ABSTRACT

ELAMAMI, ELGADDAFI ELSAAD. Augmented Strategies for 3D Elliptic Interface Problems with Piecewise Constant Discontinuous Coefficients. (Under the direction of Dr. Zhilin Li.)

The augmented strategies for three-dimensional interface problems are reviewed in this thesis. A fast second-order accurate iterative method is proposed for the elliptic interface problems in a cubic domain in 3D using Cartesian grids for three dimensional elliptic interface problems in which the coefficients, the source term, the solution and its normal flux may be discontinuous (may have jumps) across an irregular interface. The idea in our approach is to precondition the differential equation before applying the immersed interface IIM method proposed by LeVeque and Li [SIAM J. Numer. Anal., 31(1994), pp. 1019-1044]. In order to take advantage of fast Poisson solvers on a cubic domain, an intermediate unknown function of co-dimension two, the jump in the normal derivative across the interface, is introduced. Our discretization is equivalent to using a second-order difference scheme for a corresponding Poisson equation in the domain, and a second-order discretization for a Neumann-like interface condition. Thus second-order accuracy is guaranteed. Weighted least square method is also proposed to approximate interface quantities from a grid function.

Numerical experiments are provided and analyzed. The number of iterations in solving the Schur complement system appears to be independent of both the jump in the coefficient and the mesh size. The method is designed for interface problems with piecewise constant coefficient. The method is based on the fast immersed interface method and a fast 3D Poisson solver. There are at least two reasons to use augmented strategies. The first one is to get faster algorithms, particularly, to take advantages of existing fast solvers. The second reason is that, for some interface problems, an augmented approach may be the only way to derive an accurate algorithm. Using an augmented approach, one or several quantities of co-dimension two are introduced. The GMRES iterative method is employed to solve the Schur complement system derived from the discretization and is often used to solve the augmented variable(s) that are only defined along the interface or the irregular boundary. Several examples of the augmented method are provided in this thesis. An application of the method is that it has been modified to solve Poisson equations on irregular domains.

Augmented Strategies for 3D Elliptic Interface Problems with Piecewise Constant Discontinuous Coefficients

by
Elgaddafi Elsaad Elamami

A dissertation submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Mathematics

Raleigh, North Carolina

2013

APPROVED BY:

Zhilin Li Chair of Advisory Committee	Alina Chertock
Mansoor Haider	Khaled Harfoush

DEDICATION

To my sons Alaa, Wissam, and my little daughter Jana. To my wife Ranya and my parents for their support.

BIOGRAPHY

Elgaddafi Elsaad Elamami was born in Benghazi, Libya, in 1976. He graduated from high school in 1994. He spent four years for his undergraduate study and earned his Bachelors of Science in Mathematics in 2000 from University of Benghazi, formerly known as Garyounis University. He worked as mathematics teacher in Al-Fateh Center for Talented Students for five years. He began his graduate work at North Carolina State University in January 2010 with a Masters degree in mathematics that earned from University of Benghazi in 2006. He earned his Masters of Science in Applied Mathematics from North Carolina State University.

ACKNOWLEDGMENTS

First and foremost I would like to express my gratitude to my advisor, Professor Zhilin Li, for his continued support, patience and thoughtful guidance. His dedication and insight has encouraged me to continually improve as a mathematician. I am not sure what I was expecting the first day I walked into his office, but I definitely learned more than I was expecting.

I would like to thank my advisory committee members Professor Alina Chertock, Professor Mansoor Haider, and Professor Khaled Harfoush for their advice.

I am grateful to my family and wife for their support, encouragement, and patience. Also, I would also like to thank Ms Denise Seabrooks and Professor Erniest Stitzinger for their academic support. And Matt Haught for his technical support.

Last but not least, I would like to thank my friends and fellow graduate students, especially Ms. Changjuan Zhang, Sidong Zhang, Peng Song, and my wife Ranya for encouragement and advice.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

1.1 Introduction

In this thesis, we develop a second order fast algorithm to solve three-dimensional elliptic equations with piecewise constant discontinuous coefficients on a cubic domain. The problem can be described as follows:

