point (x_i, y_j).

The approach described above has been used successfully for a number of interface problems with complicated interfaces. Based on this approach we have written a Fortran package DIIM. DIIM is a double precision package for solving the elliptic interface problems on rectangular regions with Dirichlet boundary conditions. The prologue of this package can be found in the Appendix A.

Also, with the spline approach we can solve Poisson problems or elliptic equations on complicated regions with an embedding technique. We circumscribe the region with a rectangle and only modify the difference scheme at those irregular grid points inside the region. At the grid points outside of the region, we will use dummy values, say zero, for the solution. So it really does not matter what the difference scheme is outside. With the optimal relaxation parameter for the Poisson problem defined on the rectangle, both the SOR and LSOR methods require fewer iterations for the problems defined on the small region than that defined on the whole rectangle. So this approach is very competitive compared with other methods which embed the region into a larger rectangle and need additional treatment such as solving integral equations, or a few Poisson problems on the rectangle etc., see [10], [21], [35], [45].

Below we present an example of solving a Poisson problem with the Dirichlet boundary condition on a complicated region.

Example 2.5 *The equation is*

$$u_{xx} + u_{yy} = -2 \sin x \, \sin y, \qquad \textit{in the region} \quad r \le \frac{1}{2} + \frac{\sin(6\theta)}{4},$$

where (r, θ) is the polar coordinates of (x, y). The Dirichlet boundary condition is chosen from the following solution

$$u(x, y) = \frac{1}{4} + \sin x \sin y.$$

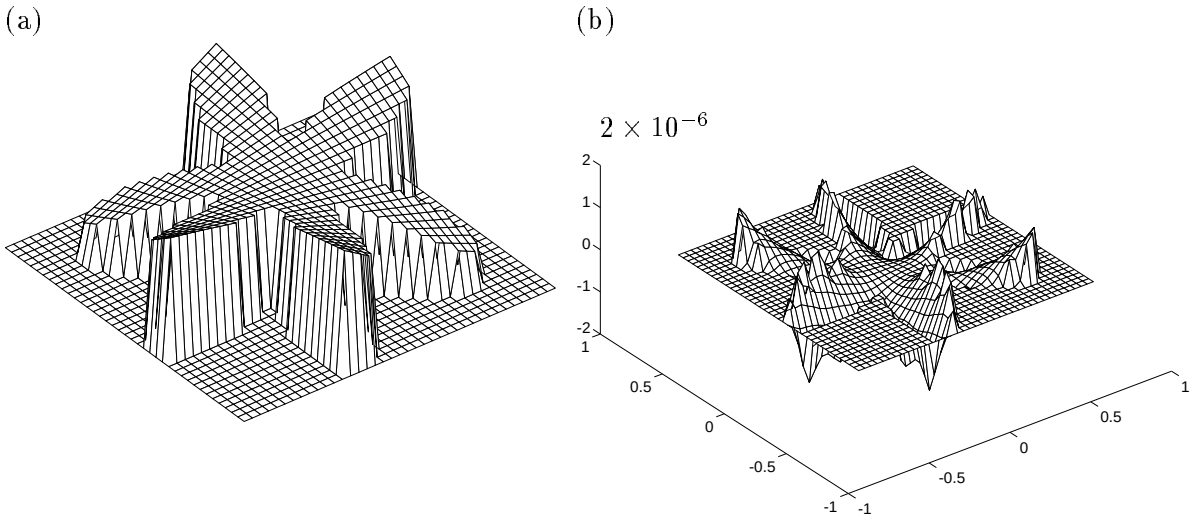


Figure 2.7: Using the immersed interface method to solve the Poisson problem 2.78 with the Dirichlet BC on the irregular rigion. (a) The computed solution with a 40×40 grid; (b) The error plot of the computed solution.

Fig 2.7 (a) plots the numerical solution using a 40×40 grid. The values on the outside of the region are set to zero. Fig 2.7 (b) plots the error in the computed solution. We see that the error is on the order of 10^{-6} with the 40×40 grid. The prologue of the Fortran package PPACK for solving the Poisson problems on irregular regions can be found in the Appendix B.

However we need to mention that this approach does not use jump conditions to determine the difference schemes at the grid points near the interface. Also note that although the resulting linear system obtained with this approach is still diagonally dominant and irreducible, it is not symmetric anymore because of the complicated region. So usually the SOR iterative method would be preferred to the LSOR approach. Without the symmetry, it is difficult to determine the optimal parameters for the *Alternating Direction Implicit* (ADI) iterative method, or to apply the *Fast Fourier Transform* (FFT). Whether multigrid techniques can yield fast convergence for the resulting system is not known at this point.

2.6 General three-dimensional problems

Now we consider general three-dimensional problems

$$(\beta\, u_x)_x + (\beta\, u_y)_y + (\beta\, u_z)_z + \kappa(x,y,z)\, u = f(x,y,z), \quad (x,y,z) \in \Omega \tag{2.78}$$

in some region Ω, where all the coefficients β, κ, f may be discontinuous, and f may even be singular across an interface, which is now a surface S: $x = x(\mu,\nu)$, $y = y(\mu,\nu)$, $z = z(\mu,\nu)$, To make the problem well-posed, we need two interface conditions of the form

$$[u] = w, \tag{2.79}$$

[1] Linear or other interpolation techniques could be used, perhaps at the expense of losing the second order accuracy.

$$[\beta \frac{\partial u}{\partial n}] = q, \tag{2.80}$$

across the interface surface.

2.6.1 Interface relations.

At a point (x^*, y^*, z^*) on the interface, we need to use local coordinates to simplify the derivation of the interface relations. The local coordinates (ξ, η, ζ) are chosen so that ξ is parallel to the normal direction of the interface pointing outward. The η- and ζ- axes are in the tangent plane passing through (x^*, y^*, z^*). In the neighborhood of this point, the interface can be expressed as

$$\xi = \chi(\eta, \zeta), \qquad \text{with} \qquad \chi(0,0) = 0, \;\; \chi_\eta(0,0) = 0, \;\; \chi_\zeta(0,0) = 0. \tag{2.81}$$

Notice that in the local coordinates the equation (2.78) is unchanged, so we will use the same notation for u, w, q, β, κ and f.

As we did before, we use the jump conditions and their derivatives as well as the differential equation itself to get the interface relations between the quantities of two sides of the interface surface. Let us first differentiate (2.79) with respect to η and ζ respectively to get

$$[u_\xi]\chi_\eta + [u_\eta] = w_\eta, \tag{2.82}$$
$$[u_\xi]\chi_\zeta + [u_\zeta] = w_\zeta. \tag{2.83}$$

Differentiating (2.82) with respect to ζ yields

$$\chi_\eta \frac{\partial}{\partial \zeta}[u_\xi] + \chi_{\eta\zeta}\,[u_\xi] + [u_{\eta\xi}]\,\chi_\zeta + [u_{\eta\zeta}] = w_{\eta\zeta}. \tag{2.84}$$

Differentiating (2.82) with respect to η and differentiating (2.83) with respect to ζ respectively we obtain

$$\chi_\eta \frac{\partial}{\partial \eta}[u_\xi] + \chi_{\eta\eta}\,[u_\xi] + \chi_\eta\,[u_{\eta\xi}] + [u_{\eta\eta}] = w_{\eta\eta}, \tag{2.85}$$
$$\chi_\zeta \frac{\partial}{\partial \zeta}[u_\xi] + \chi_{\zeta\zeta}\,[u_\xi] + \chi_\zeta\,[u_{\zeta\xi}] + [u_{\zeta\zeta}] = w_{\zeta\zeta}. \tag{2.86}$$

Before differentiating the jump of the normal derivative (2.80) we first express the unit normal vector of the interface S as

$$\vec{n} = \frac{(1,\; -\chi_\eta,\; -\chi_\zeta\;)}{\sqrt{1 + {\chi_\eta}^2 + {\chi_\zeta}^2}}. \tag{2.87}$$

So the interface condition (2.80) can be written as

$$[\,\beta\;(\;u_\xi - u_\eta\,\chi_\eta - u_\zeta\,\chi_\zeta\;)\,] = q(\eta,\zeta)\,\sqrt{1 + {\chi_\eta}^2 + {\chi_\zeta}^2}. \tag{2.88}$$

Differentiating this with respect to η gives

$$\begin{aligned}
&[\,(\,\beta_\xi\,\chi_\eta+\beta_\eta\,)\,(\,u_\xi-u_\eta\,\chi_\eta-u_\zeta\,\chi_\zeta\,)\,]\\
&+\left[\beta\left(\,u_{\xi\xi}\,\chi_\eta+u_{\xi\eta}-\chi_\eta\,\frac{\partial}{\partial\eta}u_\eta-\chi_\zeta\,\frac{\partial}{\partial\eta}u_\zeta-u_\eta\,\chi_{\eta\eta}-u_\zeta\,\chi_{\eta\zeta}\right)\right]\\
&=q_\eta\,\sqrt{1+{\chi_\eta}^2+{\chi_\zeta}^2}+q\,\frac{\chi_\eta\,\chi_{\eta\eta}}{\sqrt{1+{\chi_\eta}^2+{\chi_\zeta}^2}}.
\end{aligned}\tag{2.89}$$

Similarly, differentiating (2.88) with respect to ζ gives

$$\begin{aligned}
&[\,(\,\beta_\xi\,\chi_\zeta+\beta_\zeta\,)\,\,(\,u_\xi-u_\eta\,\chi_\eta-u_\zeta\,\chi_\zeta\,)\,]\\
&+\left[\beta\left(u_{\xi\xi}\,\chi_\zeta+u_{\xi\zeta}-\chi_\eta\,\frac{\partial}{\partial\zeta}u_\eta-\chi_\zeta\,\frac{\partial}{\partial\zeta}u_\zeta-u_\eta\,\chi_{\eta\zeta}-u_\zeta\,\chi_{\zeta\zeta}\right)\right]\\
&=q_\zeta\,\sqrt{1+{\chi_\eta}^2+{\chi_\zeta}^2}+q\,\frac{\chi_\zeta\,\chi_{\zeta\zeta}}{\sqrt{1+{\chi_\eta}^2+{\chi_\zeta}^2}}.
\end{aligned}\tag{2.90}$$

At the origin, $\chi_\eta(0,0)=\chi_\zeta(0,0)=0$, and from (2.82)–(2.90) we can conclude that

$$\begin{aligned}
u^+ &= u^-+w,\\
u^+_\xi &= \frac{\beta^-}{\beta^+}\,u^-_\xi+\frac{q}{\beta^+},\\
u^+_\eta &= u^-_\eta+w_\eta,\\
u^+_\zeta &= u^-_\zeta+w_\zeta,\\
u^+_{\eta\zeta} &= u^-_{\eta\zeta}+u^-_\xi\,\chi_{\eta\zeta}-u^+_\xi\chi_{\eta\zeta}+w_{\eta\zeta},\\
u^+_{\eta\eta} &= u^-_{\eta\eta}+(u^-_\xi-u^+_\xi)\chi_{\eta\eta}+w_{\eta\eta},\\
u^+_{\zeta\zeta} &= u^-_{\zeta\zeta}+(u^-_\xi-u^+_\xi)\chi_{\zeta\zeta}+w_{\zeta\zeta},\\
u^+_{\xi\eta} &= \frac{\beta^-}{\beta^+}\,u^-_{\xi\eta}+\left(u^+_\eta-\frac{\beta^-}{\beta^+}\,u^-_\eta\right)\chi_{\eta\eta}+\left(u^+_\zeta-\frac{\beta^-}{\beta^+}\,u^-_\zeta\right)\chi_{\eta\zeta}\\
&\quad+\frac{\beta^-_\eta}{\beta^+}\,u^-_\xi-\frac{\beta^+_\eta}{\beta^+}\,u^+_\xi+\frac{q_\eta}{\beta^+},\\
u^+_{\xi\zeta} &= \frac{\beta^-}{\beta^+}\,u^-_{\xi\zeta}+\left(u^+_\eta-\frac{\beta^-}{\beta^+}\,u^-_\eta\right)\chi_{\eta\zeta}+\left(u^+_\zeta-\frac{\beta^-}{\beta^+}\,u^-_\zeta\right)\chi_{\zeta\zeta}\\
&\quad+\frac{\beta^-_\zeta}{\beta^+}u^-_\xi-\frac{\beta^+_\zeta}{\beta^+}u^+_\xi+\frac{q_\zeta}{\beta^+}.
\end{aligned}\tag{2.91}$$

To get the relation for $u^+_{\xi\,\xi}$ we need to use the differential equation (2.78) itself from which we can write

$$[\,\beta\,(u_{\xi\xi}+u_{\eta\eta}+u_{\zeta\zeta})+\beta_\xi u_\xi+\beta_\eta u_\eta+\beta_\zeta u_\zeta+\kappa u\,]=[f]\,.\tag{2.92}$$

Notice that

$$\kappa^-u^--\kappa^+u^+=\kappa^-u^--\kappa^+u^-+\kappa^+u^--\kappa^+u^+=-[\kappa]u^--\kappa^+[u].\tag{2.93}$$

Rearranging equation (2.92) and using (2.93) above we get

$$\begin{aligned}\beta^+\left(u^+_{\xi\,\xi}+u^+_{\eta\,\eta}+u^+_{\zeta\zeta}\right)+\beta^+_\xi\,u^+_\xi+\beta^+_\eta u^+_\eta+\beta^+_\zeta u^+_\zeta=\\ \beta^-\left(u^-_{\xi\,\xi}+u^-_{\eta\,\eta}+u^-_{\zeta\zeta}\right)+\beta^-_\xi\,u^-_\xi\\ +\beta^-_\eta u^-_\eta+\beta^-_\zeta u^-_\zeta+[f]+\kappa^- u^- -\kappa^+ u^+.\end{aligned}\tag{2.94}$$

Plugging the sixth and seventh equations of (2.91) in (2.94) and collecting terms finally we have

$$\begin{aligned}u^+_{\xi\,\xi}=\frac{\beta^-}{\beta^+}\,u^-_{\xi\,\xi}+\left(\frac{\beta^-}{\beta^+}-1\right)\,u^-_{\eta\,\eta}+\left(\frac{\beta^-}{\beta^+}-1\right)u^-_{\zeta\zeta}+\\ u^+_\xi\left(\chi_{\eta\eta}+\chi_{\zeta\zeta}-\frac{\beta^+_\xi}{\beta^+}\right)-u^-_\xi\left(\chi_{\eta\eta}+\chi_{\zeta\zeta}-\frac{\beta^-_\xi}{\beta^+}\right)\\ +\frac{1}{\beta^+}\left(\beta^-_\eta u^-_\eta-\beta^+_\eta u^+_\eta\right)+\frac{1}{\beta^+}\left(\beta^-_\zeta u^-_\zeta-\beta^+_\zeta u^+_\zeta\right)\\ -\frac{1}{\beta^+}\left([\kappa]u^-+\kappa^+[u]\right)+\frac{[f]}{\beta^+}-w_{\eta\eta}-w_{\zeta\zeta}.\end{aligned}\tag{2.95}$$

2.6.2 Difference scheme

At a regular grid point, we still use the classic central difference scheme which has a seven-point stencil. So we will concentrate below on developing difference formulas for the irregular grid points. Taking a typical irregular grid point, say (x_i, y_j, z_k), we try to develop the modified difference scheme at this point. Again we only require the local truncation error for this difference scheme to be $O(h)$. Let us write the difference scheme as follows:

$$\sum_m \gamma_m\, u_{i+i_m,j+j_m,k+k_m}+\kappa_{ijk}\,u_{ijk}=f_{ijk}+C_{ijk},\tag{2.96}$$

where i_m, j_m, k_m may be $0, \pm1, \pm2, \cdots$. Of course we want the number of grid points involved to be as few as possible. So first we need to determine the stencil, and then find the coefficients γ_m for the given stencil.

The analysis is similar to the two dimensional case. We take a point (x^*, y^*, z^*) on the interface surface near (x_i, y_j, z_k) and use local coordinates (ξ, η, ζ) at (x^*, y^*, z^*). For the elliptic equation the coefficients γ_m should be of order $O(1/h^2)$. So if we expand $u_{i+i_m,j+j_m,k+k_m}$ in the difference scheme about the origin of the local coordinates from each side of the surface S, we need to match up to second derivatives to guarantee that the local truncation error is $O(h)$. Using the ten interface relations (2.91) and (2.95) to eliminate quantities at the $(+)$ side of the interface, then the Taylor expansion of (2.96) will contain u^-, u^-_ξ, u^-_η, u^-_ζ, $u^-_{\xi\,\xi}$, $u^-_{\eta\,\eta}$, $u^-_{\zeta\zeta}$, $u^-_{\xi\,\eta}$, $u^-_{\xi\zeta}$, and $u^-_{\eta\zeta}$. To match them we need altogether ten grid points to get ten equations for the γ_ms. Thus we need to find three additional points besides the standard seven-point stencil. The three additional grid points can be taken from any of the twenty grid points $(i\pm1, j\pm1, k\pm1)$, $(i\pm1, j\pm1, k)$, $(i\pm1, j, k\pm1)$, $(i, j\pm1, k\pm1)$.

Once we have determined the stencil we need to find the coefficients of the difference scheme. To get the equations for those coefficients we use the Taylor expansion of (2.96)

about (x^*, y^*, z^*), the origin of the local coordinates. If the grid point (x_i, y_j, z_k) is on the $(-)$ side, we will get

$$\begin{aligned} T_{ijk} &= a_1\, u^- + a_2\, u^+ + a_3\, u^-_\xi + a_4\, u^+_\xi + a_5\, u^-_\eta + a_6\, u^+_\eta + a_7\, u^-_\zeta + a_8\, u^+_\zeta \\ &\quad + a_9\, u^-_{\xi\,\xi} + a_{10}\, u^+_{\xi\,\xi} + a_{11}\, u^-_{\eta\,\eta} + a_{12}\, u^+_{\eta\,\eta} + a_{13}\, u^-_{\zeta\zeta} \\ &\quad + a_{14}\, u^+_{\zeta\zeta} + a_{15}\, u^-_{\xi\,\eta} + a_{16}\, u^+_{\xi\,\eta} + a_{17}\, u^-_{\xi\zeta} + a_{18}\, u^+_{\xi\zeta} \\ &\quad + a_{19}\, u^-_{\eta\zeta} + a_{20}\, u^+_{\eta\zeta} + \kappa^-\, u^- \\ &= f^- + C_{ijk} + O(h). \end{aligned} \tag{2.97}$$

Here the a_i's have similar meanings as in (2.55) except now there are more of them and they are more complicated. Using the interface relations (2.91) and (2.95), and rearranging (2.97) we have

$$\begin{aligned} T_{ijk} &= \left(a_1 - a_{10}\frac{[\kappa]}{\beta^+}\right) u^- + a_2\, u^+ + \left\{a_3 - a_{10}\left(\chi_{\eta\eta} + \chi_{\zeta\zeta} - \frac{\beta^-_\xi}{\beta^+}\right)\right. \\ &\quad \left. + a_{12}\,\chi_{\eta\eta} + a_{14}\,\chi_{\zeta\zeta} + a_{16}\,\frac{\beta^-_\eta}{\beta^+} + a_{18}\frac{\beta^-_\zeta}{\beta^+} + a_{20}\,\chi_{\eta\zeta}\right\} u^-_\xi \\ &\quad + \left\{a_4 + a_{10}\left(\chi_{\eta\eta} + \chi_{\zeta\zeta} - \frac{\beta^+_\xi}{\beta^+}\right)\right. \\ &\quad \left. - a_{12}\,\chi_{\eta\eta} - a_{14}\,\chi_{\zeta\zeta} - a_{16}\,\frac{\beta^+_\eta}{\beta^+} - a_{18}\,\frac{\beta^+_\zeta}{\beta^+} - a_{20}\,\chi_{\eta\zeta}\right\} u^+_\xi \\ &\quad + \left(a_5 + a_{10}\,\frac{\beta^-_\eta}{\beta^+} - a_{16}\,\frac{\beta^-}{\beta^+}\,\chi_{\eta\eta} - a_{18}\,\frac{\beta^-}{\beta^+}\,\chi_{\eta\zeta}\right)\, u^-_\eta \\ &\quad + \left(a_6 - a_{10}\,\frac{\beta^+_\eta}{\beta^+} + a_{16}\,\chi_{\eta\eta} + a_{18}\,\chi_{\eta\zeta}\right)\, u^+_\eta \\ &\quad + \left(a_7 + a_{10}\,\frac{\beta^-_\zeta}{\beta^+} - a_{16}\,\frac{\beta^-}{\beta^+}\,\chi_{\eta\zeta} - a_{18}\,\frac{\beta^-}{\beta^+}\,\chi_{\zeta\zeta}\right)\, u^-_\zeta \\ &\quad + \left(a_8 - a_{10}\,\frac{\beta^+_\zeta}{\beta^+} + a_{16}\,\chi_{\eta\zeta} + a_{18}\,\chi_{\zeta\zeta}\right)\, u^+_\zeta \\ &\quad + \left(a_9 + a_{10}\,\frac{\beta^-}{\beta^+}\right) u^-_{\xi\,\xi} + \left(a_{11} + a_{12} + a_{10}\,(\frac{\beta^-}{\beta^+} - 1)\right) u^-_{\eta\,\eta} \\ &\quad + \left(a_{13} + a_{14} + a_{10}\,\left(\frac{\beta^-}{\beta^+} - 1\right)\right) u^-_{\zeta\zeta} + \left(a_{15} + a_{16}\,\frac{\beta^-}{\beta^+}\right) u^-_{\xi\,\eta} \\ &\quad + \left(a_{17} + a_{18}\,\frac{\beta^-}{\beta^+}\right) u^-_{\xi\zeta} + (a_{19} + a_{20})\; u^-_{\eta\zeta} + a_{12}\, w_{\eta\eta} + a_{14}\, w_{\zeta\zeta} \\ &\quad + a_{10}\left(\frac{[f]}{\beta^+} - \frac{\kappa^+\,[u]}{\beta^+} - w_{\eta\eta} - w_{\zeta\zeta}\right) + a_{16}\,\frac{q_\eta}{\beta^+} \\ &\quad + a_{18}\,\frac{q_\zeta}{\beta^+} + a_{20}\, w_{\eta\zeta} + \kappa^-\, u^- - f^- - C_{ijk}. \end{aligned} \tag{2.98}$$

Now it is clear that to make T_{ijk} to be $O(h)$ we should set

$$a_1 - a_{10}\frac{[\kappa]}{\beta^+} + a_2 = 0, \tag{2.99}$$

$$\begin{aligned} a_3 - a_{10}\left(\chi_{\eta\eta} + \chi_{\zeta\zeta} - \frac{\beta_\xi^-}{\beta^+}\right) + a_{12}\,\chi_{\eta\eta} + a_{14}\,\chi_{\zeta\zeta} + a_{16}\frac{\beta_\eta^-}{\beta^+} \\ + a_{18}\frac{\beta_\zeta^-}{\beta^+} + a_{20}\,\chi_{\eta\zeta} + \frac{\beta^-}{\beta^+}\left\{a_4 + a_{10}\left(\chi_{\eta\eta} + \chi_{\zeta\zeta} - \frac{\beta_\xi^+}{\beta^+}\right)\right. \\ \left. - a_{12}\,\chi_{\eta\eta} - a_{14}\,\chi_{\zeta\zeta} - a_{16}\frac{\beta_\eta^+}{\beta^+} - a_{18}\frac{\beta_\zeta^+}{\beta^+} - a_{20}\,\chi_{\eta\zeta}\right\} &= \beta_\xi^-, \end{aligned} \tag{2.100}$$

$$\begin{aligned} a_5 + a_{10}\frac{\beta_\eta^-}{\beta^+} - a_{16}\frac{\beta^-}{\beta^+}\,\chi_{\eta\eta} - a_{18}\frac{\beta^-}{\beta^+}\,\chi_{\eta\zeta} \\ + a_6 - a_{10}\frac{\beta_\eta^+}{\beta^+} + a_{16}\,\chi_{\eta\eta} + a_{18}\,\chi_{\eta\zeta} &= \beta_\eta^-, \end{aligned} \tag{2.101}$$

$$\begin{aligned} a_7 + a_{10}\frac{\beta_\zeta^-}{\beta^+} - a_{16}\frac{\beta^-}{\beta^+}\,\chi_{\eta\zeta} - a_{18}\frac{\beta^-}{\beta^+}\,\chi_{\zeta\zeta} \\ + a_8 - a_{10}\frac{\beta_\zeta^+}{\beta^+} + a_{16}\,\chi_{\eta\zeta} + a_{18}\,\chi_{\zeta\zeta} &= \beta_\zeta^-, \end{aligned} \tag{2.102}$$

$$a_9 + a_{10}\frac{\beta^-}{\beta^+} = \beta^-, \tag{2.103}$$

$$a_{11} + a_{12} + a_{10}\left(\frac{\beta^-}{\beta^+} - 1\right) = \beta^-, \tag{2.104}$$

$$a_{13} + a_{14} + a_{10}\left(\frac{\beta^-}{\beta^+} - 1\right) = \beta^-, \tag{2.105}$$

$$a_{15} + a_{16}\frac{\beta^-}{\beta^+} = 0, \tag{2.106}$$

$$a_{17} + a_{18}\frac{\beta^-}{\beta^+} = 0, \tag{2.107}$$

$$a_{19} + a_{20} = 0. \tag{2.108}$$

This is a system of ten equations with ten variables. We can solve this system to get the coefficients γ_m of the difference scheme at this particular irregular grid point. Once we know the γ_m, we know the a_i as well, so we can calculate the correction term from the following:

$$\begin{aligned} C_{ijk} &= a_{10}\left(\frac{[f]}{\beta^+} - \frac{\kappa^+\,[u]}{\beta^+} - w_{\eta\eta} - w_{\zeta\zeta}\right) + a_{12}\,w_{\eta\eta} + a_{14}\,w_{\zeta\zeta} \\ &\quad + a_{16}\frac{q_\eta}{\beta^+} + a_{18}\frac{q_\zeta}{\beta^+} + a_{20}\,w_{\eta\zeta} + a_2\,[u] \\ &\quad + \frac{1}{\beta^+}\left\{a_4 + a_{10}\left(\chi_{\eta\eta} + \chi_{\zeta\zeta} - \frac{\beta_\xi^+}{\beta^+}\right) - a_{12}\,\chi_{\eta\eta}\right. \end{aligned}$$

$$
\begin{aligned}
&- a_{14}\,\chi_{\zeta\zeta} - a_{16}\,\frac{\beta_\eta^+}{\beta^+} - a_{18}\frac{\beta_\zeta^+}{\beta^+} - a_{20}\,\chi_{\eta\zeta}\Bigg\}\, q \\
&+ \left(a_6 - a_{10}\,\frac{\beta_\eta^+}{\beta^+} + a_{16}\,\chi_{\eta\eta} + a_{18}\,\chi_{\eta\zeta}\right)\, w_\eta \\
&+ \left(a_8 - a_{10}\,\frac{\beta_\zeta^+}{\beta^+} + a_{16}\,\chi_{\eta\zeta} + a_{18}\,\chi_{\zeta\zeta}\right)\, w_\zeta.
\end{aligned} \tag{2.109}
$$

If the grid point is on $(+)$ side, there are two ways to deal with it. The first one is to modify the correction term C_{ijk} and the linear system (2.99)–(2.108) a little bit. Use the following relation

$$
\begin{aligned}
\kappa^+\,u^+ &= \kappa^-\,u^- + \kappa^+\,u^+ - \kappa^-\,u^- \\
&= \kappa^-\,u^- + \kappa^+\,[u] + [\kappa]\,u^-,
\end{aligned} \tag{2.110}
$$

and let the difference scheme at this irregular (x_i, y_j, z_k) be:

$$
\sum_m \hat{\gamma}_m\, u_{i+i_m,j+j_m,k+k_m} + \kappa_{ijk}\, u_{ijk} = f_{ijk} + \hat{C}_{ijk}. \tag{2.111}
$$

Then $\hat{\gamma}_m$ still satisfy equations (2.100)–(2.108). Now the first equation becomes

$$
a_1 - a_{10}\,\frac{[\kappa]}{\beta^+} + a_2 = -[\kappa], \tag{2.112}
$$

and the correction term $\hat{C}_{ijk}$ is

$$
\hat{C_{ijk}} = C_{ijk} + \kappa^+\,[u] - [f]. \tag{2.113}
$$

The other way is simply to reverse the roles of the two sides $(+)$ and $(-)$ in the discussion above.

We have tested a couple of examples. Although we can not take very fine grids to test the second order convergence due to the size in three-dimensions, we do observe good numerical results. Below we give one test example.

Example 2.6 *We consider a problem in three dimensions with discontinuous coefficients as well as the singular sources. The equation is defined on the cube:* $-1 \le x, y, z, \le 1$ *and has the form*

$$
(\beta u_x)_x + (\beta u_y)_y + (\beta u_z)_z + \kappa\, u = f,
$$

where

$$
\beta(x,y,z) = \begin{cases} 1 + x^2 + y^2 + z^2 & \text{if } x^2+y^2+z^2 \le \frac{1}{4} \\ 1 & \text{if } x^2+y^2+z^2 > \frac{1}{4}, \end{cases}
$$

$$
f(x,y,z) = \begin{cases} 6 + 11\,(x^2+y^2+z^2) \\ \dfrac{1}{x^2+y^2+z^2} - \dfrac{1}{\sqrt{x^2+y^2+z^2}} - \log\left(2\sqrt{x^2+y^2+z^2}\right). \end{cases}
$$

Dirichlet boundary condition is determined from the exact solution

$$u(x,y,z) = \begin{cases} x^2+y^2+z^2 & \text{if } x^2+y^2+z^2 \le \frac{1}{4} \\ \dfrac{1}{\sqrt{x^2+y^2+z^2}} + \log\left(2\sqrt{x^2+y^2+z^2}\right) & \text{if } x^2+y^2+z^2 > \frac{1}{4}. \end{cases}$$

Table 2.4 lists the local truncation and global errors in the infinity norm.

Table 2.4: Numerical results for three dimensional Example 2.6.

n	$\| E_n \|_\infty$	ratio	$\| T_n \|_\infty$	ratio
20	9.2824×10^{-3}		1.1675	
40	2.8176×10^{-3}	3.2945	0.6587	1.7724
80	7.1043×10^{-4}	3.9656	0.3757	1.7528

Chapter 3

IMMERSED INTERFACE METHOD FOR HEAT EQUATIONS WITH FIXED INTERFACE(S)

In this chapter we study the immersed interface method for heat equations with fixed interface(s) in one and two dimensions.

3.1 General 1D heat equations with fixed interface(s).

Consider the model problem

$$\begin{aligned} u_t(x,t) &= (\beta(x,t)\,u_x)_x + \kappa(x,t)\,u(x,t) - f(x,t) + C(t)\delta(x-\alpha) \\ &\quad + \frac{1}{2}\,\left(\beta(\alpha^-,t) + \beta(\alpha^+,t)\right)\hat{C}(t)\,\delta'(x-\alpha), \\ &\quad 0 \le x \le 1, \qquad 0 < \alpha < 1, \qquad t \ge 0, \end{aligned} \tag{3.1}$$

with specified boundary and initial conditions. We assume $\beta(x,t)$, $\kappa(x,t)$ and $f(x,t)$ are bounded but may all have discontinuities at the interface α. From the equation we can conclude

$$\begin{aligned} [u] &= u(\alpha^+,t) - u(\alpha^-,t) = \hat{C}(t), \\ [\beta\,u_x] &= \beta(\alpha^+,t)\,u_x(\alpha^+,t) - \beta(\alpha^-,t)\,u_x(\alpha^-,t) = C(t). \end{aligned} \tag{3.2}$$

We use a uniform Cartesian grid as in § 2.1 and use the efficient Crank-Nicolson scheme at regular grid points which is unconditionally stable. The general difference scheme at time t^n is the following

$$\begin{aligned} \frac{u_i^{n+1} - u_i^n}{k} &= \frac{1}{2}\,\Big[\gamma_{i,1}^n u_{i-1}^n + \gamma_{i,2}^n u_i^n + \gamma_{i,3}^n u_{i+1}^n + \kappa_i^n\,u_i^n - f_i^n + C_i^n + \\ &\qquad \gamma_{i,1}^{n+1} u_{i-1}^{n+1} + \gamma_{i,2}^{n+1} u_i^{n+1} + \gamma_{i,3}^{n+1} u_{i+1}^{n+1} + \kappa_i^{n+1}\,u_i^{n+1} - f_i^{n+1} + C_i^{n+1}\Big], \end{aligned}$$

where k is the time step and the ratio k/h is a constant, $\kappa_i^n = \kappa(x_i,t^n)$ and so on. At regular grid points for which $\alpha \notin (x_{i-1}, x_{i+1})$, we have the standard weights

$$\begin{aligned} &\gamma_{i,1}^l = \beta_{i-\frac{1}{2}}^l/h^2, \quad \gamma_{i,2}^l = -(\beta_{i-\frac{1}{2}}^l + \beta_{i+\frac{1}{2}}^l)/h^2, \\ &\gamma_{i,3}^l = \beta_{i+\frac{1}{2}}^l/h^2 \quad \text{and} \quad C_i^l = 0, \quad l = n \quad \text{or} \quad n+1, \end{aligned} \tag{3.3}$$

where $\beta_{i-\frac{1}{2}}^l = \beta\left(x_{i-1/2}, t^l\right)$ and so on. Since the interface in fixed, the derivation for the difference scheme is just slightly different from that in § 2.1. So we will omit the details and just give the results directly.

Suppose $x_j \le \alpha < x_{j+1}$, then x_j and x_{j+1} are two irregular grid points. In this case the coefficients $\gamma_{j,1}^l$, $\gamma_{j,2}^l$ and $\gamma_{j,3}^l$ satisfy the following system of equations:

$$\gamma_{j,1}^l + \gamma_{j,2}^l + \left(1 - \frac{(x_{j+1}-\alpha)^2}{2\,(\beta^+)^l}\,[\kappa]^l\right)\,\gamma_{j,3}^l = 0,$$

$$(x_{j-1}-\alpha)\,\gamma_{j,1}^l + (x_j-\alpha)\,\gamma_{j,2}^l + \left\{\frac{(\beta^-)^l}{(\beta^+)^l}\,(x_{j+1}-\alpha) + \left(\frac{(\beta_x^-)^l}{(\beta^+)^l} - \frac{(\beta^-)^l(\beta_x^+)^l}{\{(\beta^+)^l\}^2}\right)\,\frac{(x_{j+1}-\alpha)^2}{2}\right\}\gamma_{j,3}^l = (\beta_x^-)^l,$$

$$\frac{(x_{j-1}-\alpha)^2}{2}\gamma_{j,1}^l + \frac{(x_j-\alpha)^2}{2}\gamma_{j,2}^l + \frac{(x_{j+1}-\alpha)^2\,(\beta^-)^l}{2\,(\beta^+)^l}\,\gamma_{j,3}^l = (\beta^-)^l,$$

where $l = n,\ n+1$, and

$$[\kappa]^l = \left[\kappa(\alpha, x^l)\right], \qquad (\beta^-)^l = \beta(\alpha^-, t^l), \qquad (\beta^+)^l = \beta(\alpha^+, t^l),$$
$$(\beta_x^-)^l = \beta_x(\alpha^-, t^l), \qquad (\beta_x^+)^l = \beta_x(\alpha^+, t^l).$$

The correction term C_j^l is

$$C_j^l = \gamma_{j,3}^l\left\{\hat{C}^l + (x_{j+1}-\alpha)\,\frac{C^l}{(\beta^+)^l}\right\} - \gamma_{j,3}^l\,\frac{(x_{j+1}-\alpha)^2}{2}\left\{\frac{(\beta_x^+)^l\,C^l}{\{(\beta^+)^l\}^2} + \frac{(\kappa^+)^l\,\hat{C}^l + \left(\hat{C}'\right)^l - [f]^l}{(\beta^+)^l}\right\},$$

where

$$\left(\hat{C}'\right)^l = \frac{d}{dt}\hat{C}\left(\alpha, t^l\right).$$

Notice that now we have an extra term $(\hat{C}')^l$ compared to the correction term in the general 1D elliptic problem due to the $[u_t]$ term. Similarly at the grid point x_{j+1}, the coefficients $\gamma_{j+1,1}^l$, $\gamma_{j+1,2}^l$, and $\gamma_{j+1,3}^l$ for $l = n,\ n+1$, satisfy the following system of equations:

$$\left(1 + \frac{(x_j-\alpha)^2}{2\,(\beta^-)^l}\,[\kappa]^l\right)\,\gamma_{j+1,1}^l + \gamma_{j+1,2}^l + \gamma_{j+1,3}^l = 0,$$

$$\left\{\frac{(\beta^+)^l}{(\beta^-)^l}\,(x_j-\alpha) + \left(\frac{(\beta_x^+)^l}{(\beta^-)^l} - \frac{(\beta_x^-)^l\,(\beta^+)^l}{\{(\beta^-)^l\}^2}\right)\,\frac{(x_j-\alpha)^2}{2}\right\}\gamma_{j+1,1}^l + (x_{j+1}-\alpha)\,\gamma_{j+1,2}^l + (x_{j+2}-\alpha)\,\gamma_{j+1,3}^l = \left(\beta_x^+\right)^l,$$

$$\frac{(x_j-\alpha)^2}{2}\frac{(\beta^+)^l}{(\beta^-)^l}\,\gamma_{j+1,1}^l + \frac{(x_{j+1}-\alpha)^2}{2}\,\gamma_{j+1,2}^l + \frac{(x_{j+2}-\alpha)^2}{2}\,\gamma_{j+1,3}^l = (\beta^+)^l,$$

and the correction term now is

$$C_{j+1}^l = \gamma_{j+1,1}^l \left\{ -\hat{C}^l + (\alpha - x_j)\frac{C^l}{(\beta^-)^l} \right\} - \gamma_{j+1,1}^l \frac{(\alpha - x_j)^2}{2} \left\{ \frac{(\beta_x^-)^l C^l}{\{(\beta^-)^l\}^2} - \frac{(\kappa^-)^2 \hat{C}^l + \left(\hat{C}'\right)^l - [f]^l}{(\beta^-)^l} \right\}.$$

We have written a Fortran package for the one-dimensional heat problems with a fixed interface and tested several numerical examples. All the results confirmed second order convergence. More importantly, the method is always stable no matter how large the jumps in coefficients, provided that $\beta(x,t)$ has the sign across the interface. This will not be true for the two-dimensional case when we have arbitrary interfaces.

3.2 ADI methods for heat equations with discontinuities along an arbitrary interface

3.2.1 Introduction

In this section we present a new, second order accurate ADI method for the heat equation

$$u_t = (\beta u_x)_x + (\beta u_y)_y - f(x,y,t) \tag{3.4}$$

in a domain Ω in two space dimensions. Within the region Ω suppose there is an irregular interface Γ (see Fig. (1.2)) across which the solution $u(x,y,t)$ or some of its derivatives are known to be discontinuous, and the source term can be also discontinuous or even singular. We assume that the coefficient β is continuous in this section.

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 $C(s,t)$.

Again we assume that Ω is a simple domain such as a rectangle, and that we wish to solve the equation using a finite difference method on a regular grid, e.g. a uniform Cartesian grid. The interface is typically not aligned with the grid but rather cuts between grid points. We assume we know the jump condition in the solution u and normal derivative u_n across Γ. As we mentioned earlier those jumps can often be derived from the differential equations.

For parabolic equations, it is often desirable to use implicit methods because the time step restriction is severe for explicit methods. In fact with some effort, we can get a second order accurate difference scheme for this problem by using the *Immersed Interface Method* (IIM) proposed in previous chapters and [31]. The Crank-Nicolson difference scheme when $\beta = 1$ can be written as

$$\frac{u_{ij}^{n+1} - u_{ij}^n}{\tau} = \frac{1}{2}\left(\delta_x u_{ij}^n + \delta_y u_{ij}^n - C_{ij}^n\right) + \frac{1}{2}\left(\delta_x u_{ij}^{n+1} + \delta_y u_{ij}^{n+1} - C_{ij}^{n+1}\right) - f_{ij}^{n+\frac{1}{2}}, \tag{3.5}$$

where τ is the time step and

$$\delta_x u_{ij}^n = \left(u_{i-1,j}^n - 2u_{ij}^n + u_{i+1,j}^n \right) / h^2,$$
$$\delta_y u_{ij}^n = \left(u_{i,j-1}^n - 2u_{ij}^n + u_{i,j+1}^n \right) / h^2.$$

The correction term is determined from the jump conditions in $[u]$ and in $[u_n]$ (see Chapter 2 and [31]). But at each time step, the implicit version of the IIM leads to a linear system of equations which cannot be solved efficiently by direct methods.

Historically people have used a variety of splitting or ADI methods for parabolic PDEs. In these methods 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, giving a banded system of equations which can be solved easily. The reader is referred to [9], [14], [39] and [52] for an introduction to many of the methods.

However, such methods usually have strong demands on the smoothness of the solution. The classical ADI method, for example, when $\beta = 1$ is:

$$\frac{u_{ij}^{n+\frac{1}{2}} - u_{ij}^n}{\tau/2} = \delta_x u_{ij}^{n+\frac{1}{2}} + \delta_y u_{ij}^n - f_{ij}^{n+\frac{1}{2}}, \tag{3.6}$$

$$\frac{u_{ij}^{n+1} - u_{ij}^{n+\frac{1}{2}}}{\tau/2} = \delta_x u_{ij}^{n+\frac{1}{2}} + \delta_y u_{ij}^{n+1} - f_{ij}^{n+\frac{1}{2}}. \tag{3.7}$$

For this method the local truncation error contains a term of the form

$$\tau^2 \, \delta_x^2 \, \delta_y^2 \, u_t \approx \tau^2 u_{txxyy}, \tag{3.8}$$

if the solution belongs to C^4.

Certainly we have difficulty in applying these methods directly to our problems because the solution may not even belong to C^0. It seems that we can regard the correction term in (3.5) as part of $f(x, y, t)$ and use the ADI method directly which would be

$$\frac{u_{ij}^{n+\frac{1}{2}} - u_{ij}^n}{\tau/2} = \delta_x u_{ij}^{n+\frac{1}{2}} + \delta_y u_{ij}^n - C_{ij}^{n+\frac{1}{2}} - f_{ij}^{n+\frac{1}{2}}, \tag{3.9}$$

$$\frac{u_{ij}^{n+1} - u_{ij}^{n+\frac{1}{2}}}{\tau/2} = \delta_x u_{ij}^{n+\frac{1}{2}} + \delta_y u_{ij}^{n+1} - C_{ij}^{n+\frac{1}{2}} - f_{ij}^{n+\frac{1}{2}}. \tag{3.10}$$

Unfortunately, theoretical analysis and numerical experiments shows this scheme only gives first order accuracy. The failure results from the fact that we have not split the correction term $C_{i,j}^{n+\frac{1}{2}}$.

In this section we still use a five point stencil. At regular grid points the classical ADI scheme is used. At each irregular grid point we add some correction terms in each sweep so that the local truncation error is order $O(h)$. In order to derive those correction terms we use the idea proposed in [31] but now split the correction terms in the x and y directions accordingly. Because the number of irregular grid points is $O(n)$ or even smaller we can guarantee the global error in the solution to be $O(h^2)$.

3.2.2 The notations and difference scheme

Let the parametric expression of the interface Γ be $x = x(s)$, $y = y(s)$, where s is arc length. We assume we know the jump conditions $[u]$ and $[\partial u/\partial n]$

$$[u] = w(s), \qquad [\partial u/\partial n] = v(s), \tag{3.11}$$

along the interface which can often be derived from the PDE's as in the example above. They are used to derive appropriate correction terms so that the standard five-point difference operator yields second order accuracy as in the work of Mayo [34], [35] (see also [31]). The domain Ω is assumed to be rectangular, say $[a,b]\times[c,d]$. We use a uniform $M \times N$ grid with

$$\begin{aligned} x_i &= a + ih, \qquad i = 0, 1, \cdots, M. \\ y_j &= a + jh, \qquad j = 0, 1, \cdots, N. \end{aligned}$$

where $h = (b-a)/M = (d-c)/N$. For simplicity, we will take $\beta = 1$, and concentrate our attention on the difference scheme at irregular grid points. Figure 3.1 gives an example of such points and their geometry.

Difference scheme: Our ADI method can be written as

$$\begin{aligned} \frac{u_{ij}^{n+\frac{1}{2}} - u_{ij}^{n}}{\tau/2} &= \delta_x u_{ij}^{n+\frac{1}{2}} - (C_x)_{ij}^{n+\frac{1}{2}} - Q_{ij}^{n} - R_{ij}^{n} + \delta_y u_{ij}^{n} - (C_y)_{ij}^{n} - f_{ij}^{n+\frac{1}{2}}, \\ \frac{u_{ij}^{n+1} - u_{ij}^{n+\frac{1}{2}}}{\tau/2} &= \delta_x u_{ij}^{n+\frac{1}{2}} - (C_x)_{ij}^{n+\frac{1}{2}} - Q_{ij}^{n} - R_{ij}^{n} \\ &\quad + \delta_y u_{ij}^{n+1} - (C_y)_{ij}^{n+1} - f_{ij}^{n+\frac{1}{2}}. \end{aligned} \tag{3.12}$$

At regular grid points the standard ADI method is used, in which

$$(C_x)_{ij}^{n+\frac{1}{2}} = Q_{ij}^{n} = R_{ij}^{n} = (C_y)_{ij}^{n} = Q_{ij}^{n} = (C_y)_{ij}^{n+1} = 0.$$

At each irregular grid point we need to determine these correction terms.

3.2.3 Splitting the correction terms.

We know the local truncation error at regular grid points is $O(h^2)$. To obtain second order accuracy globally we need the local truncation error to be order $O(h)$ at irregular grid points. First we try to approximate u_{xx} and u_{yy} to first order by choosing the correction term C_x and C_y and later on we try to choose the appropriate correction term R_{ij}^n and Q_{ij}^n so that the local truncation error at each irregular grid point is $O(h)$.

Take a typical irregular grid point (x_i, y_j) or (i,j) in short. Assume the interface Γ cuts the straight line $y = y_j$ at $x = x_{ij}^*$, where $x_i \le x_{ij}^* \le x_{i+1}$. Take Figure 3.1 as an example, at the point (i,j) using Taylor expansion about x^*, (it really should be x_{ij}^*, but for simplicity, we drop the i, j). After some algebra we have

$$\frac{u_{i-1,j} - 2u_{ij} + u_{i+1,j}}{h^2} = \frac{u^- - 2u^- + u^+}{h^2} + u_x^- \left(\frac{x_{i-1} - x^*}{h^2} - 2\,\frac{x_i - x^*}{h^2} \right)$$

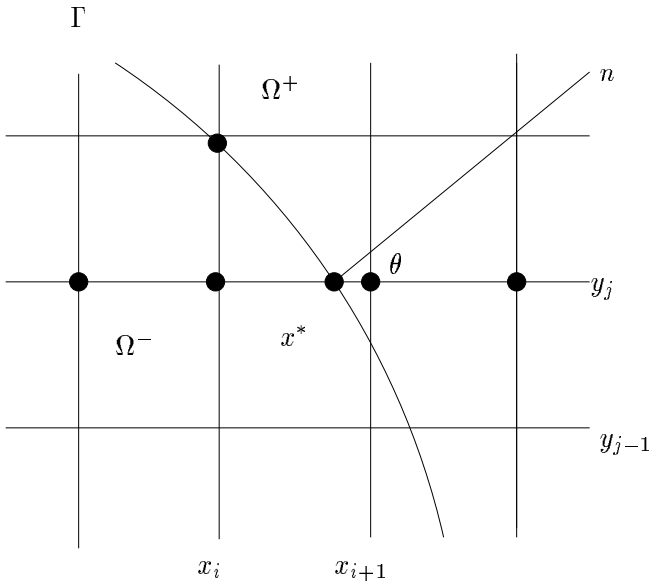


Figure 3.1: The geometry at some irregular points near the interface

$$
\begin{aligned}
&+ u_x^+ \frac{x_{i+1} - x^*}{h^2} + u_{xx}^- \left(\frac{(x_{i-1} - x^*)^2}{2h^2} - 2\frac{(x_i - x^*)^2}{2h^2} \right) \\
&+ u_{xx}^+ \frac{(x_{i+1} - x^*)^2}{2h^2} + O(h) \\
&= u_{xx}^- + \frac{[u]}{h^2} + [u_x] \frac{(x_{i+1} - x^*)}{h^2} + [u_{xx}] \frac{(x_{i+1} - x^*)^2}{2h^2} + O(h) \\
&= u_{xx}^- + (C_x)_{ij} \qquad (3.13)
\end{aligned}
$$

Note that all the quantities are computed at $(x^*, y^*) \in \Gamma$, where $y^* = y_j$ in Figure 3.1, at some time t_n, where for convenience we didn't write time indices in the expressions above. Similarly at the point $(i+1, j)$, we have

$$
\begin{aligned}
\frac{u_{i,j} - 2u_{i+1,j} + u_{i+2,j}}{h^2} &= u_{xx}^+ - \frac{[u]}{h^2} - [u_x] \frac{(x_i - x^*)}{h^2} - [u_{xx}] \frac{(x_i - x^*)^2}{2h^2} + O(h) \\
&= u_{xx}^+ + (C_x)_{i+1,j}. \qquad (3.14)
\end{aligned}
$$

In other words for any grid point (i, j) we can write

$$\delta_x u_{i,j} = u_{xx} + (C_x)_{ij} + O(h) \qquad \text{at irregular points} \qquad (3.15)$$

where

$$(C_x)_{ij} = \pm \frac{[u]}{h^2} \pm [u_x] \frac{(x_{i\pm1} - x^*)}{h^2} \pm [u_{xx}] \frac{(x_{i\pm1} - x^*)^2}{2h^2}. \qquad (3.16)$$

If the point is regular then $(C_x)_{ij} = 0$, otherwise it can be expressed in terms of x direction jumps $[u], [u_x]$ and $[u_{xx}]$. The sign is determined by the relative position of the interface Γ

and the point (x_i, y_j). We can do the same in the y direction to get

$$\delta_y u_{i,j} = u_{yy} + (C_y)_{ij} + O(h) \qquad \text{at irregular points.} \tag{3.17}$$

Now we need to find the jumps $[u_x], [u_y], [u_{xx}], [u_{yy}]$ in terms of the known information $[u]$ and $[u_n]$. If we are able to find these jumps then we can split the correction terms into the x direction and y direction, respectively. Fortunately, with the results obtained in Chapter 2, we are able to get these jumps.

We use the local coordinates transformation at (x^*, y^*) as we did in [31]:

$$\begin{aligned} \xi &= (x - x^*)\cos\theta + (y - y^*)\sin\theta, \\ \eta &= -(x - x^*)\sin\theta + (y - y^*)\cos\theta. \end{aligned}$$

Here θ is the angle between the $x-$axis and the normal direction at (x^*, y^*). In a neighborhood of this point, the interface lies roughly in the η-direction, so we can parameterize Γ locally by $\xi = \chi(\eta),\ \eta = \eta$, and we write the jumps as $[u] = w(\eta, t),\ [u_n] = v(\eta, t)$. For convenience we use the same notation for u, w, and v both in the original coordinates and the new local coordinates. It is easy to see that

$$\begin{aligned} u_\xi &= u_x \cos\theta + u_y \sin\theta, \\ u_\eta &= -u_x \sin\theta + u_y \cos\theta. \end{aligned}$$

After the local coordinate transformation we get

$$u_\xi = u_n. \tag{3.18}$$

So the jumps in u_ξ, u_η are

$$\begin{aligned} [u_\xi] &= [u_n] = v(\eta, t), \\ [u_\eta] &= [u]_\eta = w_\eta(\eta, t), \end{aligned} \tag{3.19}$$

which are known already. For the remaining second derivative jumps we can use the interface relations from Chapter 2 with minor modifications to get

$$\begin{aligned} [u_{\xi\xi}] &= \chi''[u_\xi] - w_{\eta\eta} + [f] + w_t, \\ [u_{\eta\eta}] &= -\chi''[u_\xi] + w_{\eta\eta}, \\ [u_{\xi\eta}] &= \chi''[u_\eta] + v_\eta. \end{aligned} \tag{3.20}$$

Now we have expressed all the jumps in the local coordinates in terms of the known quantities $[u]$ and $[u_n]$. From (3.19)–(3.20) we can get the jumps in the x and y directions, respectively, using the following formulas:

$$\begin{aligned} [u_x] &= [u_\xi]\cos\theta - [u_\eta]\sin\theta, \\ [u_y] &= [u_\xi]\sin\theta + [u_\eta]\cos\theta, \\ [u_{xx}] &= [u_{\xi\xi}]\cos^2\theta - 2[u_{\xi\eta}]\cos\theta\sin\theta + [u_{\eta\eta}]\sin^2\theta, \\ [u_{yy}] &= [u_{\xi\xi}]\sin^2\theta + 2[u_{\xi\eta}]\cos\theta\sin\theta + [u_{\eta\eta}]\cos^2\theta. \end{aligned} \tag{3.21}$$

With these known jumps we can compute $(C_x)_{ij}$, the split correction term in x direction, and $(C_y)_{ij}$, the split correction term in y direction respectively. Now we are ready to give the ADI method. It is natural to use the following ADI scheme:

$$\begin{aligned}
\frac{u_{ij}^{n+\frac{1}{2}} - u_{ij}^{n}}{\tau/2} &= \delta_x u_{ij}^{n+\frac{1}{2}} - (C_x)_{ij}^{n+\frac{1}{2}} + \delta_y u_{ij}^{n} - (C_y)_{ij}^{n} - f_{ij}^{n+\frac{1}{2}}, \\
\frac{u_{ij}^{n+1} - u_{ij}^{n+\frac{1}{2}}}{\tau/2} &= \delta_x u_{ij}^{n+\frac{1}{2}} - (C_x)_{ij}^{n+\frac{1}{2}} + \delta_y u_{ij}^{n+1} - (C_y)_{ij}^{n+1} - f_{ij}^{n+\frac{1}{2}}.
\end{aligned} \tag{3.22}$$

Numerical experiments gives promising results and the method appears to be second order accurate. But more subtle analysis (see §3.2.4) shows that the local truncation error is only $O(1)$ at irregular grid points which is also confirmed by the numerical results. So there must be some fortunate cancellation in the errors. At this point, we are not sure whether such cancellation will always occur or only happened in our test problems. Anyway, for safety, we are going to modify this ADI scheme further so that the local truncation errors at irregular grid points are $O(h)$ so we are guaranteed to get a second order accurate solution. We modify the ADI method in the next section.

3.2.4 Local truncation error analysis

In this section we discuss how to determine Q_{ij} and R_{ij} through local truncation error analysis . We assume that $\tau \sim h$. Now if we add the equations in (3.12) together we get

$$\begin{aligned}
\frac{u_{ij}^{n+1} - u_{ij}^{n}}{\tau} &= \delta_x u_{ij}^{n+\frac{1}{2}} - (C_x)_{ij}^{n+\frac{1}{2}} - Q_{ij}^{n} - R_{ij}^{n} \\
&\quad + \frac{1}{2}\left(\delta_y u_{ij}^{n} + \delta_y u_{ij}^{n+1}\right) - \frac{1}{2}\left((C_y)_{ij}^{n} + (C_y)_{ij}^{n+1}\right) - f_{ij}^{n+\frac{1}{2}}.
\end{aligned} \tag{3.23}$$

And if we subtract them, we have the intermediate result

$$u_{ij}^{n+\frac{1}{2}} = \frac{u_{ij}^{n} + u_{ij}^{n+1}}{2} + \frac{\tau}{4}\left(\delta_y u_{ij}^{n} - (C_y)_{ij}^{n} - \delta_y u_{ij}^{n+1} + (C_y)_{ij}^{n+1}\right). \tag{3.24}$$

Plugging this into (3.23), we get

$$\begin{aligned}
\frac{u_{ij}^{n+1} - u_{ij}^{n}}{\tau} &= \frac{1}{2}\delta_x\left(u_{ij}^{n} + u_{ij}^{n+1}\right) - (C_x)_{ij}^{n+\frac{1}{2}} - Q_{ij}^{n} - R_{ij}^{n} \\
&\quad + \frac{1}{2}\left(\delta_y u_{ij}^{n} - (C_y)_{ij}^{n} + \delta_y u_{ij}^{n+1} - (C_y)_{ij}^{n+1}\right) \\
&\quad + \frac{\tau}{4}\delta_x\left(\delta_y u_{ij}^{n} - (C_y)_{ij}^{n} - \delta_y u_{ij}^{n+1} + (C_y)_{ij}^{n+1}\right).
\end{aligned} \tag{3.25}$$

This is the *difference scheme* which we actually use to get the next time solution u_{ij}^{n+1}. At regular grid points we can regard those $(C_x)_{ij}^{n+\frac{1}{2}}, Q_{ij}^{n}, R_{ij}^{n}, (C_y)_{ij}^{n}, (C_y)_{ij}^{n+1}$ as zero. Now we consider the local truncation error so that we can determine the correction Q_{ij}^{n} and R_{ij}^{n}, keeping in mind that the interface is fixed and all the terms are continuous with time. The left hand side is

$$\frac{u_{ij}^{n+1} - u_{ij}^{n}}{\tau} = (u_t)_{ij}^{n+\frac{1}{2}} + O(\tau^2). \tag{3.26}$$

The first few terms on the right hand side can be written

$$\begin{aligned}\frac{1}{2}\delta_x\left(u_{ij}^n + u_{ij}^{n+1}\right) - (C_x)_{ij}^{n+\frac{1}{2}} - Q_{ij}^n = \\ \frac{1}{2}\delta_x u_{ij}^{n+\frac{1}{2}} + \tau^2\,\delta_x(u_{tt})_{ij}^{n+\frac{1}{2}} - (C_x)_{ij}^{n+\frac{1}{2}} - Q_{ij}^n.\end{aligned} \tag{3.27}$$

So if u is continuous, that is $[u] = w = 0$, then

$$\tau^2\,\delta_x(u_{tt})_{ij}^{n+\frac{1}{2}} \sim \tau^2/h \sim h. \tag{3.28}$$

We can simply take $Q_{ij}^n \equiv 0$. If u is not continuous, we rewrite the expression above as

$$\begin{aligned}&\frac{1}{2}\delta_x\left(u_{ij}^n + u_{ij}^{n+1}\right) - (C_x)_{ij}^{n+\frac{1}{2}} - Q_{ij}^n \;= \\ &\quad\frac{1}{2}\left(\delta_x u_{ij}^n - (C_x)_{ij}^n\right) + \frac{1}{2}\left(\delta_x u_{ij}^{n+1} - (C_x)_{ij}^{n+1}\right) \\ &\quad-(C_x)_{ij}^{n+\frac{1}{2}} + \frac{1}{2}\left((C_x)_{ij}^n + (C_x)_{ij}^{n+1}\right) - Q_{ij}^n = \\ &\quad\frac{1}{2}\left((u_{xx})_{ij}^n + (u_{xx})_{ij}^{n+1}\right) - (C_x)_{ij}^{n+\frac{1}{2}} + \frac{1}{2}\left((C_x)_{ij}^n + (C_x)_{ij}^{n+1}\right) - Q_{ij}^n + O(h) \\ &\qquad= \;(u_{xx})_{ij}^{n+\frac{1}{2}} - (C_x)_{ij}^{n+\frac{1}{2}} + \frac{1}{2}\left((C_x)_{ij}^n + (C_x)_{ij}^{n+1}\right) - Q_{ij}^n + O(h).\end{aligned}$$

Now we know we should take

$$Q_{ij}^n = \frac{1}{2}\left((C_x)_{ij}^n + (C_x)_{ij}^{n+1}\right) - (C_x)_{ij}^{n+\frac{1}{2}}. \tag{3.29}$$

Notice that

$$\begin{aligned}&\frac{1}{2}\left(\delta_y u_{ij}^n - (C_y)_{ij}^n + \delta_y u_{ij}^{n+1} - (C_y)_{ij}^{n+1}\right) \\ &\qquad= \;\frac{1}{2}\left((u_{yy})_{ij}^n + (u_{yy})_{ij}^{n+1}\right) + O(h) \\ &\qquad= \;(u_{yy})_{ij}^{n+\frac{1}{2}} + O(h).\end{aligned} \tag{3.30}$$

So these terms do not cause any trouble. Finally we turn our attention to the remaining terms. Recall that the interface is fixed and all quantities are continuous with time so we can write

$$\begin{aligned}\delta_y u_{ij}^n - (C_y)_{ij}^n &= (u_{yy})_{ij}^n + S_{ij}^n h + O(h^2), \\ \delta_y u_{ij}^{n+1} - (C_y)_{ij}^{n+1} &= (u_{yy})_{ij}^{n+1} + S_{ij}^{n+1} h + O(h^2).\end{aligned} \tag{3.31}$$

Here $|S_{ij}^n - S_{ij}^{n+1}| = O(h)$. Hence from (3.17) we know that

$$\delta_y u_{ij}^n - (C_y)_{ij}^n - \delta_y u_{ij}^{n+1} + (C_y)_{ij}^{n+1} = (u_{yy})_{ij}^n - (u_{yy})_{ij}^{n+1} + O(h^2). \tag{3.32}$$

Therefore we have

$$\begin{aligned}
&\frac{\tau}{4}\delta_x\left(\delta_y u_{ij}^n - (C_y)_{ij}^n - \delta_y u_{ij}^{n+1} + (C_y)_{ij}^{n+1}\right) - R_{ij}^n \\
&\qquad = \frac{\tau}{4}\delta_x\left((u_{yy})^n - (u_{yy})^{n+1}\right) + O(h) - R_{ij}^n \\
&\qquad = -\frac{\tau^2}{4}\delta_x(u_{yyt})_{ij}^n - R_{ij}^n + O(h).
\end{aligned} \tag{3.33}$$

Now we see that the real trouble comes from the term $\frac{\tau^2}{4}\delta_x(u_{yyt})^n$ which is $O(1)$. We have to approximate this term to accuracy $O(h)$ to make the right correction. However, notice that

$$\begin{aligned}
\frac{\tau^2}{4}\delta_x(u_{yyt})_{ij}^n &= \frac{1}{4}\left((u_{yyt})_{i-1,j}^n - 2(u_{yyt})_{ij}^n + (u_{yyt})_{i+1,j}^n\right) \\
&= \pm\frac{1}{4}[u_{yy}]_t^n + O(h).
\end{aligned} \tag{3.34}$$

Again the sign is determined by the relative position of the (i,j) grid point and the interface. For example, in Figure 3.1, we would have a $-$ sign for (i,j), and a $+$ sign for $(i+1,j)$. Fortunately we have expressed the jump $[u_{yy}]$ along the interface in terms of the known jumps $[u_n]$ (see (3.19)–(3.21)). So

$$(u_{yyt})_{ij}^n = \frac{[u_{yy}]^{n+1} - [u_{yy}]^n}{\tau} + O(h). \tag{3.35}$$

At last we can determine the R_{ij}^n, which is taken as

$$R_{ij}^n = \mp\,\frac{[u_{yy}]^{n+1} - [u_{yy}]^n}{4\,\tau}. \tag{3.36}$$

From the analysis above we know that if we take Q_{ij}^n and R_{ij}^n as in (3.29) and (3.36), then we can guarantee that the local truncation errors are $O(h^2)$ at regular grid points, and $O(h)$ at the irregular grid points near the boundary. Since the boundary is one dimension lower than the whole problem, the solution to the difference scheme will still give second order accurate solution globally.