Let Ω be a cubic domain in the space $\mathbb{R}^3$. Consider the following elliptic problem of the form:

$$\nabla.\left(\beta(x,y,z)\nabla u(x,y,z)\right) + ku(x,y,z) = f(x,y,z), \quad (x,y,z) \in \Omega\backslash\Gamma, \tag{1.1a}$$

$$[u] = w(s), \qquad [\beta u_n] = v(s), \quad \text{on } \Gamma, \tag{1.1b}$$

with a specified boundary condition on $\partial\Omega,$ where $\Gamma(s)$ is an interface that divides the domain Ω into two sub-domains, $\Omega^+$and, Ω^-, and $u_n = \nabla u.n$ is the normal derivative along the unit normal direction n, s is the arc length parameterization of Γ . We use [.] to represent the jump of a quantity across the interface Γ. The coefficients β, k, and the source term f may be discontinuous across the interface Γ. We assume that $\beta(x,y,z)$ has a constant value in each sub-domain, i.e.,

$$\beta(x,y,z) = \begin{cases} \beta^+, & in\ \Omega^+, \\ \beta^-, & in\ \Omega^-. \end{cases} \tag{1.2}$$

If $\beta^+ = \beta^- = \beta$ is a constant, then we have a Poisson equations $\Delta u = f/\beta$ with the source distributions along the interface that corresponds to the jumps in the solution and the flux. The finite difference method obtained from the immersed interface method [26,27,30] yields the standard discrete Laplacian plus some correction terms to the right hand side. Therefore, a fast Poisson solver, for example, the Fishpack [2], can be used to solve the discrete system of equations. If $\beta^+ \neq \beta^-$, we can not divide the coefficient β from the flux jump condition. The motivation is to introduce an augmented variable so that we can take advantage of fast

Poisson solver for the interface problem with only singular sources. Our approach is based on finite difference method. It is of second order accuracy and the algorithm is fast, requiring only $O(N^3 log N^3)$ arithmetic operations for a mesh of N grid points.

1.2 Some application examples

Equation (1.1) is called Poisson problem and such problems arises in several fields of physical applications in material science, some mathematical biology problems modeled by PDEs, and fluid dynamics when two distinct materials or fluids with different conductivities or densities are interfaced. Classical examples can be found in the so-called heat equation. For example, u may represent the temperature distribution in a material with heat conductivity β and a heat source f. For composite materials, the coefficient β is discontinuous across the interface between two different materials. For example, the temperature u should be continuous, which means $[u] = 0$ across the interface, where [.] denotes the jump in a quantity across the interface. The heat flux βu_n should also be continuous across any interface, i.e., $[\beta u_n] = 0$, if no heat source is present there, where n denotes the unit normal vector at the interface. If β is discontinuous, then there must be a jump in the normal derivative u_n. Mathematically, the temperature u can be modeled as follows

$$u_t = \nabla.(\beta \nabla u), \qquad \text{in} \quad \Omega,$$

with boundary conditions on $\partial\Omega$, initial conditions, and jump conditions on Γ: $[u] = 0$, $[\beta u_n] = 0$. The coefficient β is given as above, where Ω^+ and Ω^- denote the two sides of the interface, and Γ denotes the interface between Ω^+ and Ω^-. More physical applications of interface problems can be found in Hele-Shaw flow [21,22,47], two-phase flow [6,11,50], drop spreading [23] and electro-migration of voids [8,34,40], etc.

1.3 Elliptic interface problems

This thesis is concerned with numerical analysis of elliptic interface problems in three-dimensional space. Let Ω be a simple convex domain subset of $\mathbb{R}^3$ which is divided into two sub-domains by an irregular interface Γ such that $\Omega = \Omega^+ \cup \Omega^-$. Consider the elliptic equation (1.1) and (1.2a-b).

Assume that the coefficient β and source term f may be discontinuous across the interface Γ, i.e.,

$$\beta = \begin{cases} \beta^+, & in \;\; \Omega^+, \\ \beta^-, & in \;\; \Omega^-. \end{cases}$$

$$f = \begin{cases} f^+, & in \;\; \Omega^+, \\ f^-, & in \;\; \Omega^-. \end{cases}$$

See Figure 1.1 for illustrations.

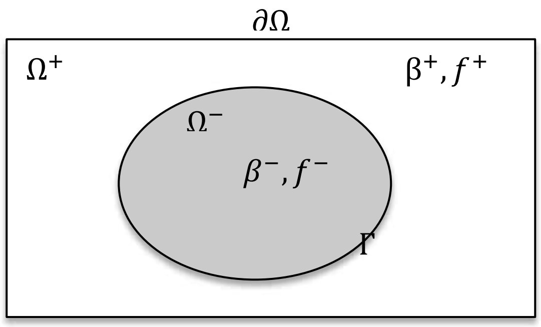

Figure 1.1: A cubic domain Ω with an immersed interface Γ. The coefficients β and the source term f may have jumps across the interface.

It is crucial for our approach that we have enough a priori knowledge about some interface conditions. Equations (1.2a) and (1.2b) can be derived either by physical reasoning or directly from the differential equation itself.