3.2.5 Numerical results

We have done a number of numerical tests. All of them confirmed our analysis in §3.2.4. With the correction terms Q_{ij}^n and R_{ij}^n, we observed second order accuracy globally in the solution. Without these terms, the convergence rate also seemed to be nearly second order accurate, but we are not sure whether it is generally true or not.

Example 3.1 In this example the solution $u(x,y,t)$ is continuous but has a constant jump in the normal derivative. The differential equation is:

$$\begin{aligned}
&u_t = u_{xx} + u_{yy} + \int_\Gamma C(t)\,\delta(x - X(s))\,\delta(y - Y(s))\,ds, \\
&C(t) = e^{-t}\left(Y_0'(0.5)\,J_0(0.5)/Y_0(0.5) - J_0'(0.5)\right),
\end{aligned} \tag{3.37}$$

on the square $-1 \le x,\, y \le 1$, where Γ is the circle $x^2 + y^2 = 1/4$. In (3.37), J_0 and Y_0 are the Bessel functions (of order 0) of the first and second kind, respectively. The Dirichlet boundary condition and the initial condition are taken from the exact solution:

$$u(x,y,t) = \begin{cases} e^{-t} J_0(r) & \text{if } r \le \frac{1}{2} \\ e^{-t} J_0(0.5)\, Y_0(r)/Y_0(0.5) & \text{if } r > \frac{1}{2}. \end{cases} \tag{3.38}$$

Table 3.1 and Table 3.2 give the numerical result at $t = 5$ with and without the correction terms $Q^n_{i,j}$ and $R^n_{i,j}$ respectively. We see that the method with correction terms $Q^n_{i,j}$ and $R^n_{i,j}$ behaves better in this example. But the method without correction terms seems also to approach second order accuracy. Figure 3.2 shows the solution.

The maximum error over all grid points,

$$\| E_N \|_\infty = \max_{i,j} \{ | u(x_i, y_j) - u_{ij} | \},$$

is presented, where u_{ij} is the computed approximation at the uniform grid points (x_i, y_j). For our method we also display $\| T_N \|_\infty$, the infinity norm of the local truncation error over all grid points. The local truncation errors are $O(h^2)$ except at those points which are close to the interface where they are $O(h)$. We also display the ratios of successive errors,

$$\text{ratio1} = \| E_{2\,N} \|_\infty / \| E_N \|_\infty, \quad \text{ratio2} = \quad \| T_{2\,N} \|_\infty / \| T_N \|_\infty.$$

A ratio of 2 corresponds to first order accuracy, while a ratio of 4 indicates second order accuracy. We will use the same notation for other examples in this section.

Table 3.1: Numerical results for Example 3.1 with correction terms $Q^n_{i,j}$ and $R^n_{i,j}$

N	$\| E_N \|_\infty \quad t = 5$	ratio1	$\| T_N \|_\infty \quad t = 0$	ratio2
20	5.39851×10^{-5}		4.04973×10^{-1}	
40	1.01368×10^{-5}	4.6345	1.885347×10^{-1}	2.1480
80	2.30004×10^{-6}	4.4072	8.94152×10^{-2}	2.1085
160	5.56747×10^{-7}	4.1312	4.33057×10^{-2}	2.0647

Example 3.2 In this example we impose a jump in the function $u(x,y,t)$ itself and also a jump in the normal derivative of $u(x,y,t)$. These jumps vary along the interface and are not symmetric. The partial differential equation is

$$u_t = u_{xx} + u_{yy} - f(x,y,t). \tag{3.39}$$

The jumps in $u(x,y,t)$ and $\partial u/\partial n$ are chosen from the following exact solution:

$$u(x,y,t) = \begin{cases} 1 & \text{if } r \le \frac{1}{2} \\ \cos(t) \sin(\frac{\pi}{4}(x+1)) \sin(\frac{\pi}{4}(y+1)) & \text{if } r > \frac{1}{2}. \end{cases} \tag{3.40}$$

Table 3.2: Numerical results for Example 3.1 without correction terms Q_{ij}^n and R_{ij}^n

N	$\| E_N \|_\infty \quad t = 5$	ratio1	$\| T_N \|_\infty \quad t = 0$	ratio2
20	7.70093×10^{-5}		2.32077	
40	2.07288×10^{-5}	3.7151	2.60534	0.8908
80	5.44382×10^{-6}	3.8078	2.73938	0.9511
160	1.42010×10^{-6}	3.8334	2.80317	0.9772

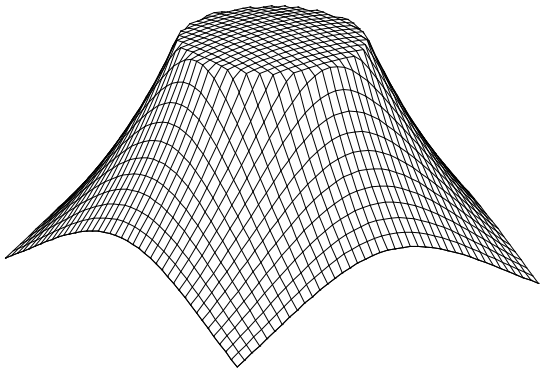


Figure 3.2: The solution ($N = 40, t = 5$) for Example 3.1 with the jump given in the normal derivative along the interface.

The function $f(x, y, t)$ is defined as:

$$f(x,y,t) = \begin{cases} 0 & \text{if } r \le \frac{1}{2} \\ \\ \Big(\sin(t) - \frac{\pi^2}{16}\cos(t)\Big)\sin(\frac{\pi}{4}(x+1))\sin(\frac{\pi}{4}(y+1)) & \text{if } r > \frac{1}{2}. \end{cases}$$

Table 3.3 and Table 3.4 give the errors of the computed solution with and without the correction terms Q_{ij}^n and R_{ij}^n. Figure 3.3 is the solution plotted at time 2π.

In summary we have developed a second order accurate Alternating Direction Implicit Method for the heat equation with singular sources (giving jumps in u_n) and dipoles (giving jumps in u) along some fixed interface. The same method can be used for arbitrary continuous coefficients β with a little change. Numerical experiments have confirmed the efficiency of the methods proposed in this section.

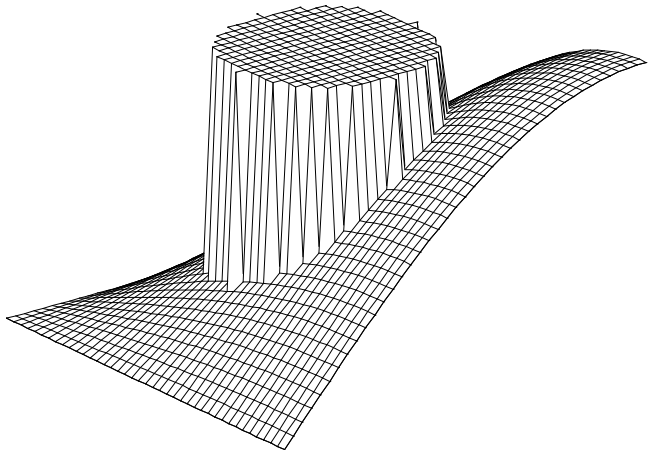


Figure 3.3: The solution ($N = 40, t = 2\pi$) for Example 3.2 with given jumps which are not symmetric.

Table 3.3: Numerical results for Example 3.2 with correction terms Q_{ij}^n and R_{ij}^n

n	$\| E_N \|_\infty \quad t = 5$	ratio1	$\| T_N \|_\infty \quad t = \pi/2$	ratio2
20	9.28331×10^{-4}		7.14922×10^{-3}	
40	2.46366×10^{-4}	3.6943	3.51033×10^{-3}	2.0366
80	6.47184×10^{-5}	3.8067	1.73839×10^{-3}	2.0193
160	1.66843×10^{-5}	3.8790	8.64918×10^{-4}	2.0099
320	4.25187×10^{-6}	3.9240	4.31816×10^{-4}	2.0030

Table 3.4: Numerical results for Example 3.2 without correction terms Q_{ij}^n and R_{ij}^n

n	$\| E_N \|_\infty \quad t = 5$	ratio1	$\| T_N \|_\infty \quad t = \pi/2$	ratio2
20	9.28331×10^{-4}		0.17446	
40	2.47616×10^{-4}	3.7491	0.16917	1.0313
80	6.48125×10^{-5}	3.8205	0.16631	1.0172
160	1.66920×10^{-5}	3.88350	0.16483	1.0090

Chapter 4

SOLVING STOKES FLOW WITH MOVING INTERFACE IN 2D

In this chapter we apply the *Immersed Interface Method* (IIM) to solve the Stokes equation with a moving interface that exerts a singular force on an incompressible fluid in two dimensional space. This new approach gives better accuracy compared with other approaches and provides a new way to solve incompressible moving-interface problems. Theoretical analysis is given to deal with the distribution of the singular sources, namely the 2D delta function and its derivatives, arising in the problem. Techniques to interpolate non-smooth or discontinuous functions to the interface using grid values are also presented. We consider a model problem studied by Tu and Peskin [51] consisting of an elastic boundary immersed in a zero Reynolds number fluid.

The motivations for considering this problem are:

- We want to determine whether the immersed interface method can give second order accurate solutions for incompressible flow problems with moving interfaces. The problem considered here is the simplest of this form.

- Tu and Peskin's method for the Stokes flow is only first order accurate in the 1-norm, and less accurate in the infinity-norm if the solution is discontinuous. Furthermore, there is a slow leaking phenomena in their method which we can see from their paper [51][1]. We will give some explanation of this phenomena in §4.4. Such leaking should not occur because we are dealing with incompressible flow and an impermeable boundary. So we want to have a better method for the Stokes equations which gives higher order accuracy and also preserves the enclosed area.

4.1 The governing equation

The model problem is two dimensional Stokes flow taken from [51]. Some of the description of the problem in this section is excerpted from their paper. We consider a two dimensional viscous incompressible fluid containing an immersed elastic weightless boundary. Generally the boundary is stretched and under tension due to the motion of the fluid surrounding it or some external force. Reciprocally the boundary exerts force on the fluid as well and will influence the motion of the fluid on the whole domain so that the entire dynamic system can approach its equilibrium state. As an intuitive example, we consider a balloon filled with air. The equilibrium shape for the balloon is a sphere. Without external force the balloon will not deform even though the boundary force is nonzero since the balloon is stretched and under tension. The elastic boundary force is balanced by a jump in pressure. However, if we squeeze the balloon at some initial time and then let it go the balloon will change its

[1] Peskin and Printz have suggested a modification to fix the leaking problem in [44], and we have not tried to implement this modification for the Stokes equations described in this chapter.

shape with time until it reaches the equilibrium again. This example is very similar to the problem discussed in this chapter except we consider very viscous and incompressible fluid which surrounds the interface. We assume that effects of inertia are negligible both for the fluid and for the immersed boundary. Note that both the fluid inside and the fluid outside the immersed boundary are physically significant in the problem and influence the motion of the boundary. As in [51], we put the entire problem in a periodic box.

On the basis of the above assumption, we can write the equations of motion that are valid for both the fluid and the immersed boundary in the Stokes equations:

$$\nabla p = \nu \Delta \vec{u} + \vec{F}(\vec{x}, t), \tag{4.1}$$

$$\nabla \cdot \vec{u} = 0, \tag{4.2}$$

where $\vec{u}$ is the fluid velocity, p is the fluid pressure, ν is the constant fluid viscosity, and $\vec{F}$ is a boundary force such as an elastic force or surface tension. The immersed boundary is represented by the Lagrangian variable $\vec{X}(s,t)$, parameterized by s. Then we can write the force distribution using Dirac δ-function notation:

$$\vec{F}(\vec{x}, t) = \int_{\Gamma(t)} \vec{f}(s,t) \delta_2(\vec{x} - \vec{X}(s,t)) ds. \tag{4.3}$$

The integral is over the entire boundary and δ_2 is a two-dimensional δ-function $\delta_2(\vec{x}) = \delta_1(x)\delta_1(y)$, where $\vec{x} = (x, y)$. Therefore, the force is singular and has support only on the immersed boundary.

For an elastic boundary, the force is the restoring force of the stretched boundary. If we use s_0 to express the arc-length of the unstretched boundary, then there is a continuous mapping between s and s_0:

$$s_0 = \psi(s). \tag{4.4}$$

Let $T(s)$ be the tension in the immersed boundary. We assume a generalized Hooke's law response,

$$T(s) = \left| \frac{\partial \vec{X}}{\partial s_0} \right| - 1 = \left| \frac{\partial \vec{X}}{\partial s} \right| / |\psi'(s)| - 1. \tag{4.5}$$

Then the density function for the force exerted by the boundary on the fluid is

$$\vec{f}(s) = \frac{\partial}{\partial s_0} (T\vec{\tau}) = \frac{\partial}{\partial s} (T\vec{\tau}) / \psi'(s), \tag{4.6}$$

where $\vec{\tau}$ is the unit tangent to the boundary

$$\vec{\tau} = \frac{\partial \vec{X}}{\partial s} \Big/ \left| \frac{\partial \vec{X}}{\partial s} \right|. \tag{4.7}$$

The final assumption is that the fluid adheres to the boundary. Thus, we impose the no-slip condition

$$\frac{\partial \vec{X}}{\partial t}(s,t) = \vec{u}(\vec{X}(s,t), t). \tag{4.8}$$

We will directly use the dimensionless form of (4.1)–(4.2) and assume $\nu = 1$ in equation (4.1) for simplicity.

There are two approaches to solve the Stokes flow problem. The first approach is to solve for the pressure and the velocity simultaneously using (4.1)–(4.2) as was done by Tu & Peskin [51]. This is discussed in § 4.4. The second approach is to solve a set of three Poisson equations, since using the incompressibility condition, we can easily decouple the Stokes equations (4.1)–(4.2) into three Poisson problems:

$$\Delta p = \nabla \cdot \vec{F}, \tag{4.9}$$

$$\Delta u = p_x - F_1, \tag{4.10}$$

$$\Delta v = p_y - F_2, \tag{4.11}$$

where $\vec{F} = (F_1, F_2)$; F_1 and F_2 are the components of the force in x- and y- directions, respectively and we take $\nu = 1$ from now on. The details will be given in § 4.5. We have found that the second approach works better in conjunction with the immersed interface method.

4.2 The derivation of the jump conditions

In order to use the IIM to solve this interface problem we need to derive the jump conditions. The velocity is continuous because of the no-slip boundary condition, but the pressure is not and there are also jumps in the normal derivatives of p, u, and v.

First we review some results from distribution theory. Below we will assume that $\phi(x, y)$ is an arbitrary twice continuously differentiable test function defined on an appropriate region Ω. Let $\delta(x - X_0)\delta(y - Y_0)$ be the two dimensional Dirac-function, a point source at (X_0, Y_0), is defined by

$$\iint_\Omega \phi(x, y)\, \delta(x - X_0)\delta(y - Y_0) = \phi(X_0, Y_0). \tag{4.12}$$

If we have a source distribution

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

along a curve $\Gamma : (X(s), Y(s))$ with strength $C(s)$, then

$$\begin{aligned} \iint_\Omega \phi(x, y) \left\{ \int_\Gamma C(s)\, \delta(x - X(s))\delta(y - Y(s))ds \right\} dx\, dy \\ = \int_\Gamma C(s)\, \phi\,(X(s), Y(s))\, ds. \end{aligned} \tag{4.13}$$

For a vector function $\vec{G} = [G_1(x, y), G_2(x, y)]^T$, by using Green's integral theorem, we know that

$$\iint_\Omega (\nabla \cdot \vec{G})\, \phi\, dx\, dy = \int_{\partial\Omega} (\vec{G} \cdot \vec{n})\, \phi\, ds - \iint_\Omega \vec{G} \cdot \nabla\phi\, dx\, dy.$$

Thus we can generalize the one dimensional result

$$\int \phi(x)\, \delta'(x - \alpha)\, dx = -\phi'(\alpha)$$

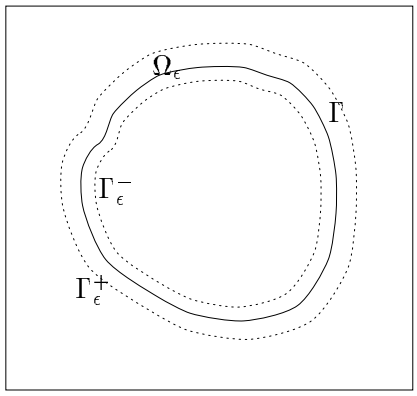


Figure 4.1: Diagram used for the derivation of jump relations

to two dimensions

$$\begin{aligned}\iint_\Omega \phi(x,y)\,\nabla\cdot\vec{F}\,dx\,dy &= \iint_\Omega \phi(x,y)\left\{\int_\Gamma \nabla\cdot\vec{f}(s,t)\;\delta_2\left(\vec{x}-\vec{X}(s,t)\right)\,ds\right\}\,dxdy \\ &= -\int_\Gamma \left(f_1(s,t)\,\frac{\partial\phi}{\partial x}\,(X(s),Y(s)) + f_2(s,t)\,\frac{\partial\phi}{\partial y}\,(X(s),Y(s))\right)ds. \qquad (4.14)\end{aligned}$$

Theorem 4.1 *For the Stokes equations (4.1)-(4.3) or (4.3) and (4.9)-(4.11), the following jump relations hold*[2]

$$\begin{aligned}[p](s) &= \hat{f}_1(s,t), &(4.15)\\ [p_n](s) &= \frac{\partial}{\partial s}\hat{f}_2(s,t), &(4.16)\\ [u_n](s) &= \hat{f}_2(s,t)\sin\theta, &(4.17)\\ [v_n](s) &= -\hat{f}_2(s,t)\cos\theta, &(4.18)\end{aligned}$$

where θ is the angle between the x-axis and the normal direction pointing outward from the immersed boundary at $(X(s),Y(s))$ and $\hat{f}_1,\hat{f}_2$ are the normal and tangential force strengths defined as

$$\begin{aligned}\hat{f}_1(s,t) &= f_1(s,t)\cos\theta + f_2(s,t)\sin\theta, &(4.19)\\ \hat{f}_2(s,t) &= -f_1(s,t)\sin\theta + f_2(s,t)\cos\theta. &(4.20)\end{aligned}$$

Proof: Referring to Fig. 4.1, we take a belt domain Ω_ϵ which encloses the interface Γ. Let Γ_ϵ^+ and Γ_ϵ^- be the outer and inner boundary of Ω_ϵ respectively. Here ϵ is some measure of the distance from Γ to Γ^+ and Γ^-.

From the Poisson equation (4.9) for pressure we have

$$\begin{aligned}\iint_{\Omega_\epsilon} \Delta p\,\phi\,dx\,dy &= \iint_{\Omega_\epsilon} \phi\,\nabla\cdot\vec{F}\,dx\,dy \\ &= -\int_\Gamma \left(f_1\,\frac{\partial\phi}{\partial x} + f_2\,\frac{\partial\phi}{\partial y}\right)ds. \qquad (4.21)\end{aligned}$$

[2] We have omitted the time dependence in our notation for some quantities such as $[p](s)$, θ etc. in order to simplify the notation. This does not affect our analysis.

Referring to Fig. 4.1, let us handle the left hand side of (4.21) first by using Green's theorem repeatedly

$$\begin{aligned}\iint_{\Omega_\epsilon} \Delta p\, \phi \, dx\, dy &= \int_{\Gamma_\epsilon^+} (\nabla p^+ \cdot \vec{n})\phi \, ds + \int_{\Gamma_\epsilon^-} (\nabla p^- \cdot (-\vec{n}))\phi \, ds - \iint_{\Omega_\epsilon} \nabla p \cdot \nabla \phi \, dx\, dy \\ &= \int_{\Gamma_\epsilon^+} p_n^+ \, \phi \, ds - \int_{\Gamma_\epsilon^-} p_n^- \, \phi \, ds - \int_{\Gamma_\epsilon^+} (\nabla \phi \cdot \vec{n}) \; p^+ \, ds \\ &\quad - \int_{\Gamma_\epsilon^-} (\nabla \phi \cdot (-\vec{n})) \; p^- \, ds + \iint_{\Omega_\epsilon} p \, \Delta \phi \, dx\, dy,\end{aligned}$$

where the superscripts + and − indicate the values taken from outside and inside of the interface Γ respectively. Notice that ϕ is twice continuously differentiable and p is bounded and discontinuous only along the interface. So as ϵ approaches zero, we have

$$\begin{aligned}\iint_{\Omega_\epsilon} p \, \Delta \phi \, dx\, dy &\longrightarrow 0 \qquad\qquad \text{as } \epsilon \to 0 \\ \iint_{\Omega_\epsilon} \Delta p \, \phi \, dx\, dy &\longrightarrow \int_\Gamma [p_n] \, \phi \, ds - \int_\Gamma [p] \, \phi_n \, ds \qquad \text{as } \epsilon \to 0. \end{aligned} \tag{4.22}$$

To deal with the right hand side of (4.21), we express $\partial\phi/\partial x$ and $\partial\phi/\partial y$ in terms of its normal and tangential derivatives along the interface

$$\phi_n = \nabla \phi \cdot \vec{n} = \frac{\partial \phi}{\partial x} \cos\theta + \frac{\partial \phi}{\partial y} \sin\theta, \tag{4.23}$$

$$\phi_s = \nabla \phi \cdot \vec{\tau} = -\frac{\partial \phi}{\partial x} \sin\theta + \frac{\partial \phi}{\partial y} \cos\theta. \tag{4.24}$$

After solving the linear equations above for $\partial\phi/\partial x$ and $\partial\phi/\partial y$ we get

$$\frac{\partial \phi}{\partial x} = \phi_n \cos\theta - \phi_s \sin\theta, \tag{4.25}$$

$$\frac{\partial \phi}{\partial y} = \phi_n \sin\theta + \phi_s \cos\theta. \tag{4.26}$$

Plugging these two equations in (4.21) and collecting terms we have

$$\begin{aligned}\int_\Gamma \left(f_1 \frac{\partial \phi}{\partial x} + f_2 \frac{\partial \phi}{\partial y} \right) ds &= \int_\Gamma \left((f_1 \cos\theta + f_2 \sin\theta) \frac{\partial \phi}{\partial n} + (f_2 \cos\theta - f_1 \sin\theta) \frac{\partial \phi}{\partial s} \right) ds \\ &= \int_\Gamma \left(\hat{f}_1 \phi_n + \hat{f}_2 \phi_s \right) ds.\end{aligned}$$

Integrating by parts in the second term and noting that Γ is closed, we may rewrite the equation above as

$$\int_\Gamma \left(f_1 \frac{\partial \phi}{\partial x} + f_2 \frac{\partial \phi}{\partial y} \right) ds = \int_\Gamma \left(\hat{f}_1 \phi_n - \frac{\partial \hat{f}_2}{\partial s} \phi \right) ds.$$

Comparing this with (4.22) and using the fact that ϕ is arbitrary, we must have

$$\begin{aligned}[p] &= \hat{f}_1, \\ [p_n] &= \frac{\partial \hat{f}_2}{\partial s}.\end{aligned}$$

To get the jump for u_n, we multiply by $\phi(x,y)$ on both sides of (4.10) and integrate

$$\iint_{\Omega_\epsilon} \Delta u\, \phi\, dx\, dy - \iint_{\Omega_\epsilon} p_x\, \phi\, dx\, dy = -\int_\Gamma f_1(s)\phi\, ds. \tag{4.27}$$

The first term of the left hand side of (4.27) is

$$\begin{aligned} \iint_{\Omega_\epsilon} \Delta u\, \phi\, dx\, dy &= \int_{\Gamma_\epsilon^+} u_n^+\, \phi\, ds - \int_{\Gamma_\epsilon^-} u_n^-\, \phi\, ds - \iint_{\Omega_\epsilon} (\Delta u \cdot \Delta\phi)\, dx\, dy \\ &\longrightarrow \int_\Gamma [u_n]\, \phi\, ds, \qquad \text{as } \epsilon \to 0. \end{aligned} \tag{4.28}$$

The second term of the left hand side of (4.27) is

$$\begin{aligned} \iint_{\Omega_\epsilon} p_x\, \phi\, dx\, dy &= \iint_{\Omega_\epsilon} \phi\, \nabla \cdot \begin{bmatrix} p \\ 0 \end{bmatrix} dx\, dy \\ &= \int_{\Gamma_\epsilon^+} \phi \left([p^+,\, 0]^T \cdot \vec{n} \right) ds - \int_{\Gamma_\epsilon^-} \phi \left([p^-,\, 0]^T \cdot \vec{n} \right) ds \\ &\quad - \iint_{\Omega_\epsilon} \left[\frac{\partial \phi}{\partial x}, \frac{\partial \phi}{\partial y} \right]^T \cdot \begin{bmatrix} p \\ 0 \end{bmatrix} dx\, dy \\ &\longrightarrow \int_\Gamma \phi\, [p]\, \cos\theta\, ds, \qquad \text{as } \epsilon \to 0, \end{aligned} \tag{4.29}$$

where $\vec{n} = [\, \cos\theta,\, \sin\theta\,]^T$. Since ϕ is arbitrary, from (4.19)–(4.29), we must have

$$\begin{aligned} [u_n] &= [p]\, \cos\theta - f_1 \\ &= \cos\theta\, (f_1 \cos\theta + f_2 \sin\theta) - f_1 \\ &= \sin\theta\, (-f_1\, \sin\theta + f_2\, \cos\theta) \\ &= \hat{f}_2\, \sin\theta. \end{aligned}$$

Similarly, for v we can get

$$\begin{aligned} [v_n] &= [p]\, \sin\theta - f_2 \\ &= \cos\theta\, (f_1\, \sin\theta - f_2\, \cos\theta) \\ &= -\hat{f}_2\, \cos\theta. \end{aligned}$$

This completes the proof of Theorem 4.1. □

Note we can also decompose the jump of the normal derivatives of p, u, and v into jumps in the x- and y- directions,

$$[p_x] = [p_n] \cos\theta - [p_s] \sin\theta, \tag{4.30}$$

$$[p_y] = [p_n] \sin\theta + [p_s] \cos\theta. \tag{4.31}$$

We know the jump $[p_n]$ already and

$$[p_s] = \frac{\partial p^+}{\partial s} - \frac{\partial p^-}{\partial s} = \frac{\partial}{\partial s}[p], \tag{4.32}$$

i.e., $[p_s]$ is the derivative of jump of pressure p along the interface. So we find

$$[p_x] = \frac{\partial \hat{f}_2}{\partial s}\cos\theta - \frac{\partial \hat{f}_1}{\partial s}\sin\theta, \tag{4.33}$$

$$[p_y] = \frac{\partial \hat{f}_2}{\partial s}\sin\theta + \frac{\partial \hat{f}_1}{\partial s}\cos\theta. \tag{4.34}$$

By the same token

$$[u_s] = \frac{\partial}{\partial s}[u] \equiv 0, \tag{4.35}$$

$$[v_s] = \frac{\partial}{\partial s}[v] \equiv 0. \tag{4.36}$$

So we can also compute $[u_x]$, $[u_y]$, $[v_x]$, $[v_y]$ according to the following:

$$[u_x] = [u_n]\cos\theta - [u_s]\sin\theta = \hat{f}_2 \sin\theta\cos\theta, \tag{4.37}$$

$$[u_y] = [u_n]\sin\theta + [u_s]\cos\theta = \hat{f}_2 \sin^2\theta, \tag{4.38}$$

$$[v_x] = [v_n]\cos\theta - [v_s]\sin\theta = -\hat{f}_2 \cos\theta\sin\theta, \tag{4.39}$$

$$[v_y] = [v_n]\sin\theta + [v_s]\cos\theta = -\hat{f}_2 \cos^2\theta. \tag{4.40}$$

In summary, we can compute the force strength from the interface configuration, and all the jumps in terms of the force strength and the properties of the interface.

4.3 Computational frame and jump calculations

For the periodic box $[a,b]\times[c,d]$ on which the Stokes flow is defined, we take a fixed uniform grid with

$$x_i = a + ih,\ y_j = a + jh,\ \ i,j = 0,1,\cdots,N,$$

where we assume that $h = (b-a)/N = (d-c)/N$. We use a periodic spline $\vec{X}(s,t) = (X(s,t), Y(s,t))$ passing through the Lagrangian points (X_k^n, Y_k^n) to express the moving interface, where s is the arc-length of the interface and (X_k^n, Y_k^n) is the position of the $k-$th point on the boundary at time $t = n\Delta t$, $k = 1,2,\cdots,N_b - 1$. Since the interface is a closed curve, the arithmetic on k is modulo N_b.