1.4 Overview of numerical methods for the elliptic equations

Solving interface problems efficiently and accurately has been a challenge because of many discontinuities or non-smoothness of their solution across the interface or discontinuities in the coefficients and singular source terms across the interfaces. Over years, interface problem have attracted a lot of attention from numerical analysts, and many numerical methods have been developed for the interface problems such as:

- Peskin's immersed boundary method (IBM), see [6,16,43,44];
- Smoothing methods for discontinuous coefficients, see [4,49,51];
- The immersed interface method (IIM), see [26,27];
- Harmonic averaging for discontinuous coefficients, see [4,49];
- Ghost fluid methods (GFM), see [17,18,19];
- Finite elements methods (FEMs); with body-fitted meshes;
- Immersed finite element method (IFEM), see [3,24].

Some methods (immersed boundary methods, ghost fluid methods) are easy to implement, but only achieve first order accuracy. Others (immersed interface method, finite element methods with body-fitted meshes) achieve second order accuracy, but they are not easy to implement. We refer the reader to [26,27,30,32,33] for an incomplete review of Cartesian grid methods for interface problems. Moreover, most of the above numerical methods can be second order accurate in L_1or L_2 norm, but not in L_∞ norm. And it seems that only the IIM method can guarantee a second order accurate solution in L_∞ norm for interface problems.

1.5 Interface expression

To solve interface problems numerically, we need the information about the interface such as the position, tangential and normal directions, and sometimes curvatures as well. We choose to use the level set approach in this thesis since we focus on three-dimensional interface problems. The level set approach was introduced by Osher and Sethian in [42] and has been applied for many moving interface problems (e.g., [9,10,21,32,50,52]) since then. In this approach the interface is modeled as the zero level set of a smooth function φ defined on the entire physical domain. The interface is then moved by updating values of φ through non-linear Hamilton-Jacobi equation of the level set function. By the narrow band approach [10], the Hamilton-Jacobi equation is solved only over a computational tube surrounding the interface, and only the values of φ within the narrow tube are updated.

Furthermore, this approach 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.

1.6 Our strategy

The main idea in our approach is to precondition the partial differential equation PDE (1.1),(1.2a) and (1.2b) before we apply the immersed interface method IIM proposed by LeVeque and Li [26], and in order to take advantage of fast Poisson solvers on a cubic domain, an intermediate unknown function of co-dimension two which is the jump in the normal derivative is introduced.

1.6.1 The sketch of our method

Our method is based on an approach that involves the following steps:

- We precondition (1.1),(1.2a-b) to get an equivalent problem before using the IIM.
- We use the IIM idea to discretize the equivalent problem and derive the Schur complement system.
- We discuss the weighted least squares approach to approximate from the grid function.
- We propose an efficient preconditioner for the Schur complement system.

1.6.2 The augmented variable and augmented equations

There are more than one way to introduce an augmented variable. Since the original partial differential equation PDE (1.1),(1.2a-b) can be written as a Poisson equation in each subdomain after we divide β from the original PDE excluding the interface Γ, it is natural to introduce $[u_n]$ as the augmented variable.

Consider the solution set $u_g(x,y,z)$ of the following problem as a functional of $g(s)$,

$$\begin{aligned} \Delta u &= f/\beta^+, \qquad in\ \Omega^+, \\ \Delta u &= f/\beta^+, \qquad in\ \Omega^-. \end{aligned} \tag{1.3}$$

with the same boundary condition on $\partial\Omega$ as in the original problem (1.1). Let the solution of (1.1) be $u^*(x,y,z)$, and define

$$g^*(s) = [u_n^*](s), \tag{1.4}$$

along the interface Γ. Then $u^*(x,y,z)$ satisfies the PDE and the jump conditions in (1.3) with $g(s) = g^*$. In other words, $u_{g^*}(x,y,z) = u^*(x,y,z)$, and

$$\left[\beta \frac{\partial u_{g^*}}{\partial n}\right] = v(s), \tag{1.5}$$

is satisfied. The expression (1.5) is called the augmented equation. Therefore, solving the original problem (1.1),(1.2a-b) is equivalent to finding the corresponding $g^*(s)$ that satisfies

(1.3) and (1.5). Note that $g^*(s)$ is only defined along the interface, so it is one-dimension lower than that of $u(x, y, z)$. If we are given $[u_n] = g(s)$, then it is easy to solve (1.3) using the IIM since only correction terms at irregular grid points are needed to add to the right hand side of the finite difference equations. A fast Poisson solver such as the FFT then can be used. The cost in solving (1.3) is just a little more than that in solving a regular Poisson equation on the cubic domain as in Figure (1.1) with a smooth solution.