Any other quantity $w(s)$ defined on the interface such as the force strength or the jump in some quantity can also be expressed in terms of a periodic spline with the same parameter s. Since cubic splines are twice differentiable we can gain access to the value of $w(s)$ and its first or second derivatives at any point on the interface in a continuous manner.

To implement the immersed interface method, we calculate the jumps of $[p]$, $[p_n]$, $[u_n]$, and $[v_n]$ in the following steps:

- Determine the tangential and normal directions.

$$
\begin{aligned}
\text{Tangent vector} \quad \vec{\tau} &= \frac{\partial \vec{X}}{\partial s} \Big/ \left| \frac{\partial \vec{X}}{\partial s} \right|, \\
\frac{\partial \vec{X}}{\partial s} &= \left(\frac{\partial X}{\partial s}, \frac{\partial Y}{\partial s} \right). \\
\text{Normal vector} \quad \vec{n} &= (\cos\theta, \sin\theta), \qquad \text{where} \\
\cos\theta = \frac{\partial Y}{\partial s} \Big/ \left| \frac{\partial \vec{X}}{\partial s} \right|; & \qquad \sin\theta = -\frac{\partial X}{\partial s} \Big/ \left| \frac{\partial \vec{X}}{\partial s} \right|,
\end{aligned} \tag{4.41}
$$

where θ is the angle between the x-axis and the normal direction pointing outward from the immersed interface.

- Calculate the tension $T(s)$ and the force strength $\vec{f}(s,t)$.

$$
\begin{aligned}
T(s) &= \left| \frac{\partial \vec{X}}{\partial s_0} \right| - 1 = \left| \frac{\partial \vec{X}}{\partial s} \right| / \left| \psi'(s) \right| - 1, \\
\vec{f}(s,t) &= \frac{\partial}{\partial s_0} \left(T(s)\, \vec{\tau}(s) \right) \\
&= \frac{\partial}{\partial s} \left(T(s)\, \vec{\tau}(s) \right) / \psi'(s).
\end{aligned}
$$

We do not differentiate $w(s) = T(s)\,\vec{\tau}(s)$ directly, but use its value at s_k to form a new spline to get the first derivative. To get $\psi'(s)$ we do not need to know the continuous mapping $\psi(s)$ but only the corresponding relation between s_k and $(s_0)_k$, $k = 1, 2, \cdots, N_b - 1$. Then we can use a spline again to get the continuous expression and its derivatives.

We have substituted this spline approach for the finite difference method used to calculate the tension and force in Peskin and Tu's Stokes solver. While the convergence rate remains the same, this gives a smaller error constant in the results.

Usually we take equally spaced $\Delta(s_0)_k$ for the unstretched boundary. Different distributions of Δs_k will lead to different problems. Once we know the corresponding relation between s_0 and s_k, we can approximate $\psi'(s)$ using either discrete formulas or a spline interpolation.

- Compute the tangential and normal force strength using equations (4.19) and (4.20).

- Compute the jumps at a point s on the interface using (4.15)–(4.18), (4.30)–(4.31), and (4.37)–(4.40).

4.4 The direct Stokes solver

Since the Stokes flow is defined on the periodic box, it is natural to use the fast Fourier transform (FFT) to solve for the pressure and velocity simultaneously using (4.1)–(4.2) as discussed in [51]. In this section we will describe how to combine the immersed interface method with the FFT.

4.4.1 Discrete Stokes equations

For $i, j = 0, 1, \cdots, n-1$, the discrete form of the Stokes equations are

$$\begin{aligned} \frac{u_{i-1,j} + u_{i+1,j} + u_{i,j-1} + u_{i,j+1} - 4u_{ij}}{h^2} - \frac{p_{i+1,j} - p_{i-1,j}}{2h} &= f^1_{ij}, \\ \frac{v_{i-1,j} + v_{i+1,j} + v_{i,j-1} + v_{i,j+1} - 4v_{ij}}{h^2} - \frac{p_{i,j+1} - p_{i,j-1}}{2h} &= f^2_{ij}, \\ \frac{u_{i+1,j} - u_{i-1,j}}{2h} + \frac{v_{i,j+1} - v_{i,j-1}}{2h} &= q_{ij}. \end{aligned} \tag{4.42}$$

If (i, j) is a regular grid point, then

$$f^1_{ij} = f^2_{ij} = q_{ij} = 0. \tag{4.43}$$

At irregular grid points we need to find these quantities using the immersed interface method so that the local truncation errors of the three equations above is $O(h)$.

Suppose (i, j) is an irregular grid point, we take a point (x^*, y^*) on the interface near (x_i, y_j). From the previous section we are able to compute the jumps $[p]$, $[p_n]$, $[u_n]$ and $[v_n]$. We use the same coordinate transformation at (x^*, y^*) as in Chapter 2:

$$\xi = (x - x^*)\cos\theta + (y - y^*)\sin\theta, \tag{4.44}$$

$$\eta = -(x - x^*)\sin\theta + (y - y^*)\cos\theta, \tag{4.45}$$

where the θ has the same meaning as before, see (2.38)-(2.39).

Using Taylor expansion about (x^*, y^*) from both sides of the interface and collecting terms accordingly, we can write

$$\begin{aligned} T_{ij} &\stackrel{\text{def}}{=} \frac{u(x_{i-1}, y_j) + u(x_{i+1}, y_j) + u(x_i, y_{j+1}) + u(x_i, y_{j-1}) - 4u(x_i, y_j)}{h^2} - \\ &\qquad \frac{p(x_{i+1}, y_j) - p(x_{i-1}, y_j)}{2h} \\ &= a_3\, u^-_\xi + a_4\, u^+_\xi + a_5\, u^-_\eta + a_6\, u^+_\eta + a_7\, u^-_{\xi\,\xi} + a_8\, u^+_{\xi\,\xi} \\ &\quad + a_9\, u^-_{\eta\,\eta} + a_{10}\, u^+_{\eta\,\eta} + a_{11}\, u^-_{\xi\,\eta} + a_{12}\, u^+_{\xi\,\eta} + b_1\, p^- \\ &\quad + b_2\, p^+ + b_3\, p^-_\xi + b_4\, p^+_\xi + b_5\, p^-_\eta + b_6\, p^+_\eta + O(h), \end{aligned} \tag{4.46}$$

$$\begin{aligned} R_{ij} &\stackrel{\text{def}}{=} \frac{u(x_{i+1}, y_j) - u(x_{i-1}, y_j) + v(x_i, y_{j+1}) - v(x_i, y_{j-1})}{2h} \\ &= c_3\, u^-_\xi + c_4\, u^+_\xi + c_5\, u^-_\eta + c_6\, u^+_\eta + d_3\, v^-_\xi + d_4\, v^+_\xi + d_5\, v^-_\eta + d_6\, v^+_\eta, \end{aligned} \tag{4.47}$$

where a_i, b_i, c_i and d_i have expressions similar to (2.55) with different γ_k's, which are the coefficients of central difference scheme for u_{xx}, p_x, u_x and v_y respectively. Define

$$
\begin{aligned}
\phi_1 &= \sum_{k\in K^-} \gamma_k & \phi_2 &= \sum_{k\in K^+} \gamma_k \\
\phi_3 &= \sum_{k\in K^-} \xi_k\gamma_k & \phi_4 &= \sum_{k\in K^+} \xi_k\gamma_k \\
\phi_5 &= \sum_{k\in K^-} \eta_k\gamma_k & \phi_6 &= \sum_{k\in K^+} \eta_k\gamma_k \\
\phi_7 &= \frac{1}{2}\sum_{k\in K^-} \xi_k^2\gamma_k & \phi_8 &= \frac{1}{2}\sum_{k\in K^+} \xi_k^2\gamma_k \\
\phi_9 &= \frac{1}{2}\sum_{k\in K^-} \eta_k^2\gamma_k & \phi_{10} &= \frac{1}{2}\sum_{k\in K^+} \eta_k^2\gamma_k \\
\phi_{11} &= \sum_{k\in K^-} \xi_k\eta_k\gamma_k & \phi_{12} &= \sum_{k\in K^+} \xi_k\eta_k\gamma_k,
\end{aligned}
\tag{4.48}
$$

where K^+ and K^- is defined by

$$K^{\pm} = \{k:\ (\xi_k, \eta_k) \text{ is on the } \pm \text{ side of } \Gamma\}.$$

Then

$$
\begin{aligned}
&a_i = \phi_i,\ i = 3,4,\cdots,12, \text{ with } \gamma_1 = \gamma_3 = \gamma_4 = \gamma_5 = \frac{1}{h^2},\ \gamma_2 = -\frac{4}{h^2}, \\
&b_i = c_i = \phi_i,\ i = 3,4,5,6, \text{ with } \gamma_1 = -\gamma_3 = \frac{1}{2h},\ \gamma_2 = \gamma_4 = \gamma_5 = 0, \\
&d_i = \phi_i,\ i = 3,4,5,6, \text{ with } \gamma_5 = -\gamma_4 = \frac{1}{2h},\ \gamma_1 = \gamma_2 = \gamma_3 = 0.
\end{aligned}
$$

Notice that we have used the fact that u and v are continuous. With almost the same derivation as in Chapter 2 we can get

$$
\begin{aligned}
u_\eta^+ &= u_\eta^-, \\
u_\xi^+ &= u_\xi^- + [u_n], && [u_n] \text{ is known}, \\
u_{\xi\,\eta}^+ &= u_{\xi\,\eta}^- + \frac{\partial}{\partial\eta}[u_n], \\
u_{\eta\,\eta}^+ &= u_{\eta\,\eta}^- - [u_\xi]\chi'', \\
u_{\xi\,\xi}^+ &= u_{\xi\,\xi}^- + [u_\xi]\chi'' + [p_x], && [p_x] \text{ is known}.
\end{aligned}
\tag{4.49}
$$

Similar relations exist for v. It has been proved that

$$
\begin{aligned}
a_3 + a_4 &= 0 & a_9 + a_{10} &= 1 \\
a_5 + a_6 &= 0 & a_{11} + a_{12} &= 1. \\
a_7 + a_8 &= 1
\end{aligned}
$$

From the coordinate transformation (4.44) and (4.45) we also have

$$b_3 + b_4 = \frac{(x_{i+1} - x^*)\cos\theta + (y_j - y^*)\sin\theta}{2h} - \frac{(x_{i-1} - x^*)\cos\theta + (y_j - y^*)\sin\theta}{2h}$$

$$= \cos\theta,$$

$$b_5 + b_6 = \frac{-(x_{i+1} - x^*)\sin\theta + (y_j - y^*)\cos\theta}{2h} - \frac{-(x_{i-1} - x^*)\sin\theta + (y_j - y^*)\cos\theta}{2h}$$
$$= -\sin\theta.$$

Hence

$$(b_3 + b_4)\, p_\xi^- + (b_5 + b_6)\, p_\eta^- = \cos\theta\, p_\xi^- - \sin\theta\, p_\eta^- = p_x^-. \tag{4.50}$$

We can get similar relations for u_x and v_y. Now we can rewrite (4.46) and (4.47) as

$$\begin{aligned} T_{ij} &= u_{xx}^- + u_{yy}^- - p_x^- + (a_4 + (a_8 - a_{10})\chi'')\,[u_n] \\ &\quad + a_8[p_x] - b_2[p] - b_4[p_\xi] - b_6\,[p_\eta] + O(h) \end{aligned} \tag{4.51}$$

and

$$R_{ij} = u_x^- + v_y^- + c_4[u_\xi] + c_6[u_\eta] + d_4[v_\xi] + d_6[v_\eta] + O(h). \tag{4.52}$$

Therefore we should choose

$$\begin{aligned} f_{ij}^1 &= (a_4 + (a_8 - a_{10})\chi'')\,[u_n] + a_8[p_x] - b_2[p] - b_4[p_\xi] - b_6\,[p_\eta], \\ q_{ij} &= c_4[u_\xi] + c_6[u_\eta] + d_4[v_\xi] + d_6[v_\eta]. \end{aligned} \tag{4.53}$$

We can determine f_{ij}^2 in the same way.

Remark: Tu and Peskin [51] used the discrete delta function approach to find the right hand sides f_{ij}^1 and f_{ij}^2. In their approach $q_{ij} = 0$. So their method smooths the solution and adds an artificial source term $-q_{ij}$ to the continuity equation. This could explain the leaking phenomena in their approach.

4.4.2 Fast Fourier transform

We consider a grid function ϕ and define the discrete Fourier transformation of ϕ as

$$\hat{\phi}_{k_1,k_2} = \frac{1}{N} \sum_{i_1,j_1=0}^{N-1} e^{-i(2\pi/N)(j_1 k_1 + j_2 k_2)}\, \phi_{j_1,j_2}, \quad 0 \le k_1,\ k_2 \le N-1. \tag{4.54}$$

The inverse of Fourier transformation can be obtained with almost the formula above except the plus sign in the exponential. According to this definition, the discrete Fourier transforms of the Stokes equations (4.42) are as follows:

$$-\frac{i}{h}\sin\frac{2\pi k_1}{N}\,\hat{p}_{k_1,k_2} - \frac{4}{h^2}\left(\sin^2\frac{\pi k_1}{N} + \sin^2\frac{\pi k_2}{N}\right)\hat{u}_{k_1,k_2} = \hat{f}^1_{k_1,k_2}, \tag{4.55}$$

$$-\frac{i}{h}\sin\frac{2\pi k_2}{N}\,\hat{p}_{k_1,k_2} - \frac{4}{h^2}\left(\sin^2\frac{\pi k_1}{N} + \sin^2\frac{\pi k_2}{N}\right)\hat{v}_{k_1,k_2} = \hat{f}^2_{k_1,k_2}, \tag{4.56}$$

$$\frac{i}{h}\sin\frac{2\pi k_1}{N}\,\hat{u}_{k_1,k_2} + \frac{i}{h}\sin\frac{2\pi k_2}{N}\,\hat{v}_{k_1,k_2} = \hat{q}_{k_1,k_2}. \tag{4.57}$$

The solution of this algebraic equation is

$$\hat{p}_{k_1,k_2} = \frac{hi\left(\sin(2\pi k_1/N)\,\hat{f}^1_{k_1,k_2} + \sin(2\pi k_2/N)\,\hat{f}^2_{k_1,k_2}\right)}{\sin^2(2\pi k_1/N) + \sin^2(2\pi k_2/N)} + \frac{4\left(\sin^2(\pi k_1/N) + \sin^2(\pi k_2/N)\right)\hat{q}_{k_1,k_2}}{\sin^2(2\pi k_1/N) + \sin^2(2\pi k_2/N)}, \tag{4.58}$$

$$\hat{u}_{k_1,k_2} = \frac{-hi\,\sin(2\pi k_1/N)\,\hat{p}_{k_1,k_2} - \hat{f}^1_{k_1,k_2}}{4\left(\sin^2(\pi k_1/N) + \sin^2(\pi k_2/N)\right)}, \tag{4.59}$$

$$\hat{v}_{k_1,k_2} = \frac{-hi\,\sin(2\pi k_2/N)\,\hat{p}_{k_1,k_2} - \hat{f}^2_{k_1,k_2}}{4\left(\sin^2(\pi k_1/N) + \sin^2(\pi k_2/N)\right)}. \tag{4.60}$$

Notice that certain values of k_1 and k_2 will cause difficulty in (4.58)–(4.60). When $(k_1, k_2) = (0,0)$, $(0, N/2)$, $(N/2, 0)$, and $(N/2, N/2)$, we should have

$$\hat{f}^1_{0,0} = \hat{f}^2_{0,0} = \hat{q}_{0,0} = 0, \tag{4.61}$$

$$\hat{q}_{0,N/2} = \hat{q}_{N/2,0} = \hat{q}_{N/2,N/2} = 0 \tag{4.62}$$

so that the linear system of equations is consistent. Due to the discretization error for the Stokes equation, which is $O(h^2)$ away from the interface and $O(h)$ near the interface, the quantities in (4.61) and (4.62) will have magnitude $O(h)$ by the Parsevel theorem. If we set those quantities in (4.61) and (4.62) to be zero, this is equivalent to perturbing the right hand side of equation (4.55)–(4.57) by magnitude of $O(h^2)$. We anticipate the solution of the perturbed system will agree with the solution of (4.55)–(4.57) to $O(h^2)$.

4.4.3 Analysis of the approach

We have implemented the method described above. The area is well preserved and it seems to be better than first order but not quite second order accurate, yielding better results than Tu and Peskin's method but not as good as the three Poisson approach discussed in the next section.

It is certainly not desirable to solve the linear system of equations (4.42) directly for p, u, and v simultaneously. The coefficient matrix is singular and very large ($3N^2 \times 3N^2$), and it is not clear what the best iterative method would be. But if we use the FFT technique we can not determine $\hat{p}_{k_1,k_2}$ for certain k_1 and k_2, so we are unable to recover the pressure.

Another big issue is that of accuracy. When we discretized the Stokes equation, we made the local truncation error to be $O(h)$ near the interface and $O(h^2)$ away from the interface. This is also true for the perturbed system. However, we do not know how well the solution of the linear system (4.55)–(4.57) approximates the exact solution of the Stokes equations.

Even for smooth solutions, it is known that the pressure obtained from the discrete Stokes equation is only $O(h)$ accurate. The error in the pressure will certainly affect the accuracy of u and v. We construct an example for a steady state Stokes system with a discontinuous source term below to see how well our method works and how the error behaves.

Example 4.1 Consider the following functions defined on $[-1,1]\times[-1,1]$:

$$u = \begin{cases} x^3+y^3 & \text{if } x^2/a^2+y^2/b^2 \le 1 \\ 0 & \text{otherwise,} \end{cases} \tag{4.63}$$

$$v = \begin{cases} x^3+\sin y & \text{if } x^2/a^2+y^2/b^2 \le 1 \\ 0 & \text{otherwise,} \end{cases} \tag{4.64}$$

$$p = \begin{cases} 3x^2+6yx+\cos y & \text{if } x^2/a^2+y^2/b^2 \le 1 \\ 0 & \text{otherwise.} \end{cases} \tag{4.65}$$

They satisfy the " Stokes equation "

$$\begin{aligned} \Delta u - p_x &= 0, \\ \Delta v - p_y &= 0, \\ u_x + v_y &= \begin{cases} 3x^2+\cos y & \text{if } x^2/a^2+y^2/b^2 \le 1 \\ 0 & \text{otherwise.} \end{cases} \end{aligned} \tag{4.66}$$

This example is more complicated than our model problem because u and v are also discontinuous. In addition there is a discontinuous source term in the continuity equation. Using the immersed interface method we can still get f^1_{ij}, f^2_{ij}, and q_{ij}. In the derivation, we need to add more terms related to the jumps $[u]$, $[v]$, and the jump in the source term. We omit the details and just present the results below. Table 4.1 shows the local truncation error for (4.42), where $\| T^k_n \|_\infty$, $k=1,2,3$ are the truncation errors of the three equations in (4.42), and r_k is defined as the ratio $\|T^k_n\|/\|T^k_{2n}\|$, for $k=1,2,3$.

Table 4.1: The truncation errors in the direct method for Stokes flow.

n	$\| T^1_n \|_\infty$	r_1	$\| T^2_n \|_\infty$	r_2	$\| T^3_n \|_\infty$	r_3
20	0.167		0.138		7.84×10^{-2}	
40	0.100	1.6657	0.130	1.0606	4.99×10^{-2}	1.4259
80	5.71×10^{-2}	1.7513	6.88×10^{-2}	1.8884	5.92×10^{-2}	1.6853
160	2.87×10^{-2}	1.9855	3.88×10^{-2}	1.7709	1.59×10^{-2}	2.0400
320	1.47×10^{-2}	1.9531	1.97×10^{-2}	1.9730	7.91×10^{-3}	2.0198

Table 4.2 gives the error of p, u, and v in spectral space. For the transformed $\hat{p}$, we exclude those points where $\hat{p}_{k_1,k_2}$ is undefined.

Table 4.3 shows the error between the computed velocity and the exact solution. We observe second order accuracy. Since we cannot get p from the FFT method, we did not compare the error in the pressure.

4.5 Three-Poisson problem approach

In this section, we describe how to solve the Stokes equation using the three Poisson equations (4.9)–(4.11). The approach seems to be faster than the direct method and also more

Table 4.2: The error of the direct method for Stokes flow in the spectral space.

n	$\| \hat{p}_n - \hat{p}_e \|_\infty$	r_1	$\| \hat{u}_n - \hat{u}_e \|_\infty$	r_2	$\| \hat{v}_n - \hat{v}_e \|_\infty$	r_3
20	6.94×10^{-2}		7.31×10^{-4}		5.31×10^{-4}	
40	4.99×10^{-2}	1.3904	1.99×10^{-4}	3.6698	1.59×10^{-4}	3.3242
80	2.54×10^{-2}	1.9656	5.02×10^{-5}	3.9678	4.18×10^{-5}	3.8242
160	1.61×10^{-2}	1.5731	1.22×10^{-5}	4.0971	1.07×10^{-5}	3.9079
320	1.02×10^{-2}	1.5736	3.04×10^{-6}	4.0258	2.72×10^{-6}	3.9278

Table 4.3: The error in velocity of the direct method for Stokes flow.

n	$\| u - u_e \|_\infty$	r_1	$\| v - v_e \|_\infty$	r_3
20	7.35×10^{-3}		5.12×10^{-3}	
40	2.24×10^{-3}	3.2736	1.97×10^{-3}	2.5946
80	5.92×10^{-4}	3.7938	5.20×10^{-4}	3.7951
160	1.58×10^{-4}	3.7395	1.33×10^{-4}	3.8868
320	4.15×10^{-5}	3.8120	3.41×10^{-5}	3.9220

accurate. Moreover we can also obtain the pressure once we specify its value at one particular point.

With the known jumps $[p]$ and $[p_n]$, we can use the immersed interface method to get the discrete equation for the pressure

$$\frac{p_{i-1,j} + p_{i+1,j} + p_{i,j-1} + p_{i,j+1} - 4p_{ij}}{h^2} = C_{ij}. \tag{4.67}$$

Now we can still use the classic five-point stencil of the central difference formula and only need to add correction terms at irregular grid points. The condition for the linear system to have solutions is the constraint

$$S_c = \sum_{i,j=0}^{N-1} C_{ij} = 0. \tag{4.68}$$

But for the immersed interface method, C_{ij} usually does not satisfy (4.68) because the local truncation error is order h at irregular grid points. So S_c will be order $O(1)$. If we perturb C_{ij} by a constant so that the constraint does hold, i.e., if we define

$$\hat{C}_{ij} = C_{ij} - \frac{S_c}{N^2} \tag{4.69}$$

and replace C_{ij} by $\hat{C}_{ij}$, then the solutions to the perturbed (4.67) exist and is the least squares solution to the unperturbed equation (4.67) (see [49]). Notice that $S_c/N^2 = h^2 S_c$ is of order h^2. This means the order of the local truncation errors, which are $O(h^2)$ away from

the interface and $O(h)$ near the interface, have not been changed. We expect the solution to the perturbed equation (4.67) will approximate the true solution of the Poisson equation to second order accuracy.

We use the Fourier method described in [49] to solve the perturbed equation (4.67) and set $p_{00} = 0$ to get a particular solution.

Once the pressure is computed, we can solve for the components of the velocity u and v through (4.10) and (4.11). But we have to determine the values of p_x and p_y first at all grid points using the known p_{ij} before we can use the Fourier method again to solve the two Poisson equations. This is not an easy job because p is discontinuous. Below we show how to interpolate p_{ij} to get $(p_x)_{ij}$ at all grid points. Since p_x appears on the right hand side of the Poisson equation for u, we only need to approximate it to first order at irregular grid points.

If (i, j) is a regular grid point, then we simply use the central difference

$$(p_x)_{ij} = \frac{p_{i+1,j} - p_{i-1,j}}{2h}. \tag{4.70}$$

If (i, j) is an irregular grid point, we distinguish the following different cases.

Case (i): The grid points (i, j) and $(i-1, j)$ are located on the same side of the interface. Then

$$(p_x)_{ij} = \frac{p_{ij} - p_{i-1,j}}{h}. \tag{4.71}$$

Case (ii): The grid points (i, j) and $(i+1, j)$ are located on the same side of the interface. Then

$$(p_x)_{ij} = \frac{p_{i+1,j} - p_{ij}}{h}. \tag{4.72}$$

Case (iii): The grid points $(i + 1, j)$ and $(i - 1, j)$ are located on the same side of the interface, but (i, j) is on the other side. In this case we interpolate $p_{i-1,j}$, p_{ij} and $p_{i+1,j}$ to get $(p_x)_{ij}$ to first order by using the known jump conditions $[p]$, $[p_x]$, and $[p_y]$ from (4.15), (4.30), and (4.31).

Let (X_k^*, Y_k^*) be the control point closest to (x_i, y_j), and x_l, ($l = i + 1$, or $i - 1$) be the one of x_i and x_{i+1} which is closer to X_k^*. Then we can use the following interpolation formula:

$$(p_x^{\pm})_{ij} = \frac{p_{ij} - p_{lj} \mp [p] \mp [p_x]\ (x_l - X_k^*) \mp [p_y]\ (y_j - Y_k^*)}{(x_i - x_l)}, \tag{4.73}$$

where the sign in the expression depends on which side of the interface the point (i, j) is on, and $[p]$, $[p_x]$ and $[p_y]$ are calculated at (X_k^*, Y_k^*). It is easy to prove that the approximation of (4.73) is first order accurate. We give a proof for the case when (i, j) is on the $-$ side, or inside of the interface. This means that $(i - 1, j)$ and $(i + 1, j)$ are on $+$ side. We expand

$$T = \frac{p(x_i, y_j) - p(x_l, y_j) + [p] + [p_x]\ (x_l - X_k^*) + [p_y]\ (y_j - Y_k^*)}{(x_i - x_l)} \tag{4.74}$$

about (X_k^*, Y_k^*) from each side of the interface to get

$$\begin{aligned}(x_i - x_l)T &= p^-(X_k^*, Y_k^*) + p_x^-(X_k^*, Y_k^*)\,(x_i - X_k^*) + p_y^-(X_k^*, Y_k^*)\,(y_j - Y_k^*) \\ &\quad - p^+(X_k^*, Y_k^*) - p_x^+(X_k^*, Y_k^*)\,(x_i - X_k^*) - p_y^+(X_k^*, Y_k^*)\,(y_j - Y_k^*) \\ &\quad + [p] + [p_x]\,(x_l - X_k^*) + [p_y]\,(y_j - Y_k^*) + O(h^2) \qquad (4.75) \\ &= p_x^-(X_k^*, Y_k^*)\,(x_i - x_l) + O(h^2) \\ &= p_x^-(x_i, y_j)(x_i - x_l) + O(h^2).\end{aligned}$$

Therefore

$$T = p_x^-(x_i, y_j) + O(h). \qquad (4.76)$$

Now we have enough information to use the immersed interface method described in Chapter 2 to get the discrete form for u. To solve the linear system uniquely for u_{ij}, we set $u_{00} = 0$ and use the Fourier method to the perturbed system.

We obtain the component of the velocity in the y direction in almost the same way by replacing the x-related quantities by the y-related ones.

4.6 Moving the interface

With the information of u_{ij} and v_{ij}, we now need to use the ODE

$$\frac{\partial \vec{X}}{\partial t} = \vec{u}(X(s), Y(s), t), \qquad (4.77)$$

to move the interface to its position at the next time step. Possible discrete forms are

$$\begin{aligned}\vec{X}^{n+1} &= \vec{X}^n + \Delta t\, \vec{U}^n \qquad \text{(explicit)}, \\ \vec{X}^{n+1} &= \vec{X}^n + \frac{\Delta t}{2}\left(\vec{U}^n + \vec{U}^{n+1}\right) \qquad \text{(implicit)}.\end{aligned}$$

Here $\vec{U}^n$ is the approximation to $\vec{u}(X(s), Y(s), t^n)$, the velocity of the interface which is the same as the velocity of the fluid in contact with it.

Whether we use the explicit or implicit method, we need to interpolate the grid functions u_{ij} and v_{ij} to get the velocity of the interface. In this section we will omit the time index n in u_{ij}^n and v_{ij}^n for simplicity. More precisely, we need to find U_k and V_k to second order accuracy at all control points (X_k, Y_k). Note that the velocity is not smooth across the interface and we need to use the jumps $[u_x]$, $[u_y]$, $[v_x]$ and $[v_y]$ in the interpolation process.

Taking a typical point (X, Y) on the interface, we show how to interpolate u_{ij} to get the x-component U of the velocity at (X, Y).

First we choose the first three grid points (x_{i1}, y_{j1}), (x_{i2}, y_{j2}), and (x_{i3}, y_{j3}) closest to (X, Y).

Then we form a linear combination of the grid values at these points plus a correction term to approximate U

$$U = \gamma_1\, u_{i1,j1} + \gamma_2\, u_{i2,j2} + \gamma_3\, u_{i3,j3} - \text{Correction}. \qquad (4.78)$$

We use Taylor expansion about (X, Y) to get the equations for the coefficients γ's so that we have a second order approximation:

$$\begin{aligned} U &= \gamma_1 u(x_{i1}, y_{j1}) + \gamma_2 u(x_{i2}, y_{j2}) + \gamma_3 u(x_{i3}, y_{j3}) - \text{Correction} \\ &= a_1 u^- + a_2 u^+ + a_3 u_x^- + a_4 u_x^+ + a_5 u_y^- + a_6 u_y^+ - \text{Correction} + O(h^2) \\ &= (a_1 + a_2) u^- + (a_3 + a_4) u_x^- + (a_5 + a_6) u_y^- + a_2 [u] \\ &\quad + a_4 [u_x] + a_6 [u_y] - \text{Correction} + O(h^2), \end{aligned}$$

where a_i has a similar meaning as in (2.55), and all the quantities are calculated at (X, Y). So we set

$$\begin{aligned} a_1 + a_2 &= 1, \\ a_3 + a_4 &= 0, \\ a_5 + a_6 &= 0. \end{aligned}$$

These are equivalent to

$$\begin{aligned} \gamma_1 + \gamma_2 + \gamma_3 &= 1, \\ \gamma_1 (x_{i1} - X) + \gamma_2 (x_{i2} - X) + \gamma_3 (x_{i3} - X) &= 0, \\ \gamma_1 (y_{j1} - Y) + \gamma_2 (y_{j2} - Y) + \gamma_3 (y_{j3} - Y) &= 0. \end{aligned}$$

The solutions to this linear system are

$$\begin{aligned} \gamma_2 &= \frac{(y_{j1} - Y)(x_{i3} - x_{i1}) - (x_{i1} - X)(y_{j3} - y_{j1})}{(x_{i2} - x_{i1})(y_{j3} - y_{j1}) - (x_{i3} - x_{i2})(y_{j2} - y_{j1})}, \\ \gamma_3 &= \frac{(y_{j1} - y_{j1})(x_{i1} - X) - (x_{i2} - x_{i1})(y_{j1} - Y)}{(x_{i2} - x_{i1})(y_{j3} - y_{j1}) - (x_{i3} - x_{i2})(y_{j2} - y_{j1})}, \\ \gamma_1 &= -(\gamma_2 + \gamma_3). \end{aligned} \tag{4.79}$$

Once we get the coefficients γ_1, γ_2, and γ_3, we are able to compute the correction term which is the sum of $a_2 [u]$, $a_4 [u_x]$, and $a_6 [u_y]$. We can use the same coefficients to find the y-component V of the velocity at (X, Y).