As mentioned earlier, it requires only $O(N^3 logN^3)$ arithmetic operations for a mesh of N grid points.

1.7 The outline of the thesis

In chapter 2, we will give the detailed description on how to apply the immersed interface method IIM to solve 3D elliptic interface problems and introduce the interface relations of the problem which will be utilized in the derivation of our augmented approach. The main task of this thesis is to develop a second order fast algorithm for 3D elliptic interface problems with piecewise constant but discontinuous coefficients which is established mainly in chapter 3. In order to use fast Poisson solvers, we introduce the jump in the normal derivative $[u_n]$ as an unknown variable (augmented variable). Then the GMRES method is employed to solve a Schure complement system for $[u_n]$. Moreover, this method has no significant storage need, $\beta^+$and $\beta^-$are inputs, and it gives us $[u_n]$ in addition to the solution u. In chapter 4, the fast Poisson solver for piecewise constant coefficients is investigated by some numerical experiments.

In chapter 5, as a part of this thesis, we will investigate some applications to the proposed fast augmented IIM method by using the domain embedding technique to solve interior or exterior Poisson equations with Dirichlet boundary conditions on irregular domain. The main idea is to embed the irregular domain into cubic domains, and treat them as interface problems, and then use the fast algorithm to solve them. Finally, in chapter 6 we summarize the contributions we have achieved and discuss several possible future research topics.

Chapter 2

The IIM for elliptic interface problems in 3D

2.1 Introduction

Let Ω be a convex domain in $\mathbb{R}^3$ and Γ be an arbitrary piecewise smooth surface in Ω. The interface divides Ω into two sub-domains Ω^+ and Ω^- and therefore $\Omega = \Omega^+ \cup \Omega^- \cup \Gamma$. See figure (1.1). In this chapter, we discuss the IIM for the elliptic interface problems of the form

$$\nabla.(\beta \nabla u(x,y,z)) + ku(x,y,z) = f(x,y,z), \qquad (x,y,z) \in \Omega \backslash \Gamma, \tag{2.1}$$

in three dimensions in a region Ω with a boundary condition on $\partial\Omega$, where all the coefficients β, k ,and the source term f may be discontinuous across the interface Γ , which is a surface S: $x = x(s_1, s_2), y = y(s_1, s_2), z = z(s_1, s_2)$. To make the problem well-posed, we assume that we have knowledge of the jump conditions in the solution and the flux,

$$[u] = w(s_1, s_2), \qquad [\beta u_n] = v(s_1, s_2), \tag{2.2}$$

where w and v are two known functions defined only along the interface Γ.

In this chapter, we will introduce the interface relations of the problem of that will be used in the derivation of our method later in chapter 3. Now, let us start to present a complete set of interface relations up to second order derivative by differentiating the jumps along the interface Γ, and making use of the original partial differential equation (PDE) (2.1). Since the flux jump condition $[\beta u_n]$ is given in the normal direction of the interface, it is convenient to use local coordinates in the normal and tangential directions.

2.2 A local coordinate system in 3D

Given a point (X, Y, Z) on the interface Γ, let ξ (with $\|\xi\| = 1$) be the normal direction of Γ pointing to a specific side, say the "+'' side. Let η and τ be two orthogonal unit vectors tangential to Γ, then a local coordinate transformation is given by

$$
\begin{aligned}
\xi &= (x - X)\alpha_{x\xi} + (y - Y)\alpha_{y\xi} + (z - Z)\alpha_{z\xi}, \\
\eta &= (x - X)\alpha_{x\eta} + (y - Y)\alpha_{y\eta} + (z - Z)\alpha_{z\eta}, \\
\tau &= (x - X)\alpha_{x\tau} + (y - Y)\alpha_{y\tau} + (z - Z)\alpha_{Z\tau},
\end{aligned}
\tag{2.3}
$$

where $\alpha_{x\xi}$ represents the directional cosine between the *x*–axis and ξ, and so forth. See figure (2.1) for an illustration. Note that the choice of the two orthogonal tangential vectors is not unique.