4.7 The Implicit method

One could consider using explicit (*e.g.* forward Euler), approximate implicit (see [43] [51]), or implicit (*e.g.* backward Euler or Trapezoidal) methods to update the interface $\vec{X}$.

In the explicit approach, the boundary force and the velocity are computed from the configuration at the beginning of the time step. Usually the differential equation (4.77) is stiff (see Fig. 4.7, Fig. 4.10). So there is a strict restriction on the time step for the explicit method.

In the approximate implicit approach described in [51], the force density is computed from an estimate of the interface configuration at the end of the time step

$$\begin{aligned} \vec{X}^{n+1,*} &= \vec{X}^n + \Delta t \, \vec{U}_k^* \\ &= \vec{X}^n + \Delta t \, \lambda \, \vec{f} \left(\vec{X}^{n+1,*} \right), \end{aligned} \tag{4.80}$$

where λ is an approximation to the magnitude of the velocity induced at a point by a unit force applied at that point ignoring the effect of force at other boundary points. The boundary force $\vec{f}\left(\vec{X}^{n+1,*}\right)$ could be a function of t as well, but for simplicity, we write it as $\vec{f}\left(\vec{X}^{n+1,*}\right)$. The derivation of (4.80) can be found in [41]. Instead of solving (4.80) for $\vec{X}^{n+1,*}$ directly, we can form a nonlinear equation for $\vec{f}\left(\vec{X}^{n+1,*}\right)$ using the energy function defined as

$$\vec{f}\left(\vec{X}\right) = -\nabla E(\vec{X}) = -\nabla E\left[\ \vec{X}^n + \Delta t\,\lambda\,\vec{f}\left(\vec{X}^{n+1,*}\right)\right].$$

Treating $\vec{f}\left(\vec{X}\right)$ as a new independent variable, we then get the nonlinear system for $\vec{f}\left(\vec{X}\right)$,

$$\vec{f} + \nabla E\left(\vec{X}^n + \Delta t\,\lambda\,\vec{f}\right) = 0.$$

The Newton iterative method can be written as follows

$$\left(I + \Delta t\,\lambda H_E\left(\vec{X}^n + \Delta t\,\lambda\vec{f}^m\right)\right)\left(\vec{f}^{m+1} - \vec{f}^m\right) = \\ -\left(\vec{f}^m + \nabla E\left(\vec{X}^n + \Delta t\,\lambda\vec{f}^m\right)\right), \tag{4.81}$$

where H_E is the Hessian matrix of E. H_E is a periodic block-tridiagonal system (with 2×2 block). Once an approximate solution for the force is obtained, we can solve the Stokes equation to get the velocity of the interface and update the location explicitly. So the only difference between the explicit and approximate implicit methods is that the force is calculated implicitly in the approximate implicit approach.

In this approach, each iteration in solving the nonlinear system requires about $O(N^2\log N)$ multiplications if the FFT transformation is used to solve the Stokes flow. The total cost will depend on the number of iterations.

The approximate implicit method has better stability properties than an explicit method but not as good as the fully implicit method.

In the fully implicit approach, the boundary force is computed from the unknown configuration at the end of the time step. Tu and Peskin proposed an approach which uses the fundamental solution of the Stokes equations. Let G be the discrete Green's function of the Stokes equations on a periodic domain, so each entry of $G(i,j)$ is the solution of the Stokes equation with the force being a two dimensional delta function at (x_i, y_j). So G is an $N\times N$ array with 2×2 matrix entries. The matrix G needs to be calculated just once. Define

$$\tilde{G}_{kl} = h^4\sum_{i,j=0}^{N-1}\sum_{i',j'=0}^{N-1} G\left(i-i', j-j'\right)\,\delta_h\left(\vec{x}_{ij} - \vec{X}_k^n\right)\left(\delta_h\vec{x}_{i'j'} - \vec{X}_l^n\right), \tag{4.82}$$

where δ_h is the two dimensional discrete cos-delta function (1.8). Then the system for the new location $\vec{X}^{n+1}$ is

$$\vec{X}^{n+1} = \vec{X}^n + \Delta t\,\Delta s\,\tilde{G}\,\vec{f}\left(\vec{X}^{n+1}\right), \tag{4.83}$$

see [51] for the proof. Similar to the approximate implicit method, using the energy function again we get a nonlinear system for $\vec{f}\left(\vec{X}^{n+1}\right)$

$$\tilde{G}\,\vec{f} + \tilde{G}\,\nabla E\left(\vec{X}^n + \Delta t\,\Delta s\,\tilde{G}\,\vec{f}\right) = 0. \tag{4.84}$$

The Newton iterative formula now is

$$\left[\tilde{G} + \Delta t\, \Delta s\, \tilde{G}\, H_E\ \left(\vec{X}^n + \Delta t\, \Delta s\, \tilde{G}\, \vec{f}^m\right) \tilde{G}\right]\ \left(\vec{f}^{m+1} - \vec{f}^m\right) = \\ -\left(\tilde{G}\, \vec{f}^m + \tilde{G}\, \nabla E\left(\vec{X}^n + \Delta t\, \Delta s\, \tilde{G}\, \vec{f}^m\right)\right). \qquad (4.85)$$

Once an approximate solution is computed, we get the new location using (4.83).

Now each iteration in solving the nonlinear system requires about $O(N_b^3) \sim O(N^3)$ operations, much more than the approximate implicit approach if $N_b \sim N$.

Using the discrete delta function approach, we need to take enough control points N_b to prevent the boundary from leaking. In other words, we need enough control points so that all grid points near the interface feel the effect of the boundary through the spreading of the discrete delta functions. So while it is stable, the fully implicit method described above is very expensive.

By contrast in the IIM (immersed interface method) approach the boundary can be represented by relatively few control points. Cubic splines passing through these points are used to define the boundary and jump conditions at any point needed.

Neither of the approximate implicit or the fully implicit methods discussed above solve the Stokes equations for p, u and v in the process of finding the solution of the nonlinear equations for the boundary force at the end of time step. However, to form the nonlinear equations can be expensive. In the approximate implicit approach, we need to have the estimate of $\lambda(\vec{X})$, $\nabla E\left(\vec{X}^n + \Delta t\, \lambda\, \vec{f}^m\right)$ and $H_E\left(\vec{X}^n + \Delta t\, \lambda\, \vec{f}^m\right)$. The last two terms change with the iteration for $\vec{f}^m$. For the fully implicit approach, we need $\tilde{G}$, $H_E\left(\vec{X}^n + \Delta t\, \Delta s\, \tilde{G}\, \vec{f}^m\right)$, and $\nabla E\left(\vec{X}^n + \Delta t\, \Delta s\, \tilde{G}\, \vec{f}^m\right)$. $\tilde{G}$ needs to be updated at every time step and the other two terms need to be changed in every iteration for the force $\vec{f}^m$. Also these two methods are based on the forward and backward Euler's formula, hence they are only first order accurate. They are not suitable for the immersed interface method because it will damage the second order accuracy. We naturally want to use the second order accurate Crank-Nicolson implicit method.

4.7.1 A second order-implicit method using rank-2 updating

With the IIM, we use the trapezoidal formula to update the interface,

$$\vec{X}^{n+1} = \vec{X}^n + \frac{\Delta t}{2}\left[\vec{U}^n + \vec{\mathcal{U}}\left(\vec{X}^{n+1}\right)\right], \qquad (4.86)$$

where $\vec{\mathcal{U}}\left(\vec{X}\right)$ is defined to be the velocity of the interface of the location $\vec{X}$. We try to solve this nonlinear equation directly using the quasi-Newton method. Define

$$f(\vec{X}) = \vec{X} - \vec{X}^n - \frac{\Delta t}{2}\left[\vec{U}^n + \vec{\mathcal{U}}\left(\vec{X}\right)\right]. \qquad (4.87)$$

If we solve the nonlinear system $f(\vec{X}) = 0$ of dimension $2N_b$, we can update the interface at the end of a time step.

Certainly, it would be costly to compute the Jacobian matrix $J(t)$ at every time step and every iteration for the nonlinear system. For example, to compute a column of the

Jacobian matrix involves solving three Poisson problems. However the Jacobian matrix $J(t)$ is close to the identity matrix, $J(t) = I + O(\Delta t)$, and is also continuous in time, so $J(t) \approx J(t - \Delta t)$. This suggests that we use a quasi-Newton method in which we maintain an approximation B to J^{-1} and update this approximation in each time step. The quasi-Newton method is super-linearly convergent even for a poorly approximated Jacobian. So we can use various low-rank-updating techniques which usually only require the computation of inner products without affecting the convergence speed. To go from $\vec{X}^n$ to get $\vec{X}^{n+1}$, the following BFGS (Broyden-Fletcher-Goldfarb-Shanno) method [29], [47] for solving the nonlinear system seems to be very efficient.

$$
\begin{aligned}
&\text{For } \ m = 0, 1, 2, \cdots, m^*, \\
&\qquad (m^*, \text{ the step when the iterative method converges }) \\
&\vec{X}_0^{n+1} = \vec{X}_{m^*}^n, \quad B_0^n = B_{m^*}^{n-1}, \\
&\vec{X}_{m+1}^{n+1} = \vec{X}_m^{n+1} - B_m^n f\left(\vec{X}_m^{n+1}\right), \qquad (4.88) \\
&B_{m+1}^n = B_m^n + \frac{\mu_m s_m s_m^T - s_m y_m^T B_m^n - B_m^n y_m s_m^T}{s_m^T y_m}, \qquad (4.89)
\end{aligned}
$$

where

$$
\begin{aligned}
s_m &= -B_m^n f\left(\vec{X}_m^{n+1}\right), \\
y_m &= f\left(\vec{X}_{m+1}^{n+1}\right) - f\left(\vec{X}_m^{n+1}\right), \qquad (4.90) \\
\mu_m &= 1 + \frac{y_m^T B_m^n y_m}{s_m^T y_m},
\end{aligned}
$$

and we have omitted the time index n for s_m, μ_m and y_m. At the initial time step $t = 0$, we take $B_0^0 = I$. This is reasonable since $J = I + O(\Delta t)$.

In our implicit approach, each iteration for solving the nonlinear equation calls the Stokes solver, which requires 3 Poisson solvers and hence $O(N^2 log(N))$ operations if we use the FFT method. In addition we need a couple of inner products in order to compute $B_m^n f\left(\vec{X}_m^{n+1}\right)$. The total computational cost per iteration is $O(N^2 log(N))$. Usually we only need about 2 or 3 iterations per time step.

4.8 Some typical numerical examples for Stokes flow

Now we present some typical numerical examples to see how well our method works and compare our method with Tu and Peskin's approach.

Example 4.2 This example is taken from Tu and Peskin [51]. The initial interface (the solid line in Fig. 4.2) is an ellipse with major and minor axes $a = 0.75, b = 0.5$, respectively. The unstretched interface (the dash-dot line in Fig. 4.2) is a circle with radius $r = 0.5$. Due to the restoring force, the ellipse will converge to an equilibrium circle (the dashed line in Fig. 4.2) with radius $r = \sqrt{ab}$; this is larger than the unstretched interface because of the incompressibility of the enclosed fluid. So the interface is still stretched at the equilibrium state, and the non-zero boundary force is balanced by a nonzero jump in the pressure (Fig. 4.3).

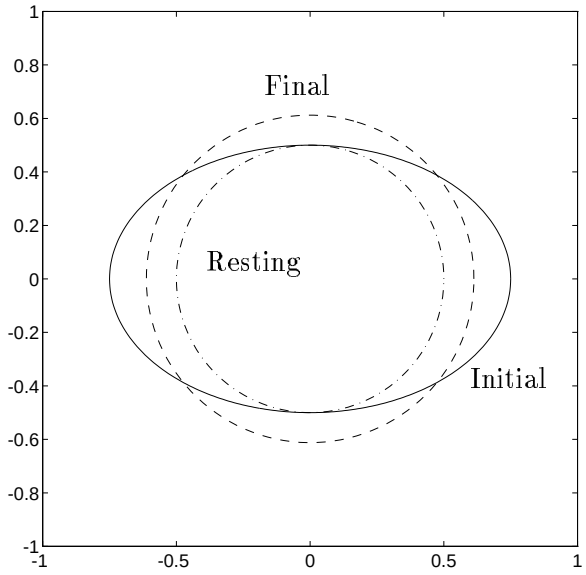


Figure 4.2: The interface at different states: Initial interface (solid line), the ellipse with $a = 0.75$, $b = 0.5$. Equilibrium position (dashed line), the circle with $r = \sqrt{ab} = 0.6123\cdots$. The resting circle (dash-dot line), the circle with $r = 0.5$.

We can not compare the accuracy in the pressure since this is not available in Tu & Peskin's method. For the velocity u (Fig. 4.4), and v, if we compare a slice of the x-component of the velocity as shown in Fig. 4.5), we can see that their approach smooths the velocity profile in the neighborhood of the interface, as is expected with the discrete delta function spreading of forces.

Now let us compare the accuracy of p, u and v at $t = 0$ as we did in Example 4.1. In this example, however, we do not know the exact solution. The conventional approach for estimating the convergence rate is the following:

Suppose a method is order q accurate is some norm. Let $\tilde{u}(h)$ be the approximation to the exact solution u_{exact} in a problem, obtained by using the method with some grid size h. Then we can write

$$\tilde{u}(h) = u_{exact} + C\, h^q + o(h^q), \tag{4.91}$$

where C is a constant. Let h^* be the finest grid used for the method; then on a coarser grid with the step size h, we have

$$\frac{\tilde{u}(h) - \tilde{u}(h^*)}{\tilde{u}(h/2) - \tilde{u}(h^*)} \approx \frac{2^q\,(1 - (h^*/h)^q)}{1 - (2h^*/h)^q}. \tag{4.92}$$

From this ratio, we can estimate the order of accuracy.

For example, if we double the number of grid points successively, *i.e.*,

$$\frac{h^*}{h} = 2^{-k}, \qquad k = 1, 2, \cdots,$$

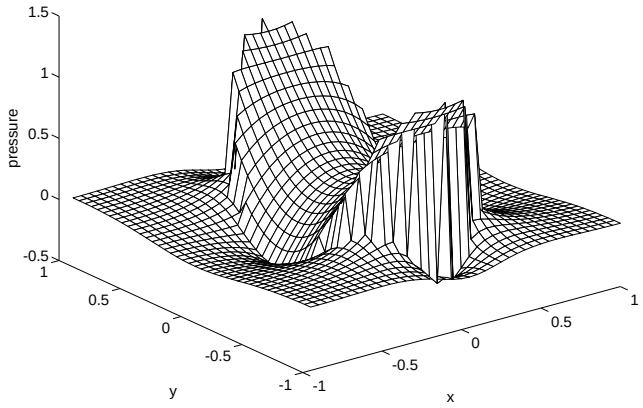


Figure 4.3: The computed pressure distribution of the Stokes flow at $t = 0$. The pressure is discontinuous.

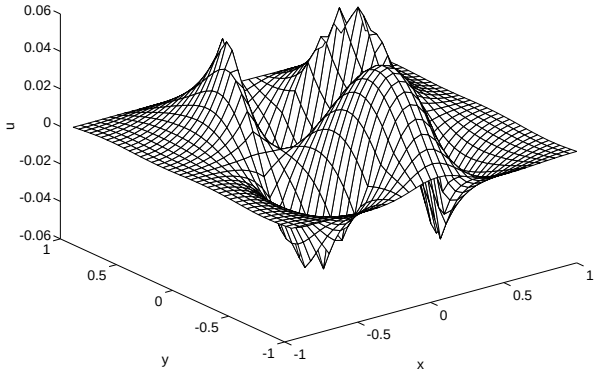


Figure 4.4: The x-component of velocity u in the Stokes flow at $t = 0$. It is continuous but not smooth across the interface.

(a) (b)

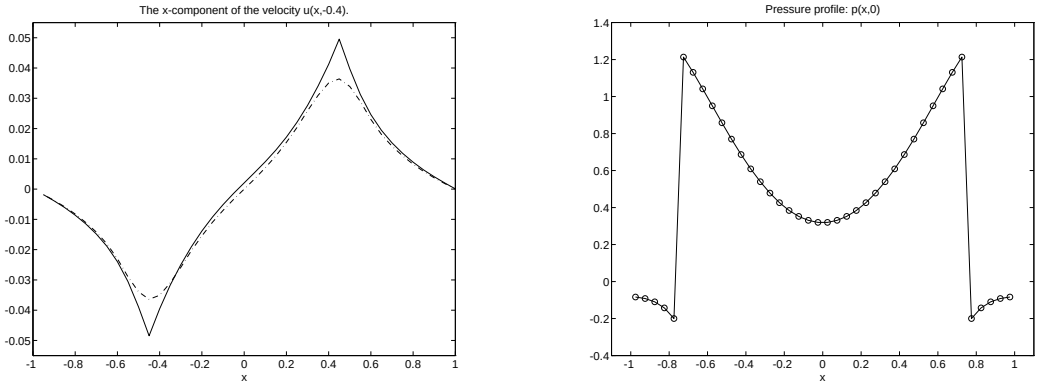


Figure 4.5: (a) A slice of u, the x-component of velocity in the Stokes flow at $t = 0$ and $y = -0.4$. It is continuous but u_x should be discontinuous across the interface. Solid line: IIM results, dot-dashed line: Results with method of Tu & Peskin. (b) A slice of pressure at $t = 0$ and $y = 0$, little 'o"s are the computed results with the IIM at the grid points. Note the large jump in pressure across the interface.

then the ratio in (4.92) is

$$\frac{\tilde{u}(h) - \tilde{u}(h^*)}{\tilde{u}(\frac{h}{2}) - \tilde{u}(h^*)} = \frac{2^q \left(1 - 2^{-kq}\right)}{1 - 2^{q(1-k)}}. \tag{4.93}$$

In particular, for a first order method ($q = 1$), this becomes

$$\frac{\tilde{u}(h) - \tilde{u}(h^*)}{\tilde{u}(\frac{h}{2}) - \tilde{u}(h^*)} = \frac{2 \left(1 - 2^{-k}\right)}{1 - 2^{1-k}} = \frac{2^k - 1}{2^{k-1} - 1}.$$

For $k = 2, 3, \cdots$, these ratios are

$$3, \quad \frac{7}{3} \simeq 2.333, \quad \frac{15}{7} \simeq 2.1429, \quad \frac{31}{15} \simeq 2.067, \cdots.$$

Similarly for a second order method ($q = 2$), (4.93) becomes

$$\frac{\tilde{u}(h) - \tilde{u}(h^*)}{\tilde{u}(\frac{h}{2}) - \tilde{u}(h^*)} = \frac{4 \left(1 - 4^{-k}\right)}{1 - 4^{1-k}} = \frac{4^k - 1}{4^{k-1} - 1}.$$

For $k = 2, 3, \cdots$, the ratios are

$$5, \quad \frac{63}{15} = 4.2, \quad \frac{255}{63} \simeq 4.0476, \quad \frac{1023}{255} \simeq 4.0118, \cdots.$$

Below we will denote the number of grid points in each direction in the uniform grid by N and the number of the control points on the interface as N_b.

Now we use the technique described above to estimate the accuracy of our immersed interface method and Tu and Peskin's approach for various quantities. Table 4.4 and

Table 4.4: The errors of computed p, u, and v at $t = 0$ using the IIM method via three Poisson equations. Second order accuracy can be observed.

N	$\| p_N - p_{320} \|_\infty$	r_1	$\| u_N - u_{320} \|_\infty$	r_2	$\| v_N - v_{320} \|_\infty$	r_3
40	1.9730×10^{-2}		2.6739×10^{-3}		5.0411×10^{-3}	
80	1.5416×10^{-3}	12.7986	6.3611×10^{-4}	4.2035	5.5415×10^{-4}	9.0969
160	2.6087×10^{-4}	5.9094	1.1161×10^{-4}	5.6996	1.0713×10^{-4}	5.1729

Table 4.5: The errors of computed u and v at $t = 0$ using Tu & Peskin's method. First order accuracy can be observed.

N	$\| u_N - u_{320} \|_\infty$	r_2	$\| v_N - v_{320} \|_\infty$	r_3
40	1.0170×10^{-2}		5.0540×10^{-3}	
80	4.4694×10^{-3}	2.2755	2.0512×10^{-3}	2.4639
160	1.5012×10^{-3}	2.9773	7.4032×10^{-4}	2.7707

Table 4.5 show the computed results using different grid sizes N with $N_b = N$ for the two different methods. In Tu & Peskin's approach, direct discretization and FFT (§4.4) are used, so the pressure is not available and is not listed in Table 4.5. As we expect, the immersed interface method via three Poisson equations exhibits second order convergence while Tu & Peskin's approach behaved as a first order method. In the Table 4.4 and Table 4.5, p_N, u_N and v_N are the grid functions approximating the pressure and the velocities in x- and y- directions, respectively, on an $N \times N$ grid. The ratios are defined as

$$r_1 = \frac{\| p_{2N} - p_{320} \|_\infty}{\| p_N - p_{320} \|_\infty}, \qquad r_2 = \frac{\| u_{2N} - u_{320} \|_\infty}{\| u_N - u_{320} \|_\infty}, \qquad r_2 = \frac{\| v_{2N} - v_{320} \|_\infty}{\| v_N - v_{320} \|_\infty}.$$

Determining the convergence rate for the moving interface is very difficult, especially in the infinity norm since the error in the location of the interface is not a monotonically decreasing function of the grid size h. It also depends on the relative position between the interface and the grid used. For example, let r_{max}, r_{min} be the longest and shortest distance of all control points on the interface from the origin. We use the results computed from the finest grid ($N = N_b = 320$) as the "exact solution" and compare the error for these two quantities with different grid sizes. We use the least squares method to get straight line fits to the data and use the slopes as the average convergence rate. Fig. 4.6 shows log – log plots of the error in our method and in Tu and Peskin's approach. For short time, say $t = 0.01$, the results are predictable. The errors go down as N increases. Our method not only has a faster convergence rate, but also smaller errors; see Fig. 4.6 (a). As time increases (*e.g.*, at $t = 1$), our method still converges quadratically. The behavior of Tu and Peskin's approach shown in Fig. 4.6 (b) is much more chaotic. The reason for this behavior is that if a point moves faster toward the equilibrium than it should, later on the restoring force will become smaller and smaller, so that the movement will slow down. In other words, there is an automatic adjustment mechanism for this type of moving interface method.

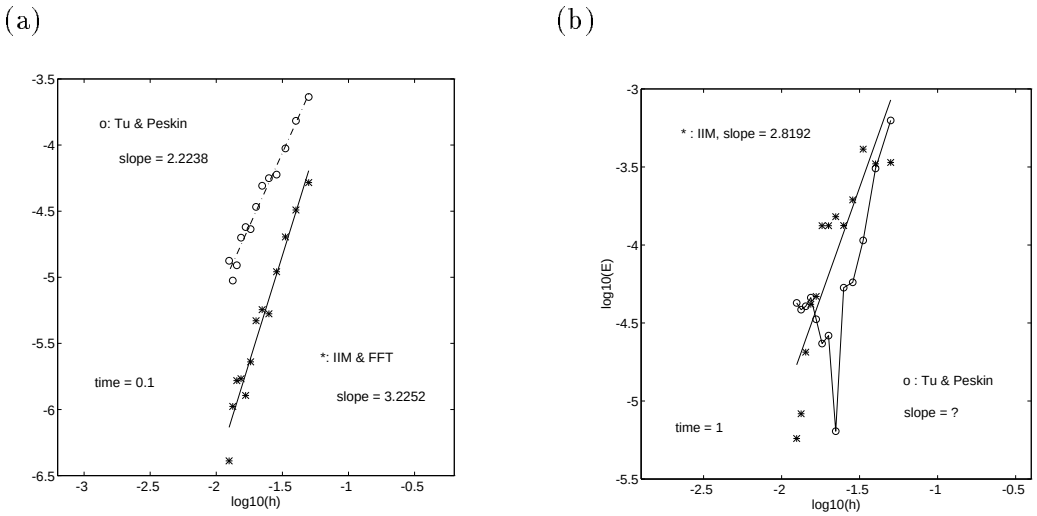


Figure 4.6: Comparison of the convergence rate for r_{max}. The solid line and the stars $(*)$ are the results of our method, the dash-dot line and 'o' are the results of Tu and Peskin's method. (a) At $t = 0.01$. (b) At $t = 1$.

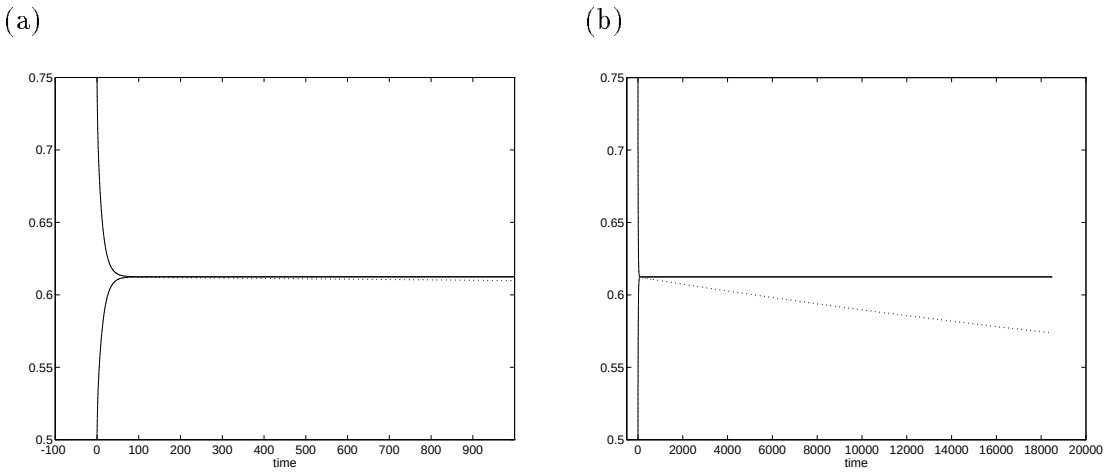


Figure 4.7: The longest (r_{max}) and shortest (r_{min}) distance of the control points from the origin as the function of time t on a 160x160 grid with $N_b = 160$. The solid line is computed using the immersed interface method via three Poisson equations. The dotted line is the result of Tu and Peskin's approach. (a) Blow up for short time $0 \leq t \leq 10^3$. Both methods show convergence to a circle, but slow leaking in the Tu & Peskin's method is visible. (b) Over a large time scale, it is seen that their leaking continues.

Therefore we have only been able to get a reasonable estimate for the convergence rate at the initial time and at large times when the interface reaches the equilibrium. However, as Peskin pointed out in [44], there is a systematic tendency in Tu and Peskin's method to lose volume slowly at a rate proportional to the pressure difference across the interface. This is the case for this test example since there is a jump in the pressure at the equilibrium state. So the results do not converge to the correct equilibrium using Tu and Peskin's method. Fig. 4.7 shows the change of r_{max} and r_{min} with the time. Theoretically they both converge to $\sqrt{ab} = 0.6123\cdots$. But because of the leaking, the result obtained with Tu and Peskin's method continuously shrinks beyond the equilibrium. Fig. 4.8 shows the change in the area with time. The immersed interface method preserves the area quite well while Tu and Peskin's approach continues to lose area slowly until it finally converges to the resting circle.

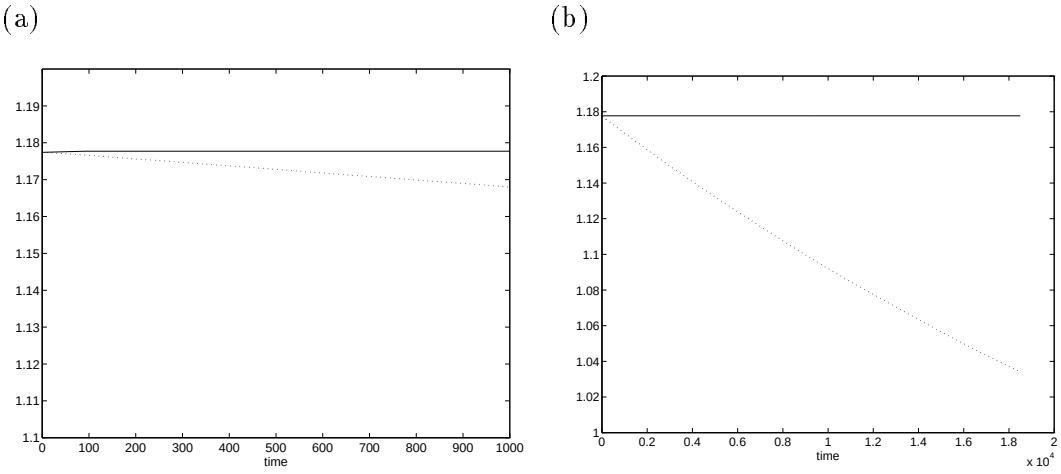


Figure 4.8: The computed area with the same notation as in Fig. 4.7. (a) Short time behavior, $0 \le t \le 10^3$. (b) Long time behavior, $0 \le t \le 2 \times 10^4$.

Above we only considered the error in r_{max}. We could also look at some norm of the error along the entire interface, for example the 2-norm. Let us take $N^* = N_b^*$ as the finest grid. For the coarser grid with $N \times N$, we take $N_b = N^*/l$, where $l = int(N^*/N)$. In this way we are guaranteed that each control point $\left(x_i^{(N)}, y_i^{(N)}\right)$, $i = 1, 2, \cdots, N_b$ on the interface is also a control point for the finest grid $N^* \times N^*$ and N_b^*. So it is possible to compute the error

$$e_N = \frac{1}{N_b} \sqrt{\sum_{i=1}^{N_b} \left(x_i^{(N)} - x_{i*l}^{(N^*)}\right)^2 + \left(y_i^{(N)} - y_{i*l}^{(N^*)}\right)^2}. \tag{4.94}$$

Our test results show that the error defined above is indeed a monotone decreasing function as N increases. In Fig. 4.9, we plot the global error at $t = 1$ with the finest grid being $N^* = 320$, and N and N_b being the pairs of $(40, 40)$, $(50, 40)$, $(60, 40)$, $(70, 40)$, $(80, 80)$, $(90, 80), \cdots$, $(150, 80)$, $(160, 160)$. Note that the number of control points N_b does not decrease smoothly with N.

Fig. 4.9 shows that our method converges with a smaller error and a faster convergence rate[3] than the method used by Tu and Peskin.

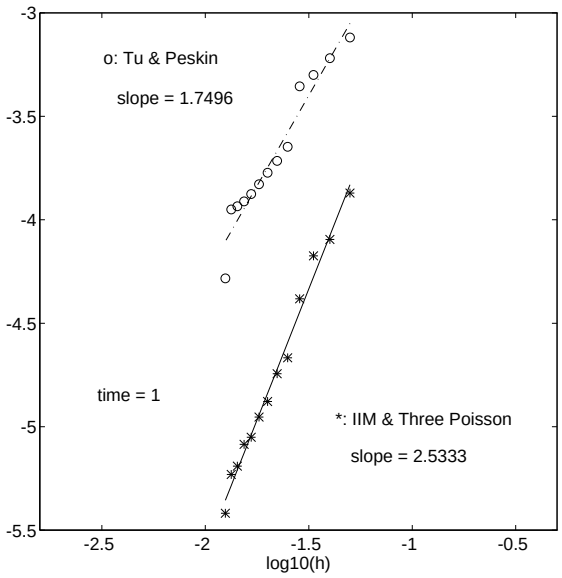


Figure 4.9: The global error at $t = 1$ in the 2-norm. Solid line and the star $(*)$ are the result computed with the immersed interface method via three Poisson equations. Dash-dot line and the small 'o"s are the results computed with Tu and Peskin's approach.

Another interesting phenomena we can observe from Fig. 4.9 is that the number of control points N_b on the interface plays an important role in Tu and Peskin's method. If we refine the space grid but keep the number of control points on the interface N_b fixed, say $N = 40, 50, \cdots, 70$ with $N_b = 40$; and then $N = 80, 90, \cdots, 150$ with $N_b = 80$; and finally $N = 160$ with $N_b = 160$ in Fig. 4.9, the errors obtained with their approach with N_b fixed will gradually cease to decline even if we refine the mesh grid because the error in expressing the interface will dominate. Then a refinement of the interface grid will lead to a relatively large fall in the error as we can see in Fig. 4.9. There is sharp drop in the error in Tu and Peskin's approach when N_b changes from 40 to 80 and from 80 to 160. So we should refine the grid for the domain and the interface simultaneously if we use their approach. However in our approach, as we mentioned in § 1.4.4, we can take fewer control points on the interface with little effect on the accuracy with our method, as we can see from Fig. 4.9, where we have the same structure mentioned above but no big jump in the error.

Now let us change the example slightly and let the radius of the resting circle be exactly the same as that of the equilibrium circle. There should be neither force nor a jump in the pressure across the interface when the equilibrium is reached. So both methods can reach the equilibrium state eventually. However before reaching the equilibrium state, the force will be very small and both methods will stop converging when the error in the

[3] Since we use the results obtained from the finest grid as the "exact solution", the slopes or ratios are greater than their actual values.

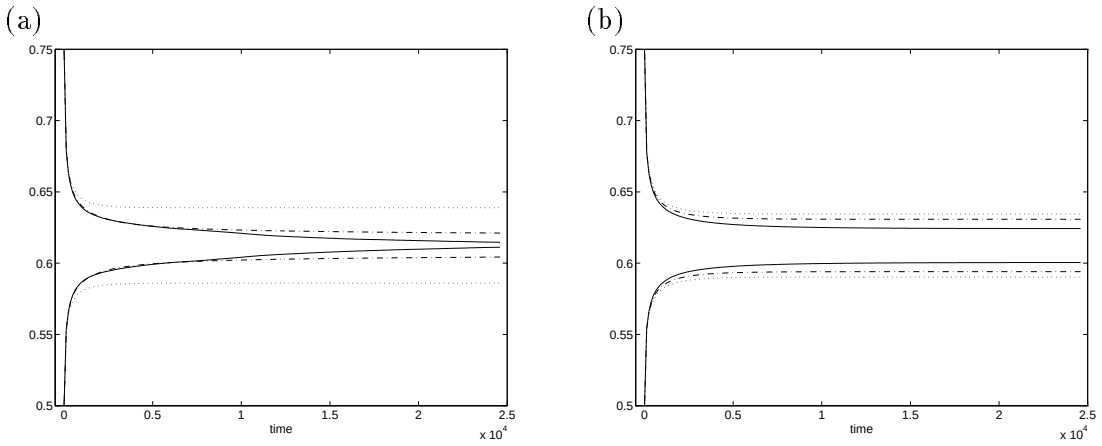


Figure 4.10: Comparison of the convergence rate for the case when the equilibrium state is the same as the resting circle. Solid line, dash-dot line and dotted line are the results obtained with 160×160, 80×80 and 40×40 grids respectively. (a) Immersed interface method via three Poisson equations. (b) Tu and Peskin's method.

discretization dominates the force. We see in Fig. 4.10 that the result obtained from Tu and Peskin's method is much less accurate than that obtained from our method.

Example 4.3 This example shows we can handle more complicated regions. The initial interface in polar coordinates is $\rho = 0.6 + 0.3 \sin 8\theta$. The unstretched interface is the circle with the radius $r = 0.3$. Because the interface is complicated, we take $N_b = 160$ so that we can express accurately. The problem is very stiff and we need to take the time step to be reasonably small. We compared our method with Tu & Peskin's approach and it revealed a similar behavior as the first example. So we will not give detailed numerical results but instead present the change in the interface in Fig. 4.11.

In this chapter we mainly discussed the case when the force is elastic. With slight modifications to the force calculation, the approach described in this chapter can apply to different boundary forces. For example, if the boundary force is due to surface tension at the interface between two different fluids, then the force strength at time t is

$$\vec{f}(s,t) = C\,\frac{\partial^2 \vec{X}}{\partial s^2}(s,t), \tag{4.95}$$

where s is the arc-length. With minor modification to handle the discontinuity in density, it should be possible to use this same approach to solve multi-flow problems with free boundaries.

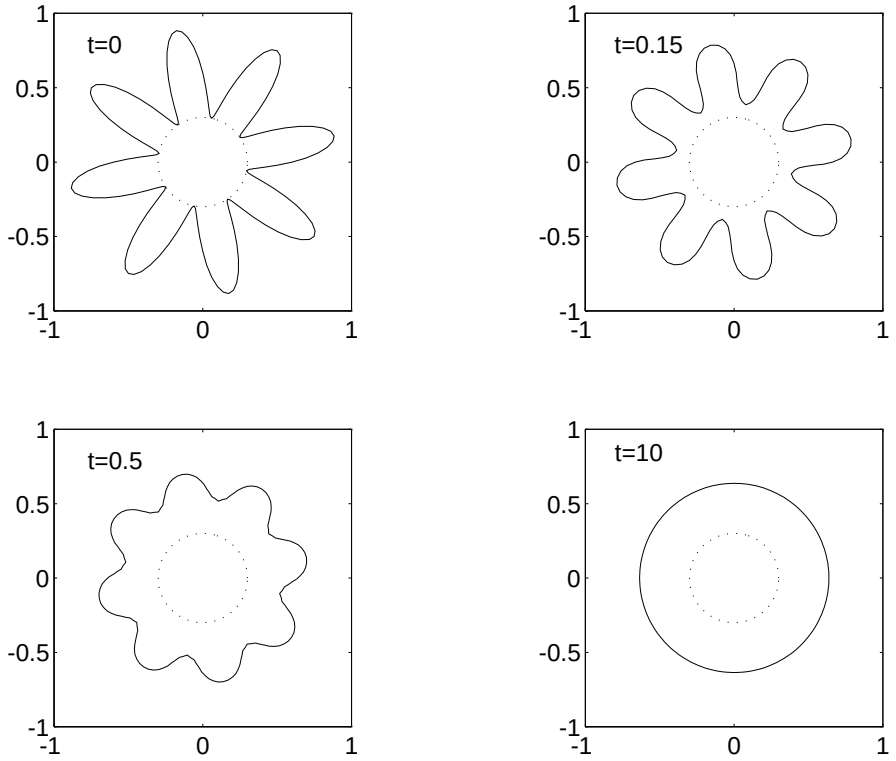


Figure 4.11: The interface at different times with a 160×160 grid. The dotted circle is the unstretched interface with $r = 0.3$.

Chapter 5

IMMERSED INTERFACE METHODS FOR 1D HEAT EQUATIONS WITH MOVING INTERFACES

In this chapter we study the immersed interface method for the one-dimensional heat equation with singular source terms and non-smooth solutions across some interface, which is also moving with time. As in Chapter 4, the interface is determined by an ordinary differential equation except now the ODE can be more complicated.

Beyer and LeVeque [6] studied various one-dimensional moving interface problems for the heat equation assuming a *priori* knowledge of the interface. At each time step a tridiagonal linear system of equations is solved to get the approximate solution if the Crank-Nicolson method is used. However, for the interface problems discussed here, the interface is unknown and the discrete difference form is a nonlinear system of equations involving the solution and the interface. We use *a predictor-corrector* scheme to find the solution and the interface simultaneously.

The purpose of Beyer and LeVeque's work in [6] is to analyse and improve Peskin's *immersed boundary method* for one-dimensional interface problems. A discrete delta function is carefully selected and some correction terms are added if necessary in their approach to get second order accuracy. However, with the *immersed interface method*, we use the jump relations to derive the modified difference scheme at irregular grid points.

The interest in this topic was motivated by the following:

- In Chapter 4, we discussed a simplication of the Navier-Stokes equations with the terms $\vec{u}_t$ and $(\vec{u} \cdot \nabla)\, \vec{u}$ absent. Eventually we want to apply the immersed interface method to the full Navier-Stokes equations with moving interfaces. So the work in this chapter is preparation for work in such a direction. One difficulty with $\vec{u}_t$ term is that it may discontinuous when the interface crosses grid lines. We need to modify the difference scheme to handle this discontinuity.

- Also the work here is a necessary step in applying the immersed interface method to solve Stefan-like problems

$$\vec{u}_t = \nabla \cdot (\beta \nabla \vec{u}),$$

$$\frac{\partial \vec{X}}{\partial t} = [\, \beta \, \vec{u}_n \,],$$

  where $\vec{X}$ is the interface along which discontinuities occur and the latent heat sources are applied. Such solidification (or melting) problems have broad applications and have been studied over the years. With the work discussed in Chapter 2, 3 and this chapter, it is not very difficult to use the immersed interface method for one-dimensional Stefan problems. The two-dimensional problem is more challenging and will be studied in the future.

5.1 The model problem and the outline of the numerical scheme.

Consider the following model problem:

$$u_t = u_{xx}, \qquad 0 \le x < \alpha(t) \qquad \text{and} \qquad \alpha(t) < x \le 1, \tag{5.1}$$

where the interface $\alpha(t)$ is determined by the following nonlinear ordinary differential equation

$$\frac{d\alpha}{dt} = w(\alpha, t, u(\alpha, t)), \tag{5.2}$$

where w is a known function and we have dropped t in the expression of $\alpha(t)$ and hereafter if no confusion occurs. The boundary and initial conditions are

$$\begin{aligned} &u(0,t) = h(t), \quad u(1,t) = g(t), \\ &u(x,0) = f(x). \end{aligned} \tag{5.3}$$

The interface $\alpha(t)$ divides the solution domain into two parts: $0 \le x < \alpha(t)$ and $\alpha(t) < x \le 1$ which we will denote as domain I and II domain, respectively.

To make the problem well-posed, we also need two interface conditions. First we assume that u is continuous for simplicity[1], which means $u(\alpha^-, t) = u(\alpha^+, t)$. The interface condition takes one of the following forms:

- Case 1: The jump in the derivative u_x is known

$$u_x(\alpha^+, t) - u_x(\alpha^-, t) = c(t). \tag{5.4}$$

  In this case, the equation can be written as

$$u_t = u_{xx} - c(t)\,\delta(x - \alpha(t)), \qquad 0 \le x \le 1, \tag{5.5}$$

  and the solutions in domains I and II are coupled together.

- Case 2: the interface value $u(\alpha, t) = r(t)$ is given. In this case, the problem can be solved separately in domains I and II if an explicit method is used.

- Case 3: The mixed boundary condition

$$a(t)\,u_x(\alpha^-, t) + b(t)\,u(\alpha, t) = v(t) \tag{5.6}$$

  or

$$a(t)\,u_x(\alpha^+, t) + b(t)\,u(\alpha, t) = v(t) \tag{5.7}$$

  is given. In this case, the problem can be solved independently in domain I (or II) with explicit approaches. But the solution in domain II (or I) depends on the other domain.

---

[1] It is not very difficult to handle the case where u is discontinuous using the techniques described in the previous chapters.

We use a uniform grid

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

and try to solve the problem for both domain I and II simultaneously. Let α^n be the computed approximation to the interface $\alpha(t^n)$. Using the Crank–Nicolson scheme, we write the difference scheme in the following general form

$$\frac{u_j^{n+1} - u_j^n}{k} = \frac{1}{2}\left[u_{xx,j}^n + u_{xx,j}^{n+1}\right] + Q_j^{n+\frac{1}{2}}. \tag{5.8}$$

As usual, we assume the ratio k/h is a constant, which means we can write $O(k)$ as $O(h)$ and so on. If $\alpha^l \notin [x_{j-1}, x_{j+1}]$ for $l = n$ or $n+1$, then

$$u_{xx,j}^l \approx \frac{u_{j-1}^l - 2u_j^l + u_{j+1}^l}{h^2}. \tag{5.9}$$

If there is no grid crossing, which means $x_j \notin (\alpha^n, \alpha^{n+1})$, then $Q_j^{n+\frac{1}{2}} = 0$. The interface is also determined by the trapezoidal method applied to (5.2)

$$\frac{\alpha^{n+1} - \alpha^n}{k} = \frac{1}{2}\left[w(\alpha^n, t^n, u^n) + w(\alpha^{n+1}, t^{n+1}, u^{n+1})\right], \tag{5.10}$$

here u^n and u^{n+1} are approximations to $u(\alpha, t^n)$ and $u(\alpha, t^{n+1})$ respectively. For different interface conditions, we will discuss different approaches to get $u_{xx,j}^n$, $u_{xx,j}^{n+1}$, u^n, u^{n+1} and $Q_j^{n+\frac{1}{2}}$. We are going to determine these quantities so that the local truncation errors are order $O(h)$ at the irregular grid points. In this way we still can ensure that the computed solution is second order accurate globally.

5.2 Grid crossing

If the interface crosses the grid from one time step to another, say there are some τ_j^n, with $t^n < \tau_j^n < t^{n+1}$ (see, Fig 5.1), such that $\alpha(\tau_j^n) = x_j$, then usually there is a jump in u_t and so $Q_j^{n+\frac{1}{2}}$ will not be zero. From [6] we know that

$$\begin{aligned} u(x_j, t^{n+1}) - u(x_j, t^n) &= \frac{k}{2}\left[u_t(x_j, t^n) + u_t(x_j, t^{n+1})\right] \\ &\quad + (t^{n+\frac{1}{2}} - \tau_j^n)\,[u_t]_{\tau_j^n} + O(k^2), \end{aligned} \tag{5.11}$$

where we define $[u_t]_\tau = u_t(x_j, \tau^+) - u_t(x_j, \tau^-)$. Therefore we should take

$$Q_j^{n+\frac{1}{2}} = \frac{1}{k}\left(t^{n+\frac{1}{2}} - \tau_j^n\right)\,[u_t]_{\tau_j^n}. \tag{5.12}$$

Differentiating the continuity condition $[u] = 0$ with respect to t we have

$$[u_x]\,\frac{d\alpha}{dt} + [u_t] = 0,$$

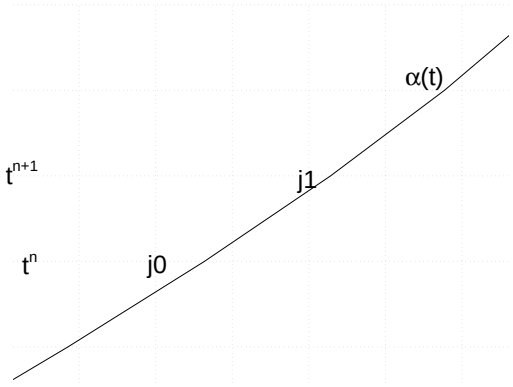


Figure 5.1: Interface crossing the grid.

i.e.,

$$[u_t] = [u_{xx}] = -[u_x]\,\frac{d\alpha}{dt} = -[u_x]\,w. \tag{5.13}$$

We will discuss how to find $u_{\tau_j^n} = u(\alpha, \tau_j^n)$ in the computation of $w(\alpha, \tau_j^n, u_{\tau_j^n})$ in the coming sections. We first discuss how to find τ_j^n if it exists. Using the Crank–Nicolson formula twice we get:

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

Replacing $\alpha(\tau_j^n)$ by x_j and eliminating the $w(\alpha, \tau_j^n, u_{\tau_j^n})$ term, we get

$$\begin{aligned}
&\frac{x_j - \alpha(t^n)}{\tau_j^n - t^n} - \frac{\alpha(t^{n+1}) - x_j}{t^{n+1} - \tau_j^n} = \\
&\qquad \frac{1}{2}\left[w\left(\alpha(t^n), t^n, u(\alpha, t^n)\right) - w\left(\alpha(t^{n+1}), t^{n+1}, u(\alpha, t^{n+1})\right)\right].
\end{aligned} \tag{5.14}$$

From this quadratic equation, we can solve for the crossing time τ_j^n. Numerically we actually use

$$\frac{x_j - \alpha^n}{\tau_j^n - t^n} - \frac{\alpha^{n+1} - x_j}{t^{n+1} - \tau_j^n} = \frac{1}{2}\left[w(\alpha^n, t^n, u^n) - w(\alpha^{n+1}, t^{n+1}, u^{n+1})\right], \tag{5.15}$$

where u^n and u^{n+1} are approximations to $u(\alpha^n, t^n)$ and $u(\alpha^{n+1}, t^{n+1})$ respectively.

Below we discuss the numerical algorithm for different interface conditions. We assume that $x_{j_0} \le \alpha^n < x_{j_0+1}$ and $x_{j_1} \le \alpha^{n+1} < x_{j_1+1}$, for $1 < j_0, j_1 < N$.

5.3 The source strength $c(t) = [u_x]$ is known.

In this case we interpolate u_j^l, $l = n$, or $n+1$, to get u^l, the approximate solution at the interface at time t^l. Define the following discrete delta function d_h^4

$$d_h^4(x) = \frac{1}{h} \begin{cases} 1 - (x/h)^2, & |x| \le h, \\ 2 - |3x/h| + (x/h)^2, & h \le |x| \le 2h, \\ 0 & \text{otherwise.} \end{cases} \tag{5.16}$$

From Lemma 4.2 in [6] we know that

$$u(\alpha, t) = h \sum_j u(x_j, t)\, d_h^4(x_j - \alpha) + O(h^2). \tag{5.17}$$

Thus u^n, u^{n+1} are computed from the following formula

$$u^n = h \sum_j u_j^n\, d_h^4(x_j - \alpha^n), \tag{5.18}$$

$$u^{n+1} = h \sum_j u_j^{n+1}\, d_h^4(x_j - \alpha^{n+1}). \tag{5.19}$$

If $x_j \le \alpha(t) < x_{j+1}$, using Taylor expansion about $\alpha(t)$ we can easily show that

$$\begin{aligned} u_{xx}(x_j, t) &= \frac{u(x_{j-1}, t) - 2u(x_j, t) + u(x_{j+1}, t)}{h^2} - \frac{x_{j+1} - \alpha(t)}{h^2}\,[u_x(\alpha, t)] \\ &\quad - \frac{(x_{j+1} - \alpha(t))^2}{h^2}\,[u_{xx}(\alpha, t)] + O(h), \end{aligned} \tag{5.20}$$

$$\begin{aligned} u_{xx}(x_{j+1}, t) &= \frac{u(x_j, t) - 2u(x_{j+1}, t) + u(x_{j+2}, t)}{h^2} + \frac{x_j - \alpha(t)}{h^2}\,[u_x(\alpha, t)] \\ &\quad + \frac{(x_j - \alpha(t))^2}{h^2}\,[u_{xx}(\alpha, t)]. \end{aligned} \tag{5.21}$$

Note: If $x_j = \alpha(t)$, then $u_x(x_j, t)$ and $u_{xx}(x_j, t)$ are defined as the left limiting value of $\alpha(t)$.

Therefore we can discretize u_{xx}^n and u_{xx}^{n+1} as follows

$$u_{xx,j}^n = \frac{u_{j-1}^n - 2u_j^n + u_{j+1}^n}{h^2} + C_j^n, \tag{5.22}$$

$$u_{xx,j}^{n+1} = \frac{u_{j-1}^{n+1} - 2u_j^{n+1} + u_{j+1}^{n+1}}{h^2} + C_j^{n+1}. \tag{5.23}$$

where C_j^n and C_j^{n+1} are determined in the following way:

$$\begin{aligned} &C_j^n = 0, \quad j = 1, 2, \cdots, j_0 - 1,\ j_0 + 2, \cdots, N - 1, \\ &C_{j_0}^n = -\frac{x_{j_0+1} - \alpha^n}{h^2}\, c(t^n) - \frac{(x_{j_0+1} - \alpha^n)^2}{h^2}\, c(t^n)\, w^n, \\ &C_{j_0+1}^n = \frac{x_{j_0} - \alpha^n}{h^2}\, c(t^n) + \frac{(x_{j_0} - \alpha^n)^2}{h^2}\, c(t^n)\, w^n, \end{aligned} \tag{5.24}$$

where $w^n = w(\alpha^n, t^n, u^n)$. Similarly

$$\begin{aligned} C_j^{n+1} &= 0, \quad j = 1, 2, \cdots, j_1 - 1, j_1 + 2, \cdots, N - 1, \\ C_{j_1}^{n+1} &= -\frac{x_{j_1+1} - \alpha^{n+1}}{h^2} c(t^{n+1}) - \frac{(x_{j_1+1} - \alpha^{n+1})^2}{h^2} c(t^{n+1})\, w^{n+1}, \\ C_{j_1+1}^{n+1} &= \frac{x_{j_1} - \alpha^{n+1}}{h^2} c(t^{n+1}) + \frac{(x_{j_1} - \alpha^{n+1})^2}{h^2} c(t^{n+1})\, w^{n+1}. \end{aligned} \tag{5.25}$$

If the interface crosses the grid line $x = x_j$, then we need to find $Q_j^{n+\frac{1}{2}}$ using (5.12). From (5.13) we know

$$[u_t(x_j, \tau_j^n)] = -c(\tau_j^n)\, w(x_j, \tau_j^n, u_{\tau_j^n}). \tag{5.26}$$

There are two ways to compute the unknown $u_{\tau_j^n} = u(x_j, \tau_j^n)$. The first one is simply to take the average

$$u_{\tau_j^n} = \frac{1}{2} \left(u^n + u^{n+1} \right). \tag{5.27}$$

Since $u(\alpha, t)$ is continuous, we can expect that

$$u(x_j, \tau_j^n) = \frac{1}{2} \left(u^n + u^{n+1} \right) + O(h).$$

Notice that $[u_t(x_j, \tau_j^n)]$ is multiplied by $\left(t^{n+\frac{1}{2}} - \tau_j^n \right)$ in (5.12), so this $O(h)$ error only contributes $O(h^2)$ to the local truncation error. The second approach is to use the forward Euler formula

$$u_{\tau_j^n} = u_j^n + \left(\tau_j^n - t^n \right) u_{xx,j}^n, \tag{5.28}$$

where we have used the relation $u_t = u_{xx}$. Now the difference scheme has been all set up for this case.

5.4 The solution on the interface $u(\alpha, t) = r(t)$ is known.

In this case, it is obvious that we can take

$$u^n = r(t^n), \qquad u^{n+1} = r(t^{n+1}). \tag{5.29}$$

To get the discrete $u_{xx,j}^n$ and $u_{xx,j}^{n+1}$, we use the following lemma.

Lemma 5.1 *Suppose $u(x)$ has continuous second derivative. If $h_1\, h_2 \neq 0$ and $h_1 \neq h_2$ then*

$$u_{xx}(x) = \gamma_1\, u(x + h_2) + \gamma_2\, u(x) + \gamma_3\, u(x + h_1) + O(h_1) + O(h_2), \tag{5.30}$$

where

$$\begin{aligned} &\gamma_1 = \frac{2}{h_2^2 - h_1\, h_2}, \qquad \gamma_3 = \frac{2}{h_1^2 - h_1\, h_2}, \\ &\gamma_2 = -\gamma_1 - \gamma_3. \end{aligned} \tag{5.31}$$

If $x_{j_0} = \alpha^n$, and $\alpha^{n+1} > \alpha^n$, this means that the point (x_j, t^{n+1}) is on the left side of the interface, so we use the one sided difference scheme to the left to get

$$u^n_{xx,j_0} = \frac{u^n_{j_0-2} - 2u^n_{j_0-1} + u^n_{j_0}}{h^2}. \tag{5.32}$$

Similarly if $x_{j_0} = \alpha^n$, and $\alpha^{n+1} < \alpha^n$ we use the one sided difference scheme to the right:

$$u^n_{xx,j_0} = \frac{u^n_{j_0} - 2u^n_{j_0+1} + u^n_{j_0+2}}{h^2}. \tag{5.33}$$

If $x_{j_0} < \alpha^n < x_{j_0+1}$, we use the Lemma above to set

$$u^n_{xx,j_0} = \gamma^n_{j_0-1}\, u^n_{j-1} + \gamma^n_{j_0}\, u^n_j + \gamma^n_{j_0,\alpha}\, r(t^n); \tag{5.34}$$

here $\gamma^n_{j_0-1}$, $\gamma^n_{j_0}$ and $\gamma^n_{j_0,\alpha}$ are determined by Lemma 5.1 with $h_1 = -h$, $h_2 = \alpha^n - x_j$. By applying the same process at x_{j_0+1} we set

$$u^n_{xx,j_0+1} = \gamma^n_{j_0+1,\alpha}\, r(t^n) + \gamma^n_{j_0+1,2}\, u^n_j + \gamma^n_{j_0+1,3}\, u^n_{j+1}; \tag{5.35}$$

here again $\gamma^n_{j_0+1,\alpha}$, $\gamma^n_{j_0+1,2}$ and $\gamma^n_{j_0+1,3}$ are determined by Lemma 5.1 with $h_1 = \alpha^n - x_j$, $h_2 = h$.

If $x_{j_1} = \alpha^{n+1}$, since we know $u(\alpha^{n+1}, t^{n+1})$ the equation at x_{j_1} simply becomes

$$u_{j_1} = r(t^{n+1}). \tag{5.36}$$

Otherwise if $x_{j_1} \le \alpha^{n+1} < x_{j_1+1}$, we set

$$u^{n+1}_{xx,j_1} = \gamma^{n+1}_{j_1,1}\, u^{n+1}_{j_1-1} + \gamma^{n+1}_{j_1,2}\, u^{n+1}_{j_1} + \gamma^{n+1}_{j_1,\alpha}\, r(t^{n+1}); \tag{5.37}$$

here again the coefficients are determined by the formula in Lemma 5.1 with $h_2 = -h$, and $h_1 = \alpha^{n+1} - x_{j_1}$, and

$$u^{n+1}_{xx,j_1+1} = \gamma^{n+1}_{j_1+1,\alpha}\, r(t^{n+1}) + \gamma^{n+1}_{j_1+1,2}\, u^{n+1}_{j_1+1} + \gamma^{n+1}_{j_1+1,3}\, u^{n+1}_{j_1+2} \tag{5.38}$$

with $h_2 = x_{j_1+1} - \alpha^{n+1}$, $h_1 = h$.

If the grid crossing occurs at x_j, then $[u_t]_{\tau^n_j}$ in (5.12) can be calculated using

$$[u_t]_{\tau^n_j} = \frac{u^{n+1}_j - r(\tau^n_j)}{t^{n+1} - \tau^n_j} - \frac{r(\tau^n_j) - u^n_j}{\tau^n_j - t^n}. \tag{5.39}$$

This approach, however, may be unstable sometimes, especially for the mixed interface condition discussed in the next section. A more stable method is the following formula:

$$[u_t]_{\tau^n_j} = \begin{cases} u^n_{xx,j_0} - \dfrac{r(\tau^n_j) - u^n_j}{\tau^n_j - t^n} & \text{if } \alpha^n < \alpha^{n+1}, \\ \\ u^n_{xx,j_0+1} - \dfrac{r(\tau^n_j) - u^n_j}{\tau^n_j - t^n} & \text{if } \alpha^n > \alpha^{n+1}, \end{cases} \tag{5.40}$$

where u^n_{xx,j_0} is determined from one of (5.32)–(5.34), and u^n_{xx,j_0+1} is from (5.35). Although this approach may have larger error constant than (5.39), it is more stable.

5.5 The mixed interface condition is known.

If we know the mixed interface condition

$$a(t)u_x(\alpha^-,t)+b(t)u(\alpha,t)=v(t), \tag{5.41}$$

we still can use (5.18) and (5.19) to approximate u^n and u^{n+1}. However a simpler way is to use the mixed boundary condition directly. We have the following lemma.

Lemma 5.2 *Suppose* $x_j \le \alpha < x_{j+1}$, *and* $au_x(\alpha,t)+bu(\alpha,t)=v(t)$. *Then*

$$u(\alpha,t)=\gamma_1 u(x_j,t)+\gamma_2 v(t)+O(h^2), \tag{5.42}$$

where

$$\begin{aligned}\gamma_1 &= \frac{a}{a-b\,(x_j-\alpha)},\\ \gamma_2 &= -\frac{(x_j-\alpha)}{a-b\,(x_j-\alpha)}.\end{aligned} \tag{5.43}$$

Proof: From the mixed boundary condition we have

$$au(\alpha,t)+au_x(\alpha,t)(x_j-\alpha)=v(t)(x_j-\alpha)-bu(\alpha,t)(x_j-\alpha)+au(\alpha,t).$$

Notice that

$$u(x_j,t)=u(\alpha,t)+u_x(\alpha,t)(x_j-\alpha)+O(h^2).$$

So we can write

$$au(x_j,t)=u(\alpha,t)(a-b(x_j-\alpha))+v(t)(x_j-\alpha)+O(h^2).$$

That is,

$$u(\alpha,t)=\frac{au(x_j,t)-v(t)(x_j-\alpha)}{a-b(x_j-\alpha)}+O(h^2),$$

which also gives (5.42) and (5.43). □

With this lemma we can get a new interpolation formula for u^n and u^{n+1}.

$$u^n=\frac{a(t^n)u^n_{j_0}-v(t^n)(x_{j_0}-\alpha^n)}{a(t^n)-b(t^n)(x_{j_0}-\alpha^n)}, \tag{5.44}$$

$$u^{n+1}=\frac{a(t^{n+1})u^{n+1}_{j_1}-v(t^{n+1})(x_{j_1}-\alpha^{n+1})}{a(t^{n+1})-b(t^{n+1})(x_{j_1}-\alpha^{n+1})}. \tag{5.45}$$

Numerical experiments show little difference between the formulas (5.18) and (5.44), though the latter one seems to give slightly better results. We will see a more significant application of Lemma 5.2 and (5.45) when we try to get u^{n+1}_{xx,j_1+1}.

To get u^n_{xx,j_0} and u^{n+1}_{xx,j_1} we need somehow to incorporate the mixed interface condition. We need the following lemma.

Lemma 5.3 *Suppose* $x_j \le \alpha < x_{j+1}$ *and* $au_x(\alpha,t) + bu(\alpha,t) = v(t)$. *Then*

$$u_{xx}(\alpha,t) = \gamma_1 u(x_{j-1},t) + \gamma_2 u(x_j,t) + \gamma_3 v(t) + O(h), \tag{5.46}$$

where

$$\gamma_1 = \frac{2\,(a - b(x_j - \alpha))}{D}; \qquad \gamma_2 = \frac{-2\,(a - b(x_{j-1} - \alpha))}{D},$$

$$D = a(x_{j-1} - \alpha)^2 - b(x_{j-1} - \alpha)^2(x_j - \alpha) - a(x_j - \alpha)^2 + b(x_{j-1} - \alpha)(x_j - \alpha)^2,$$

$$\gamma_3 = \frac{2}{-a(x_{j-1} - \alpha) - a(x_j - \alpha) + b(x_{j-1} - \alpha)(x_j - \alpha)},$$

where γ_1, γ_2 *and* γ_3 *are the solutions of the following linear system*

$$\begin{aligned} \gamma_1 + \gamma_2 + \gamma_3\, b &= 0, \\ \gamma_1(x_{j-1} - \alpha) + \gamma_2(x_j - \alpha) + \gamma_3\, a &= 0, \\ \gamma_1 \frac{(x_{j-1} - \alpha)^2}{2} + \gamma_2 \frac{(x_j - \alpha)^2}{2} &= 1. \end{aligned} \tag{5.47}$$

Note: The formula (5.46) is quite useful in getting second order difference schemes for 1D differential equations with mixed boundary conditions. For example, if we want to solve $u'' = f$ with the mixed boundary condition $au_x(\alpha) + bu = c$ at $x_n = b$, and discretize the mixed BC directly

$$a\frac{u_n - u_{n-1}}{h} + bu_n = c,$$

we would get the solution with only first order accuracy globally unless all of b, c and f_n are zero. Here $h = x_n - x_{n-1}$. But if we use the formula of the lemma which now is

$$\frac{2}{h^2}\,(u_{n-1} - u_n) - \frac{2\,b}{a\,h}\,u_n + \frac{2}{a\,h}\,c = f_n,$$

we would get a second order accurate solution globally. Notice that we still only use information on u_n and u_{n-1}, so the structure of the resulting system is still tridiagonal.

With the formula (5.46), we can get expressions for u^n_{xx,j_0} and u^{n+1}_{xx,j_1}. If $x_{j_0} \le \alpha^n < x_{j_0+1}$, then

$$u^n_{xx,j_0} = \gamma^n_{j_0,1}\,u^n_{j_0-1} + \gamma^n_{j_0,2}\,u^n_{j_0} + \gamma^n_{j_0,\alpha}\,v(t^n), \tag{5.48}$$

where $\gamma^n_{j_0,1}$, $\gamma^n_{j_0,2}$ and $\gamma^n_{j_0,\alpha}$ are determined by Lemma 5.3 with $a = a(t^n)$, $b = b(t^n)$, and $v(t) = v(t^n)$. Similarly

$$u^{n+1}_{xx,j_1} = \gamma^{n+1}_{j_1,1}\,u^{n+1}_{j_1-1} + \gamma^{n+1}_{j_1,2}\,u^{n+1}_{j-1} + \gamma^{n+1}_{j_1,\alpha}\,v(t^{n+1}) \tag{5.49}$$

with $a = a(t^{n+1}), b = b(t^{n+1})$ and $v(t) = v(t^{n+1})$.

To compute u^n_{xx,j_0+1} we use (5.35) with $r(t^n)$ replaced by u^n, which we can get either from (5.18) or from (5.44). Generally the results are pretty much the same if u^n_js are second-order accurate.

But for u^{n+1}_{xx,j_1+1}, it seems that (5.45) gives better results than (5.19) since we use the interface condition more directly. Below we derive the discretized formula for $u_{xx}(x_{j_1+1}, t^{n+1})$. From (5.38) we know

$$u^{n+1}_{xx,j_1+1} = \gamma^{n+1}_{j_1+1,\alpha}\, u^{n+1} + \gamma^{n+1}_{j_1+1,2}\, u^{n+1}_{j_1+1} + \gamma^{n+1}_{j_1+1,3}\, u^{n+1}_{j_1+2}, \tag{5.50}$$

where $\gamma^{n+1}_{j_1+1,\alpha}$, $\gamma^{n+1}_{j_1+1,2}$ and $\gamma^{n+1}_{j_1+1,3}$ are determined by (5.31) with $h_1 = \alpha^{n+1} - x_{j_1}$, $h_2 = h$. Plugging (5.45) in we can rewrite it as

$$\begin{aligned} u^{n+1}_{xx,j_1+1} &= \gamma^{n+1}_{j_1+1,1} \frac{a(t^{n+1})u^{n+1}_{j_1} - v(t^{n+1})(x_{j_1} - \alpha^{n+1})}{a(t^{n+1}) - b(t^{n+1})(x_{j_1} - \alpha^{n+1})} + \gamma^{n+1}_{j_1+1,2}\, u^{n+1}_{j_1+1} + \gamma^{n+1}_{j_1+1,3}\, u^{n+1}_{j_1+2} \\ &= \gamma^{n+1}_{j_1+1,1'}\, u^{n+1}_{j_1} + \gamma^{n+1}_{j_1+1,2}\, u^{n+1}_{j_1+1} + \gamma^{n+1}_{j_1+1,3}\, u^{n+1}_{j_1+2} + C^{n+1}_{j_1+1}, \end{aligned} \tag{5.51}$$

where

$$\gamma^{n+1}_{j_1+1,1'} = \gamma^{n+1}_{j_1+1,\alpha} \frac{a(t^{n+1})}{a(t^{n+1}) - b(t^{n+1})(x_{j_1} - \alpha^{n+1})}, \tag{5.52}$$

$$C^{n+1}_{j_1+1} = -\gamma^{n+1}_{j_1+1,\alpha} \frac{v(t^{n+1})(x_{j_1} - \alpha^{n+1})}{a(t^{n+1}) - b(t^{n+1})(x_{j_1} - \alpha^{n+1})}. \tag{5.53}$$

When grid crossing occurs, one method to calculate $u(\alpha, \tau^n_j)$ needed for $Q^{n+\frac{1}{2}}_j$ in (5.12) is

$$[u_t]_{\tau^n_j} = \begin{cases} u^n_j + \left(\tau^n_j - t^n\right) u^n_{xx,j} & \text{if } \alpha^n < \alpha^{n+1}, \\ u^n_j + \left(\tau^n_j - t^n\right) u^n_{xx,j+1} & \text{if } \alpha^n > \alpha^{n+1}. \end{cases} \tag{5.54}$$

Another approach is described below. First we get

$$u_{j-1,\tau^n_j} = u^n_{j-1} + \left(\tau^n_j - t^n\right) u^n_{xx,j}, \tag{5.55}$$

where u_{j-1,τ^n_j} is the approximation of $u\left(x_{j-1}, \tau^n_j\right)$. Then use Lemma 5.3 to get $u_{\tau^n_j}$:

$$u_{\tau^n_j} = \beta^n_{j,1}\, u_{j-1,\tau^n_j} + \beta^n_{j,2}\, v(\tau^n_j), \tag{5.56}$$

where

$$\begin{aligned} \beta^n_{j,1} &= \frac{a}{a + h\,b}, \\ \beta^n_{j,2} &= \frac{h}{a + h\,b}. \end{aligned} \tag{5.57}$$

Both methods are approximately second order accurate. But the latter method gives slightly better results. Also the error decreases more smoothly with the latter choice.

5.6 Solving the nonlinear system: a predictor–Corrector method

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

At a regular grid point with $\alpha^n \notin (x_{j-1}, x_{j+1})$ and $\alpha^{n+1} \notin (x_{j-1}, x_{j+1})$, and there is no grid crossing, the difference scheme is

$$\frac{u_j^{n+1} - u_j^n}{k} = \frac{1}{2}\left[\frac{u_{j-1}^n - 2u_j^n + u_{j+1}^n}{h^2} + \frac{u_{j-1}^{n+1} - 2u_j^{n+1} + u_{j+1}^{n+1}}{h^2}\right].$$

Otherwise

$$\begin{aligned}
\frac{u_j^{n+1} - u_j^n}{k} &= \frac{1}{2}\left[\gamma_{j-1}^n u_{j-1}^n + \gamma_j^n u_j^n + \gamma_{j+1}^n u_{j+1}^n + C_j^n + \right. \\
&\quad \left. \gamma_{j-1}^{n+1} u_{j-1}^{n+1} + \gamma_j^{n+1} u_j^{n+1} + \gamma_{j+1}^{n+1} u_{j+1}^{n+1} + C_{j+1}^{n+1}\right] + Q_j^{n+\frac{1}{2}}, \\
\frac{\alpha^{n+1} - \alpha^n}{k} &= \frac{1}{2}\left[w(\alpha^n, t^n, u^n) + w(\alpha^{n+1}, t^{n+1}, u^{n+1})\right].
\end{aligned}$$

The quantities $\{\gamma_j^n\}$ depend on α^n, u^n, the approximation to $u(\alpha, t^n)$, and the interface condition, and the quantities $\{\gamma_j^{n+1}\}$ depend on α^{n+1}, u^{n+1}, as well as the interface condition. The correction term $Q_j^{n+\frac{1}{2}}$ to $(u_j^{n+1} - u_j^n)/k$ depends not only on $\alpha^n, u^n, \alpha^{n+1}, u^{n+1}$, and the interface condition, but also the relative position between α^n, α^{n+1} and the uniform grid. We have shown how to get these quantities in previous sections for different interface conditions.

So we have a quite complicated nonlinear system to solve. The difficulty is that we can not use a nonlinear equations solver just once to solve the system because some quantities such as $Q_j^{n+\frac{1}{2}}, C_{j+1}^{n+1}$ are not known until we know the solution. So the best method to solve the resulting system seems to be a *predictor-corrector* method by which we can adjust those quantities during the predicting and correcting process. Below we give the outline of this approach.

Suppose we have gotten all necessary quantities at the time t^n, and the next time step is k, so $t^{n+1} = t^n + k$. We want to get all corresponding quantities at time level t^{n+1}.

- Determine j_0 such that $x_{j_0} \le \alpha^n < x_{j_0+1}$. Get the approximation to $u_{xx,j}^n$. For example, if $j \neq j_0,\ j_0 + 1$ then we can use the central difference scheme. Otherwise we use the discretization we have studied in the previous section.

- Set

$$u_1^{n+1} = u^n, \qquad \alpha_1^{n+1} = \alpha^n + k\, w(\alpha^n, t^n, u^n).$$

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

- (**) Determine $\{j_1\}_m$ such that $x_{\{j_1\}_m} \le \alpha_1^{n+1} < x_{\{j_1\}_m+1}$. Get the discretized expressions for $\{u_{xx,j}^{n+1}\}_m$ so that we can get the coefficients of the linear system for $\{u_j^{n+1}\}_m$.

- If $j_0 \neq \{j_1\}_m$, then for $l = j_0 + 1, \cdots, \{j_1\}_m$, when $j_0 < \{j_1\}_m$, or for $l = \{j_1\}_m, \{j_1\}_m + 1, \cdots, j_0$, when $j_0 > \{j_1\}_m$, first get $\{\tau_j^n\}_m$ using (5.15), then determine the correction $\{Q_l^{n+\frac{1}{2}}\}_m$ to $(\{u_l^{n+1}\}_m - u_l^n)/k$ using the technique described in previous section.

- Solve the tridiagonal system for $\{u_j^{n+1}\}_m$.

- Use one of the interpolation methods discussed before to get u_m^{n+1}.

- Determine

$$\alpha_{m+1}^{n+1} = \alpha^n + \frac{k}{2}\left[w(\alpha^n, t^n, u^n) + w(\alpha_m^{n+1}, t^{n+1}, u_{m+1}^{n+1})\right]. \tag{5.58}$$

- If $|\alpha_m^{n+1} - \alpha_{m+1}^{n+1}| > \epsilon$, a given tolerance, then $m = m + 1$, go to (**).

- If $|\alpha_m^{n+1} - \alpha_{m+1}^{n+1}| < \epsilon$, then set all quantities $\{\ \}_m^{n+1}$ to $\{\ \}^{n+1}$, in other words we drop the $\{m\}$ notation and accept these values at time t^{n+1}. Go to the next time step.

5.7 Numerical examples

In order to check the algorithm we proposed here, we use the following true solution.

$$u(x,t) = \begin{cases} \sin(\omega_1 x) e^{-{\omega_1}^2 t} & \text{if } x \leq \alpha(t) \\ \\ \sin\left(\omega_2 - \omega_2 x\right) e^{-{\omega_2}^2 t} & \text{if } x \geq \alpha(t) \end{cases} \tag{5.59}$$

for some choice of ω_1 and ω_2. The interface $\alpha(t)$ is determined by the scalar equation

$$\sin(\omega_1 \alpha)\, e^{-{\omega_1}^2 t} = \sin\left(\omega_2 - \omega_2 \alpha\right) e^{-{\omega_2}^2 t}. \tag{5.60}$$

This equation has a unique solution if we take, for example, $\pi < \omega_1 < 2\pi$ and also $\pi < \omega_2 < 2\pi$. Figure 5.2 gives the plot of $\alpha(t)$ on a uniform grid. We can see how the interface crosses the grid. This example is adapted from [6].

We will test the same PDE with the same initial and boundary conditions but different *interface conditions*. The PDE is

$$u_t = u_{xx}, \qquad 0 \leq x < \alpha(t) \qquad \text{and} \qquad \alpha(t) < x \leq 1,$$

where the interface $\alpha(t)$ is determined by the following nonlinear ordinary differential equation

$$\frac{d\alpha}{dt} = \frac{(\omega_1^2 - \omega_2^2)\, u(\alpha, t)}{\omega_1 \cos(\omega_1 \alpha)\, e^{-{\omega_1}^2 t} + \omega_2 \cos\left(\omega_2 - \omega_2 \alpha\right) e^{-{\omega_2}^2 t}}. \tag{5.61}$$

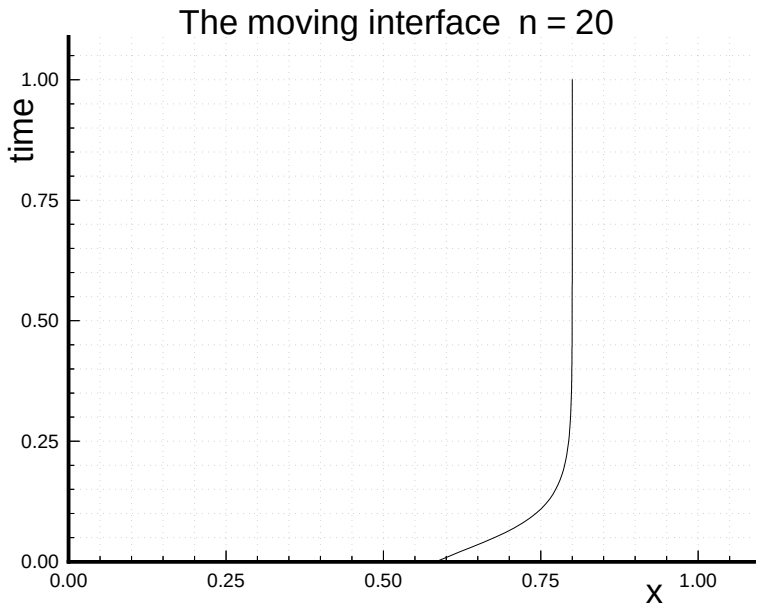


Figure 5.2: Moving interface $\alpha(t)$, $0 \le t \le 1$.

The initial and boundary conditions are

$$u(0,t) = 0, \qquad u(1,t) = 0, \tag{5.62}$$

$$u(x,0) = \begin{cases} \sin(\omega_1 x) & \text{if } x \le \alpha(0), \\ \sin\left(\omega_2 - \omega_2\, x\right) & \text{if } x \ge \alpha(0). \end{cases} \tag{5.63}$$

One of the interface conditions is $u(\alpha^-, t) = u(\alpha^+, t)$. Below we show numerical results for $u(x, 0.1)$ for different interface conditions.

Example 5.1 *In this example we specify the jump condition in the derivative u_x:*

$$[u_x(\alpha,t)] = -\omega_2 \cos\left(\omega_2 - \omega_2\,\alpha\right) e^{-{\omega_2}^2 t} - \omega_1 \cos(\omega_1 \alpha)\, e^{-{\omega_1}^2 t}. \tag{5.64}$$

Figure 5.3 shows the computed profile of $u(x,t)$ as t changes from 0 to 0.1. We can clearly see how the interface moves and crosses the grid with time.

Table 5.1 shows the computed results and the convergence rate both for the solution and the computed interface when we take $k = h$. In this case, the interface crosses two grid points during the first few time steps (see Fig 5.2). Table 5.2 shows the results when we take $k = h/2$. Now the interface only crosses at most one grid. We can see that the results are much better compared to the previous case. But in both cases we can observe second order convergence.

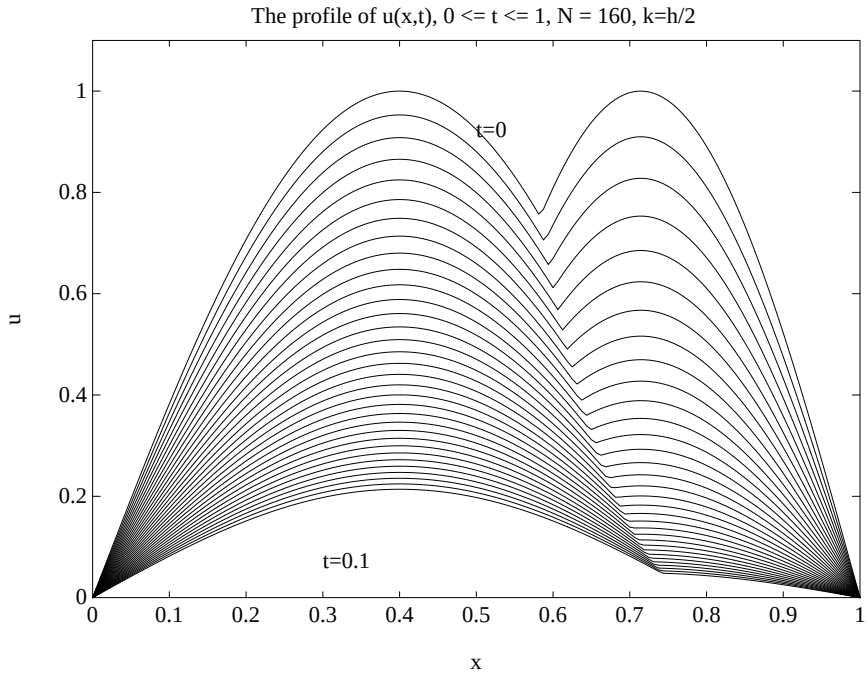


Figure 5.3: The profile of the computed solution $u(x,t)$ from $t = 0$ to $t = 0.1$ for Example 5.1 with $N = 160$, $k = h$. The interface condition: $[u_x(\alpha,t)]$ is given.

Table 5.1: Numerical results for Example 5.1 at $t = 0.1$ with $N = 40$, $k = h$. The interface condition: $[u_x(\alpha,t)]$ is given.

N	$\| T_N \|_\infty$	rate	$\| E_N \|_\infty$	rate	$\lvert E_\alpha \rvert$	rate
40	2.2991×10^{-1}		1.8721×10^{-2}		1.2988×10^{-2}	
80	1.0658×10^{-1}	2.1572	4.7573×10^{-3}	3.9353	3.2314×10^{-3}	4.0193
160	5.0962×10^{-2}	2.0914	1.1620×10^{-3}	4.0940	7.9777×10^{-4}	4.0505
320	2.5074×10^{-2}	2.0324	2.9005×10^{-4}	4.0062	1.9890×10^{-4}	4.0108
640	1.2496×10^{-2}	2.0065	7.2502×10^{-5}	4.0006	4.9867×10^{-5}	3.9887
1280	6.2127×10^{-3}	2.0115	1.8087×10^{-5}	4.0084	1.2440×10^{-5}	4.0084
2560	3.0998×10^{-3}	2.0042	4.5182×10^{-6}	4.0032	3.1091×10^{-6}	4.0014
5120	1.5482×10^{-3}	2.0022	1.1293×10^{-6}	4.0009	7.7710×10^{-7}	4.0009

Table 5.2: Numerical results for Example 5.1 at $t = 0.1$ with $N = 40$, $k = h/2$. The interface condition: $[u_x(\alpha, t)]$ is given.

N	$\| T_n \|_\infty$	rate	$\| E_n \|_\infty$	rate	$\lvert E_\alpha \rvert$	rate
40	2.1754×10^{-2}		3.7607×10^{-3}		2.3650×10^{-3}	
80	1.0641×10^{-2}	2.0443	9.8655×10^{-4}	3.8120	6.1332×10^{-4}	3.8562
160	5.0008×10^{-3}	2.1279	2.4299×10^{-4}	4.0599	1.5398×10^{-4}	3.9830
320	2.4580×10^{-3}	2.0345	6.0753×10^{-5}	3.9997	3.8577×10^{-5}	3.9916
640	1.2127×10^{-3}	2.0269	1.5179×10^{-5}	4.0023	9.6215×10^{-6}	4.0094
1280	6.0273×10^{-4}	2.0120	3.7976×10^{-6}	3.9971	2.4058×10^{-6}	3.9992
2560	3.3004×10^{-4}	2.0063	9.48813×10^{-7}	4.0025	6.0145×10^{-7}	4.0001
5120	1.5015×10^{-4}	2.0008	2.3703×10^{-7}	4.0028	1.5057×10^{-7}	3.9943

In this section, $\| E_N \|_\infty$ is defined as the infinity norm of the error at the fixed time t, *i.e.*,

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

where u_i^N is the computed solution at the uniform grid points x_i, $i = 1, 2, \ldots$, at some time t, with the number of grid points being N. T_N is the local truncation error at time t, and E_α is the error between $\alpha(t)$ and computed interface at the time t. The rates are defined as the ratio of the errors with the number of grid points N and $N/2$; for example

$$\text{rate} = \| E_{2\,N} \|_\infty / \| E_N \|_\infty.$$

Figure 5.4 (a) shows the true solution and the computed solution for $k = h/2$. Figure 5.4 (b) shows the corresponding error plot. We see that the error is relatively large around the interface but not significant so. Globally we obtain second order accurate results at all grid points.

Example 5.2 *In this example we specify the solution on the interface $u(\alpha, t) = c(t)$:*

$$c(t) = \sin(\omega_1 \alpha)\, e^{-\omega_1^2 t}.$$

Table 5.3 shows the results for $N = 40, k = h/2$. Figure 5.5 shows the corresponding error plot.

Example 5.3 *The mixed interface condition is known. We take*

$$a(t) = b(t) = 1$$
$$v(t) = u_x(\alpha, t) + u(\alpha, t) = e^{-\omega_1^2 t}\, \left(-\omega_1 \cos(-\omega_1 x) + \sin(-\omega_1 x)\right).$$

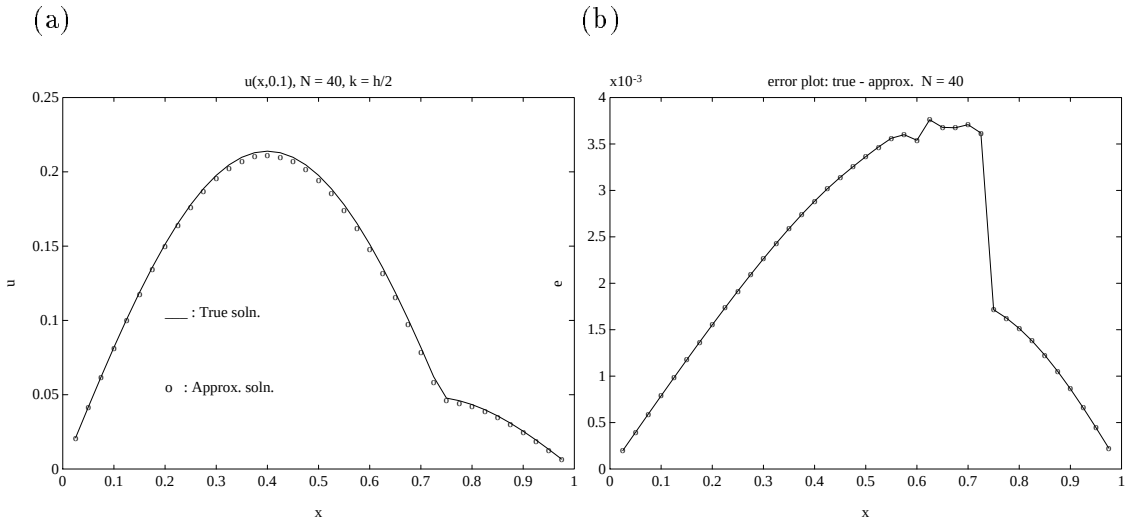


Figure 5.4: The comparison of the exact and computed solution at $t = 0.1$ for Example 5.1 with $N = 40$ and $k = h/2$. (a) The solid line is the exact solution and the little 'o's' are the computed results at grid points. (b) The corresponding error plot.

Table 5.3: Numerical results for Example 5.2 at $t = 0.1$ with $N = 40$, $k = h$. The interface condition: $u(\alpha, t)$ is given.

N	$\| T_N \|_\infty$	rate	$\| E_N \|_\infty$	rate	$\lvert E_\alpha \rvert$	rate
40	9.4190×10^{-3}		6.7882×10^{-3}		1.3056×10^{-4}	
80	1.6565×10^{-3}	5.6859	1.4829×10^{-4}	4.5776	3.2481×10^{-5}	4.0197
160	1.6894×10^{-3}	0.9806	3.7328×10^{-5}	3.9727	8.1105×10^{-6}	4.0049
320	3.5552×10^{-4}	4.7518	9.3387×10^{-6}	3.9971	2.0270×10^{-6}	4.0012
640	6.7189×10^{-5}	5.2914	2.3357×10^{-6}	3.9982	5.0671×10^{-7}	4.0003
1280	3.5602×10^{-5}	1.7626	5.8414×10^{-7}	3.9985	1.2667×10^{-7}	4.0000
2560	2.0199×10^{-5}	1.7626	1.4607×10^{-7}	3.9990	3.1668×10^{-8}	4.0000
5120	1.3454×10^{-5}	1.5013	3.6540×10^{-8}	3.9975	7.9172×10^{-9}	4.0000

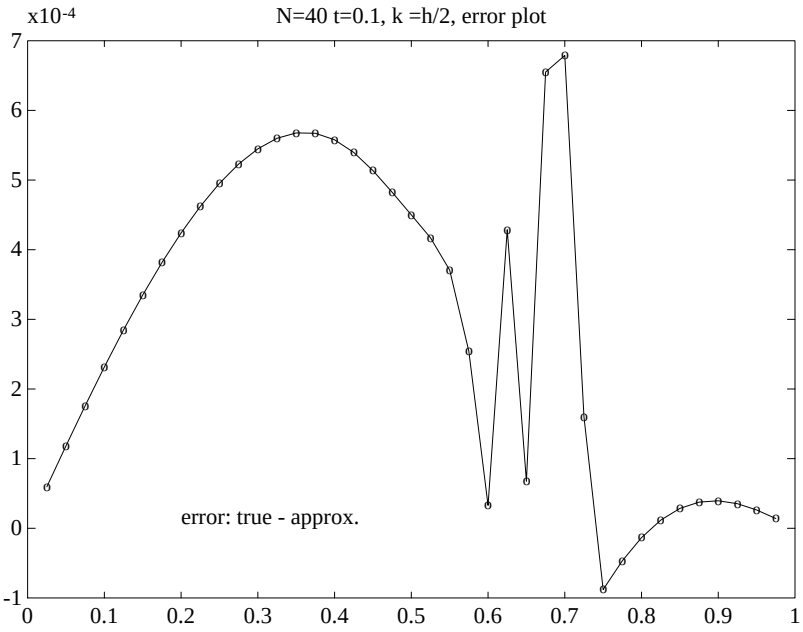


Figure 5.5: The error plot for Example 5.2 at $t = 0.1$ with $N = 40$, $k = h/2$, when $u(\alpha, t)$ is known.

Table 5.4: Numerical results for Example 5.3 at $t = 0.1$ with $N = 40$, $k = h$. The mixed interface condition: $u_x(\alpha-, t) + u(\alpha-, t)$ is given.

N	$\| T_N \|_\infty$	rate	$\| E_N \|_\infty$	rate	$\lvert E_\alpha \rvert$	rate
40	4.4010×10^{-3}		4.1399×10^{-4}		3.9243×10^{-4}	
80	1.4157×10^{-3}	3.1086	7.4926×10^{-5}	5.5253	8.1050×10^{-5}	4.8418
160	1.4910×10^{-3}	0.9495	2.2238×10^{-5}	3.3692	2.5377×10^{-5}	3.1938
320	4.2727×10^{-4}	3.4896	6.1500×10^{-6}	3.6160	6.2013×10^{-6}	4.0922
640	1.0325×10^{-4}	4.1381	1.1984×10^{-6}	5.1316	1.5761×10^{-6}	3.9344
1280	8.3619×10^{-5}	1.2348	4.7471×10^{-7}	2.5246	4.6013×10^{-7}	3.4255
2560	3.0763×10^{-5}	1.1522	7.6285×10^{-8}	6.2229	9.8304×10^{-8}	4.6807
5120	2.6699×10^{-5}	1.1522	3.5189×10^{-7}	2.1679	3.3159×10^{-8}	2.9647

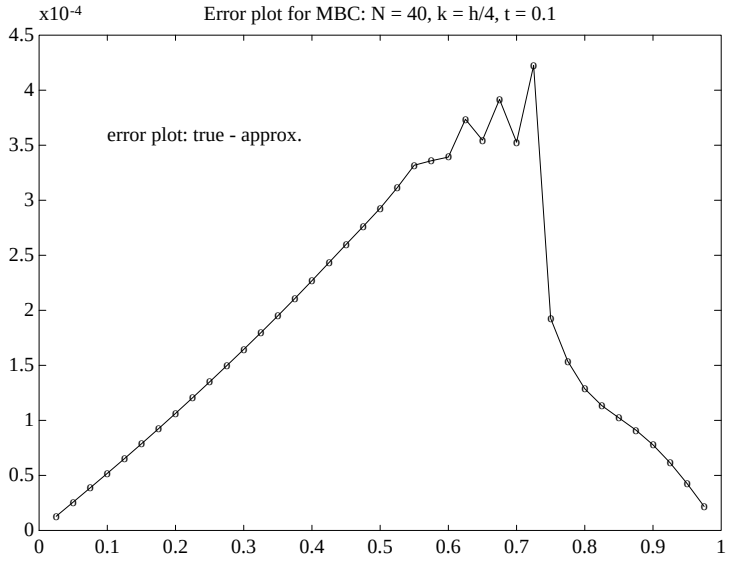


Figure 5.6: The error plot for example 5.3 at $t = 0.1$ with $N = 40$, $k = h/4$. Mixed interface condition: $u_x(\alpha-, t) + u(\alpha-, t)$ is given.

Table 5.4 shows the results when $N = 40, k = h/4$. Figure 5.6 shows the error plot for this case. The error does not decrease smoothly. But it exhibits roughly second order accuracy.

In summary we have developed a second order accurate immersed interface method for the 1D heat equation with a moving a interface for three different types of interface conditions: (i) a derivative jump condition (ii) a Dirichlet conditions (iii) a mixed interface condition. Numerical experiments have confirmed the efficiency of the methods proposed in this chapter. In the near future we are planning to study similar numerical method for discontinuous coefficients, *e.g.*, the Stefan problem for phase transition, and also higher dimensional problems.

Chapter 6

THESIS CONTRIBUTION AND FUTURE RESEARCH PLAN

6.1 Thesis contribution

6.1.1 The Immersed interface method.

In this thesis a new approach, which we call *the immersed interface method*, for solving interface problems has been proposed and studied. This approach can handle quite complicated problems including the following: discontinuous coefficients, singular sources or dipoles across interfaces, fixed or moving interfaces which can be arbitrary and usually are not aligned with the grid lines. The approach is also robust in the sense that when the singularities disappear, the approach reduces to a conventional method for a regular problem.

The key idea of this approach is to use the jump conditions across the interface and the differential equation itself to derive appropriate numerical schemes. In this thesis, the basic computational framework is finite difference schemes on uniform Cartesian grids. The convergence discussion is based on local truncation error analysis. For the interface problems discussed in this thesis we are able to get second order accuracy in the infinity norm even if the solution is discontinuous. We believe that the ideas discussed in this thesis can be used in other computational frameworks such as adaptive methods, or finite element methods as well.

In the process of implementing the immersed interface methods for the problems discussed in this thesis, we also derived the interface relations for the solution and its derivatives across the interface based on the given jump conditions, coordinates transformation and the differential equation itself. Those interface relations are very useful for not only deriving efficient numerical schemes but also for revealing better understanding of the problem.

Also in this thesis, a number of interpolation formulas for the solution with discontinuities have been proposed. For example, in Chapter 4 we interpolated the grid function of the pressure p_{ij}, which is discontinuous across the interface to get $(p_x)_{ij}$ and $(p_y)_{ij}$, and the grid function of velocity $\vec{u}_{ij}$, which is non-smooth, to get the velocity of the interface $\vec{X}_k$.

In solving the interface problems described in this thesis, a number of advanced numerical techniques have been applied which include: cubic spline interpolation in expressing the interface, fast Fourier transformations, optimization approaches, and a predictor-corrector method for solving nonlinear equations etc.

6.1.2 The application of the immersed interface method

The immersed interface method has been successfully applied to several important interface problems.

In Chapter 2, we discussed the general elliptic interface problems in one, two and three dimensions. We have written a Fortran package (DIIM) for such problems in two dimensions (see Appendix A).

The immersed interface method for heat equations in two space dimensions with fixed interfaces is discussed in Chapter 3. When the singularities of the problem are due to singular sources or dipoles rather than the coefficients, we have derived the modified ADI method which only requires solving a set of tridiagonal systems at each time step.

In Chapter 4, we solved two dimensional Stokes flow with moving interfaces. We need to determine the pressure, velocity components in the x and y direction as well as the velocity of the interface. So it is a multi-variable and non-linear problem due to the moving interface. We derived the jump conditions for the pressure, the velocity and their normal derivatives by manipulating the two dimensional delta function and its derivative using distribution theory. A quasi-Newton method is used for solving the nonlinear equations when the fully implicit method is used.

In Chapter 5, we studied the immersed interface method for the one-dimensional heat equation with singular source terms and non-smooth solutions across some interface, which is also moving with time. Unlike the Stokes flow discussed in Chapter 4, the u_t term is present and the differential equation determining the movement of the interface has a more complicated form. Special care has to be taken to handle the discontinuity in u_t when the interface crosses the grid line.

6.2 Future research plan

6.2.1 Theoretical analysis for the immersed interface method

The immersed interface method discussed in this thesis seems to be very promising for interface problems. There are many possibilities for extending this thesis work.

Theoretically, we hope to develop more rigorous analysis of the convergence of the immersed interface method. For the problems discussed in this thesis, the local truncation errors near the interface are one order lower than that at most of the grid points on the whole region. It is commonly believed, and for some problems has been proved, that the convergence rate of the global error will not be affected. For different interface problems, it is still a challenge to find out the conditions under which the claim above is true.

Several computational issues have been raised for the immersed interface method. For example, when we solve the elliptic interface problem we may have a large sparse system of equations which may be neither symmetric nor diagonally dominant. Techniques to solve such systems efficiently are important for the success of the immersed interface method. It is not clear how to modify some of the state-of-the-art techniques such as the multi-grid method, GMRES, QMR, etc. for the linear system derived from our method.

Peskin's immersed boundary method is usually only first order accurate and will smooth the solution near the interface. However, the error distribution in the computed solution changes in a continuous manner. So his approach usually has better stability. In our approach, because the truncation error analysis is based on each grid point, we usually can get high order accuracy but the errors have random distribution near the interface. For the moving interface problem, such distributions can cause an aliasing instability. This is a commonly observed phenomenon for non-smooth high order methods (see [30]). In our numerical experiments we have used a filtering technique to control such an aliasing instability when it is necessary. It is desirable to have a high order method while the error has a smoother distribution. This will be another research project.

It is also worthwhile to combine other computational techniques such as adaptive or/and composite grids, variational principles, finite element methods etc. with the immersed interface method.

In this thesis, we use cubic splines to express the interface. This approach is simple and the cost is very low. Theoretically this approach can handle the singular interface when the interfaces develop cusps and spikes, or break or merge in two dimensions. However, it may be difficult to implement. Another possible approach is to use *level sets*, where the interface is modeled as the zero set of a function ϕ defined on the entire domain. Each time we need to solve an additional differential equation for the evolution of ϕ and must then determine the level set. So this approach costs more and sometimes it is difficult to extend the velocity of the interface to the whole domain. However, this approach makes it easier to handle cases when interfaces become singular as mentioned above. We want to combine the immersed interface method with the level set approach to solve some interface problems where the interface may develop singularities.

6.2.2 Application of the immersed interface method

There are a number of interface problems in computational fluid mechanics that we want to solve using the immersed interface method.

- *The Stefan problem in one and two dimensions.* Stefan problems are heat equations with discontinuous conductions and moving fronts.

$$\frac{\partial u_i}{\partial t} = \nabla \cdot (\beta_i \nabla u_i), \quad i = 1, 2, \cdots,$$

$$F_k \left(t, \Gamma, u_1, u_2, \cdots, \frac{\partial u_1}{\partial x_1}, \cdots, \right) = 0, \quad k = 1, 2, \cdots,$$

  where Γ is the interface. These problems have a lot of applications. In Chapter 5 we have discussed the simple case where we only have one variable and continuous coefficients.

- *Hele-Shaw flow.* In 1958, Saffman and Taylor performed experiments replacing a viscous fluid from between two closely spaced plates with a less viscous fluid. The shape of the interface exhibited a fingering phenomenon. The non-dimensional form of the governing equation is

$$\begin{aligned} \vec{u} &= -\beta \nabla p, \\ \nabla \cdot \vec{u} &= \phi, \end{aligned}$$

  where β is discontinuous coefficients, and ϕ is the source term, the jump conditions are

$$[p] = \tau\kappa; \qquad [\beta\, p_n] = 0,$$

  where τ is the surface tension and κ is the curvature of the interface. The difficulty in solving Hele-Shaw flow is that the interface is unstable.

- *ADI methods for elliptic and heat equations with discontinuous coefficients.*

- *Extending the work for Stokes flow with discontinuous coefficients such as density or viscosity.* This work allows modeling free boundaries between different liquids, such as the surface of a bubble. Surface tension provides the singular forcing term.

- *Full Navier-Stokes equations with moving interfaces.* This will require handling the nonlinear term and also the nonsmoothness of u_t as the boundary crosses a grid line.

BIBLIOGRAPHY

[1] R. E. Alcouffe, A. Brandt, J. E. Dendy, and J. W. Painter. The multi-grid method for the diffusion equation with strongly discontinuous coefficients. *SIAM J. Sci. Stat. Comput.*, 2:430–454, 1981.

[2] K. Aziz and A. Settari. *Petroleum Reservoir Simulation.* Applied Science Publishers Ltd, 1979.

[3] R. E. Bank. *PLTMG: A Software Package for Solving Elliptic Partial Differential Equations, User's Guide 6.0.* SIAM, 1990.

[4] J. B. Bell, C. N. Dawson, and G. R. Shubin. An unsplit, higher order Godunov method for scalar conservation laws in multiple dimensions. *J. Comput. Phys.*, 74:1–24, 1988.

[5] R. P. Beyer. A computational model of the cochlea using the immersed boundary method. *J. Comput. Phys.*, 98:145–162, 1992.

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

[7] C. Börgers and O. Widlund. On finite element domain imbedding methods. *SIAM J. Num. Anal.*, 27:963–978, 1990.

[8] F. Brakhagen and T. W. Fogwell. Multigrid methods in modelling porous media flow. In *The Mathematics of Oil Recovery*, pages 323–336. Clarendon Press, 1992.

[9] W.R. Briley and H. McDonald. On the structure and use of linearized block implicit schemes. *J. Comput. Phys.*, 34:54–73, 1980.

[10] B. L. Buzbee, F. W. Dorr, J. A. George, and G. H. Golub. The direct solution of the discrete Poisson equation on irregular grids. *SIAM J. Num. Anal.*, 8:722–736, 1971.

[11] Y.C. Chang, T. Y. Hou, B. Merriman, and S. Osher. A level set formulation of Eulerian interface capturing method for incompressible fluid flows, 1994.

[12] D. L. Chopp. Computing minimal surfaces via level set curvature flow. *J. Comput. Phys.*, 106:77–91, 1993.

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

[14] D.L. Dwoyer and F.C. Thames. Accuracy and stability of time-split finite-difference schemes. 5th AIAA Computational Fluid Dynamics Conference, Palo Alto, CA, 1981.

[15] edited by A. Fasano and M. Primicerio. *Free Boundary Problems: Theory and Applications, Vol. I.* Priman Advanced publishing Program, 1983.

[16] edited by D. G. Wilson, A.D. Solomon, and P.T. Boggs. *Moving Boundary Problems.* Academic Press, 1978.

[17] edited by Enrico Magenes. *Free Boundary Problems, Vol. II.* Proceedings of a seminar in Pavia, 1979.

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

[19] edited by J. R. Ockendon and W. R. Hodgkins. *Moving Boundary Problems in Heat Flow and Diffusion.* Clarendon Press, 1975.

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

[21] V. Faber, A. White, and R. Sweet. In extension of the implicit capacitance matrix algorithm for solving poisson equations on irregular regions. In R. Vichnevetsky and R. S. Stepleman, editors, *Advances in computer methods for partial differential equations*, pages 339–342. Pub. I M A C S, 1981.

[22] L. Fauci and C. S. Peskin. A computational model of aquatic animal locomotion. *J. Comput. Phys.*, 77:85–108, 1988.

[23] L. J. Fauci. Interaction of oscillating filaments – a computational study. *J. Comput. Phys.*, 86:294–313, 1990.

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

[25] A. L. Fogelson. A mathematical model and numerical method for studying platelet adhesion and aggregation during blood clotting. *J. Comput. Phys.*, 56:111–134, 1984.

[26] A. L. Fogelson and C. S. Peskin. Numerical solution of the three dimensional Stokes equations in the presence of suspended particles. In *Proc. SIAM Conf. Multi-phase Flow.* SIAM, June 1986.

[27] B. Fornberg and Rita Meyer-Spasche. A finite difference procedure for a class of free boundary problems. *J. Comput. Phys.*, 102:72–77, 1992.

[28] R. M. Furzeland. A comparative study of numerical methods for moving boundary problems. *J. Inst. Maths Applics*, 26:411–429, 1980.

[29] P. E. Gill, W. Murray, and M. H. Wright. *Practical optimization.* Academic Press, 1981.

[30] T. Y. Hou, J. S. Lowengrub, and M. J. Shelley. Removing the stiffness from interfacial flows with surface tension. preprint, 1993.

[31] R. J. LeVeque and Z. Li. The immersed interface method for elliptic equations with discontinuous coefficients and singular sources. preprint, available by anonymous ftp from amath.washington.edu in the file pub/leveque/rjl-li:elliptic.ps, 1992.

[32] Z. Li and A. Mayo. ADI methods for heat equations with discontinuities along an arbitrary interface. Proceedings of the Vancouver conference *Mathematics of Computation 1943–1993*, editor: W. Gautschi, 1994.

[33] R. J. MacKinnon and G. F. Carey. Analysis of material interface discontinuities and superconvergent fluxes in finite difference theory. *J. Comput. Phys.*, 75:151–167, 1988.

[34] A. Mayo. On the rapid evaluation of heat potentials on general regions. IBM Technical report 14305.

[35] A. Mayo. The fast solution of Poisson's and the biharmonic equations on irregular regions. *SIAM J. Num. Anal.*, 21:285–299, 1984.

[36] A. Mayo. Rapid, high order accurate evaluation of volume integrals of potential theory. *J. Comput. Phys.*, 100:236, 1992.

[37] A. Mayo. Rapid, accurate methods for the solution of the stokes problem in the presence of an immersed boundary. IBM Research Report, 1993.

[38] A. Mayo and A. Greenbaum. Fast parallel iterative solution of Poisson's and the biharmonic equations on irregular regions. *SIAM J. Sci. Stat. Comput.*, 13:101–118, 1992.

[39] A.R. Mitchell. Computational methods in partial differential equations. *J. Soc. Indust. Appl. Math.*, 3:28–41, 1955.

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

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

[42] C. S. Peskin. Lectures on mathematical aspects of physiology. *Lectures in Appl. Math.*, 19:69–107, 1981.

[43] C. S. Peskin and D. M. McQueen. A three-dimensional computational method for blood flow in the heart: (i) immersed elastic fibers in a viscous incompressible fluid. *J. Comput. Phys.*, 81:372–405, 1989.

[44] C. S. Peskin and B. F. Printz. Improved volume conservation in the computation of flows with immersed elastic boundaries. *J. Comput. Phys.*, 105:33–46, 1993.

[45] W. Proskurowski and O. Widlund. On the numerical solution of Helmholtz's equation by the capacitance matrix method. *Math. Comp.*, 30:433–468, 1976.

[46] G. R. Shubin and J. B. Bell. An analysis of the grid orientation effect in numerical simulation of miscible displacement. *Comp. Meth. Appl. Mech. Eng.*, 47:47–71, 1984.

[47] J. Stoer and R. Bulirsch. *Introduction to Numerical Analysis*. Springer-Verlag, 1980.

[48] Sussman-Smereka-Osher. A level set approach for computing solutions to incompressible two-phase flow. UCLA Computational ans Applied Mathematics Report 93-18, 1993.

[49] Paul N. Swarztrauber. Fast poisson solver. In Gene H. Golub, editor, *Studies in Numerical Analysis*, volume 24, pages 319–370. The Mathematical Association of America, 1984.

[50] A. N. Tikhonov and A. A. Samarskii. Homogeneous difference schemes. *USSR Comput. Math. and Math. Phys.*, 1:5–67, 1962.

[51] C. Tu and C. S. Peskin. Stability and instability in the computation of flows with moving immersed boundaries: a comparison of three methods. *SIAM J. Sci. Stat. Comput.*, 13:1361–1376, 1992.

[52] Yanenko. *The Method of Fractional Steps*. Springer-Verlag, 1971.

[53] J. Zhu and J.A. Sethian. Projection methods couples to level set interface technique. *J. Comput. Phys.*, 102:128–138, 1992.

Appendix A

PROLOGUE OF THE PACKAGE DIIM–A FORTRAN PACKAGE FOR SOLVING ELLIPTIC INTERFACE PROBLEMS

```
      subroutine diim(m,n,n1,ki,infoj,ijump,a,b,c,d,bin,bout,amega,
     1                                f,fin,fout,uj,unj,u,u1,u2,xyjump,tol2)

C*********************************************************************************
C                                                                                 C
C              IMMERSED INTERFACE METHOD PACKAGE                  C
C                                                                                 C
C*********************************************************************************
C
C   diim  solves the following interface problems in double precision for an
C         arbitrary interface
C
C               \beta ( u_xx + u_yy ) = f       a <= x <= b
C                                               c <= y <= d
C        where \beta, f, u, u_n, ... may be discontinuous across some
C        general closed interface which is determined by the user supplied
C        control points (x1(i),y1(i)), i = 1, ...,n1.
C        The interface conditions are jump condition in the solution
C        [ u ] and in the flux [ \beta u_n ] which are supplied by
C        the user on the control points. The boundary condition on the
C        rectangular is Dirichlet condition which should be specified in
C        u(i,0), u(i,n), i = 0,...m, and u(0,j),u(m,j), j = 0,1,...,n.
C.................................................................................
C
C   On INPUTS:
C
C     m, n    The numbers of grids on x and y direction.
C     n1      The numbers of control points on the interface.
C     infoj   infoj(5) is the flag for the different problems.
C        infoj(1) = 0:  Regular Poisson problem on the rectangular.
C                 = 1:  Interface problem.
C        infoj(2) = 0:  [ u ] = 0, i.e. The solution is continuous.
C                 = 1:  There is a jump in the solution.
C        infoj(3) = 0:  [\beta u_n ] = 0
C                 = 1:  [\beta u_n ] is not equal zero.
C                 = 2:  [\beta] = 0, So the coefficients are standard central
C                       formula.
C        infoj(4) = 0:  The right hand side f(x,y) is continuous.
C                 = 1:  f is not continuous but we know the values
C                       of f (at control points) from the both
C                       sides of the interface.
C                 = 2:  f is not continuous and only given at grid points f(i,j).
C                       We call subroutine spreaf.f to determine the values
```

```
C                       of f (at control points) from the both
C                       sides of the interface.
C     infoj(5)       The choice for iterative method. When [\beta] = 0,
C                    i.e. infoj(3) = 2, we use the optimal parameter,
C                    so amega is not needed.
C        infoj(5) = 1:  Use sor iteration.i
C        infoj(5) = 2:  Use line sor (lsor) iteration.
C        infoj(5) = 3:  Use line ADI iteration, recommend only when [\beta] = 0.
C
C     a,b,c,d:       The end points for the rectangle: a <= x <= b;
C                    c <= y <= d. We suppose (b-a)/m = (d-c)/n.
C
C     bin, bout      The coefficients of \beta inside and outside
C                    the interface respectively.
C     amega          Over-relaxation parameter for lsor or sor methods.
C     x1(n1+1),y1(n1+1)  The coordinates of the control points.
C                        x1(1) = x1(n1+1) and y1(1) = y1(n1+1).
C     f(m,n)         The right hand side in discrete form.
C     fin(n1+1),fout(n1+1)    If infoj(4) = 1, we need the values of
C                    f(x,y) at the control points from inside and
C                    outside the interface. fin(1) = fin(n1+1) and so on.
C     uj(n1+1)       The jump in the solution on the control points when
C                    infoj(2) = 1. uj(1) = uj(n1+1).
C     unj(n1+1)      The jump in the flux [\beta u_n] on the control
C                    control points when infoj(3) = 1. unj(1) = unj(n1+1).
C     tol2           The tolerance for the lsor iteration.
C
C.....................................................................
C
C   On OUTPUTS:
C
C     u2(0:m,0:n)    The computed solution at grid points.
C
C     ki             The number of iterations used in solving the linear system
C.....................................................................
C
C   WORKING SPACES:
C
C     ijump   ijump(m,n,3). ijump contains the index information of the
C             grid points (see subroutine index for the detail).
C
C     xyjump  xyjump(m,n,9). see subroutine irreg for the detail.
C
C     u, u1, u2   u(0:m+1,0:n+1),u1(0:m+1,0:n+1),u2(0:m+1,0:n+1)
C             working spaces for different iterative methods.
C
C.....................................................................
C
C   LIBRARY CALLED:     LINPACK
C.....................................................................
C
C   Subroutines called:
```

```
C
C     splcl:   The periodic spline interpolation package.
C
C     spread:  Determining the approximation of inner and outside limit of a
C        function at the control points of the interface. So later we
C        can use spline interpolation.
C
C     index    Indexing grid points and record the other informations for
C              irregular grid points.
C
C     irreg    Determine the coefficients for the difference scheme at
C              irregular grid points.
C
C     adi      adi iterative method in double precision.
C
C     sor      sor  iterative method in double precision.
C
C     lsor:    Line sor  iterative method in double precision.
C
C     rootp3   rootp3 finder for cubic polynomial.
C
C     blas:    Basic linear algebra computation routines:
C
C.....................................................................
C
C   Written by Zhilin Li, February, 1993
C
C*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
C
C                      END OF DOCUMENTATION FOR DIIM
C
C*********************************************************************************
```

Appendix B

PROLOGUE OF THE PACKAGE PPACK – A FORTRAN PACKAGE FOR SOLVING THE POISSON EQUATIONS ON IRREGULAR REGIONS

```
C subroutine ppack(m,n,n1,info,a,b,c,d,x,y,x1,y1,ijump,xyjump,
C    1                                    ub,fub,rhsf,f,u,u1,u2,scale,tol,tu)

C*************************************************************************
C                                                                        C
C              POISSON PACKAGE PPACK FOR IRREGULAR REGION                C
C                                                                        C
C*************************************************************************
C
C   ppack solves the POISSON equation u_xx + u_yy = f(x,y) with Dirichlet
C         boundary condition on arbitrary closed region in double precision.
C
C         This package uses a smallest rectangular to enclose the domain and
C         uses the periodic spline interpolation to express the boundary. it
C         uses a uniform grid so the step size is the same both in x and y
C         direction.
C.........................................................................
C
C   On INPUTS:
C
C     info info(4) are flags for different input options.
C
C        info(1) = 0:  Rectangular problem.
C                = 1:  The region is irregular.
C
C        info(2) = 0:  The boundary condition is supplied by function fub(x,y).
C                = 1:  The boundary condition is given discretely at control points.
C
C        info(3) = 0:  The right hand side f(x,y) is supplied by function rhsf(x,y).
C                = 1:  The right hand side is given discretely at grid points.
C
C        info(4) = 1:  use sor iteration.
C                = 2:  use lsor iteration.
C                = 3:  use adi iteration.
C
C     m, n    The numbers of grid points on x and y directions. If the region is
C             irregular (i.e. not a rectangle), and the maximum extent of
C             the domain in x/y direction is longer than that in y/x
C             direction, then we should take (m >= n)/(n >= m). Be make sure
C             that the shorter integer is taken so that the domain is
```

```
C             indeed inclosed in the rectangular. The safest way is to take
C             m = n. This only affects the speed of the iterative methods.
C
C     a,b,c,d: If the domain is rectangular (info(0) = 0), a and b (a<b)
C             are two ends in x direction, c and d (c< d) are two ends
C             in y direction. Now the fastest method would be adi iteration
C       (info(4) = 3) with scale = 1.0.
C
C     f       f(m,n) Not needed when external function rhsf is provided.
C       The discrete right hand side on the grid points. For
C             irregular region only needed inside the region (see the
C             comments below).
C
C     u       u(0:m+1,0:n+1) If the region is rectangular (info(0) = 0),
C             u(0,j), u(m+1,j) j = 0, 1, ..., n+1 are the boundary
C             condition on x = a and x = b. And u(i,0), u(i,n+1) are
C             the boundary condition on y = c and y = d, i = 0, ..., m+1.
C             Not needed when external function fub is provided
C       ( i.e. info(2) = 0).
C
C     scale   Needed when info(4) = 3. The scale parameter for adi iteration.
C            For the rectangular region (info(1) = 0), scale = 1.
C             For the irregular region, scale can varies from problems .
C             (see the comments below).
C
C     tol     The tolerance for the iterative method.
C
C     fub     Needed when info(2) = 0. The fub must be declared in an
C             external statement in the user calling program, and should
C             be written as follows:
C
C              double precision function fub(x,y)
C              double precision x,y
C              fub = ....
C
C     rhsf    Needed when info(3) = 0. The rhsf must be declared in an
C             external statement in the user calling program, and should
C             be written as follows:
C
C              double precision function rhsf(x,y)
C              double precision x,y
C              rhsf = ....
C
C  The following inputs are for irregular region:
C
C     n1      The numbers of control points on the irregular boundary. If
C             the boundary condition is given in exact form fub(x,y), we
C             can take n1 close to max(m,n). But if the boundary is not
C             smooth or has complicated shape, and the boundary condition
C             is only given on the control points and changes rapidly on
C             the boundary we should take enough points so the spline
C             interpolation for the boundary condition gives appropriate
```

```
C             accuracy. The magnitude of n1 has little effect on the
C             computation cost.
C
C     x1,y1   x1(n1+1), y1(n1+1) The coordinates of control points for
C             the boundary. x1(n1+1) = x1(1); y1(n1+1) = y1(1). Not
C             needed when info(0) = 0.
C
C     ub      ub(n1+1) The Dirichlet boundary condition on the control
C             points. Not needed when info(2) = 0. ub(1) = ub(n1+1).
C
C.....................................................................
C
C   On OUTPUTS:
C
C     u2(0:m,0:n)    u2(i,j) i = 1, ..., m; j = 1, ..., n are the
C      computed solution at grid points.
C
C.....................................................................C
C
C   WORKING SPACES:
C
C     x,y     x(m), y(n) After calling regset when info(1) = 0 or extreme
C             when info(1) = 1 they contain grid lines in x and y direction.
C
C     ijump   ijump(m,n,4). ijump contains the index information of the
C             grid points (see subroutine index for the detail).
C
C     xyjump  xyjump(m,n,8). see subroutine irreg for the detail.
C
C     u, u1, u2   u(0:m+1,0:n+1),u1(0:m+1,0:n+1),u2(0:m+1,0:n+1)
C             working spaces for different iterative methods.
C
C.....................................................................
C
C   LIBRARY CALLED:     LINPACK
C
C.....................................................................
C
C   Subroutines called:
C
C     regset: If the domain is rectangular (info(1) = 0). Set up process.
C
C     splcl:  The periodic spline interpolation package.
C
C     extreme: The subroutine determine the extreme values for each spline
C              interval and return the smallest rectangular region and
C              step size h.
C
C     index    Indexing grid points and record the other informations for
C              irregular grid points.
C
C     irreg    Determine the coefficients for the difference scheme at
```

```
C              irregular grid points.
C
C     dadi     adi iterative method in double precision.
C
C     sor      sor  iterative method in double precision.
C
C     lsor:    Line sor  iterative method in double precision.
C
C     rootp3   rootp3 finder for cubic polynomial.
C
C     blas:    Basic linear algebra computation routines:
C              copymat1; checkp; dgltsl, copyvec
C
C.....................................................................
C
C   COMMENTS:
C
C     1. The ppack is very easily generalized to a more general elliptic problem:
C
C         a(x,y)*(u_xx + u_yy) + b(x)*u_x + c(x,y)*u_y + d(x,y)*u = f(x,y)
C
C        with Dirichlet boundary condition. The convergence speed is
C        pretty much the same.
C
C     2. The subroutine ppack.f is the collection of several other routines.
C        The user can get different information by other routines.
C        For example if you want to get N points (more than the
C        control points on the spline), you can do
C
C          open(50,file='d.m',status='unknown')
C          ds = hs1(n1+1)/N
C          do i=1,N+1
C             s = (i-1.0d0)*ds
C             call splval(n1,s,x1,y1,cox1,coy1,hs1,hs2,x3,y3,info)
C             write(50,*)x3,y3
C          enddo
C
C        after calling splcl (the spline interpolation package).
C        Then the file d.m will contains the information of the coordinates
C        of the points on the spline.
C
C        The other example is that when the right hand side f is given in
C        discrete form (info(3) = 1), we only need to specify those f(i,j)
C        inside the domain instead of the entire rectangular region. After
C        calling index subroutine we are able to know which grid points
C        are inside the domain (ijump(i,j,1) .eq. 2 .or. 1) and
C        (ijump(i,j,3) .eq. 2 .or. 1). We can specify f(i,j) accordingly.
C
C     3. Generally all three iterative methods sor, lsor and adi  converge
C        pretty fast. For sor and lsor methods, they converge faster than
C        the methods applied to the same rectangular region (less
C        iterations). Besides these methods only compute the iteration
```

```
C       inside the region so they are cheaper per iteration than the
C       methods used for the rectangular region. The sor method seems to
C       be better than lsor method.
C
C       Generally adi method is a faster solver for POISSON problems on
C       rectangular regions. In our approach, the adi method is still
C       faster than sor and lsor method if we choose right parameters.
C       Usually the parameters will vary for different geometries. But
C       with the parameter chosen from the smallest rectangular and
C       multiply it with some scale, we find that the right scale is
C       very close for different mesh size corresponding to the same
C       geometry. So we can test the best scale for space mesh size and
C       then increase it a little bit as the mesh size get finer.
C
C
C*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
C
C           END OF DOCUMENTATION FOR PPACK
C
C-------------------------------------------------------------------------------
```