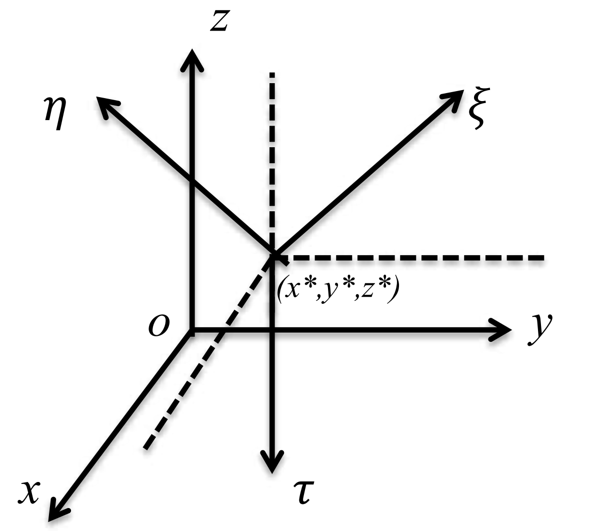

Figure 2.1: A sketch of local coordinates transformation in 3D.

The three-dimensional coordinate transformation above can also be written in a matrix-vector form. Define the local transformation matrix as

$$A = \begin{pmatrix} \alpha_{x\xi} & \alpha_{y\xi} & \alpha_{z\xi} \\ \alpha_{x\eta} & \alpha_{y\eta} & \alpha_{z\eta} \\ \alpha_{x\tau} & \alpha_{y\tau} & \alpha_{z\tau} \end{pmatrix}, \tag{2.4}$$

Then we have

$$\begin{pmatrix} \xi \\ \eta \\ \tau \end{pmatrix} = A \begin{pmatrix} x - X \\ y - Y \\ z - Z \end{pmatrix}. \tag{2.5}$$

Also, for any differentiable function $p(x, y, z)$ we have

$$\begin{pmatrix} \bar{p}_\xi \\ \bar{p}_\eta \\ \bar{p}_\tau \end{pmatrix} = A \begin{pmatrix} p_x \\ p_y \\ p_z \end{pmatrix}, \tag{2.6}$$

$$\begin{pmatrix} \bar{p}_{\xi\xi} & \bar{p}_{\xi\eta} & \bar{p}_{\xi\tau} \\ \bar{p}_{\eta\xi} & \bar{p}_{\eta\eta} & \bar{p}_{\eta\tau} \\ \bar{p}_{\tau\xi} & \bar{p}_{\tau\eta} & \bar{p}_{\tau\tau} \end{pmatrix} = A \begin{pmatrix} p_{xx} & p_{xy} & p_{xz} \\ p_{yx} & p_{yy} & p_{yz} \\ p_{zx} & p_{zy} & p_{zz} \end{pmatrix} \tag{2.7}$$

where $p(\xi, \eta, \tau) = p(x, y, z)$ and A^T is the transpose of A. It is easy to verify that $A^T A = I$, the identity matrix. For two dimensional problems, the matrix-vector relations under the local coordinates can be found in [29]. Note that under the local coordinates transformation (2.3), the PDE (2.1) is invariant. Therefore we will drop the bars for simplicity.

2.3 Interface relations for elliptic interface problems in 3D

Using the superscript " $+$ " or " $-$ " to denote the limiting values of a function from the Ω^+ side or the Ω^- side of the interface, respectively, we can write the limiting differential equation from the " $-$ " side as

$$\beta^-\left(u^-_{\xi\xi} + u^-_{\eta\eta} + u^-_{\tau\tau}\right) + \beta^-_\xi u^-_\xi + \beta^-_\eta u^-_\eta + \beta^-_\tau u^-_\tau - k^- u^- - f^- = 0 \tag{2.8}$$

Under the local coordinate system $\xi - \eta - \tau$, the interface Γ can be expressed as

$$\xi = \chi(\eta, \tau), \quad with \ \ \chi(0,0) = 0, \ \chi_\eta(0,0) = 0, \ \chi_\tau(0,0) = 0. \tag{2.9}$$

From the jump condition (2.2) and the differential equation (2.1), we can derive more interface relations, which are summarized in the following theorem.

Theorem 2.1. Assume that the differential equation (2.1) has a solution $u(\mathbf{x})$ in a neighborhood of Γ. Assume also that $u(\mathbf{x})$ is a piecewise C^2 function in both sides of the neighborhood of Γ excluding the interface Γ. Then we have the interface relations

$$u^+ = u^- + w, \tag{2.10a}$$

$$u_\xi^+ = \frac{\beta^-}{\beta^+} u_\xi^- + \frac{v}{\beta^+}, \tag{2.10b}$$

$$u_\eta^+ = u_\eta^- + w_\eta, \tag{2.10c}$$

$$u_\tau^+ = u_\tau^- + w_\tau, \tag{2.10d}$$

$$u_{\eta\tau}^+ = u_{\eta\tau}^- + \left(u_\xi^- - u_\xi^+\right)\chi_{\eta\tau} + w_{\eta\tau}, \tag{2.10e}$$

$$u_{\eta\eta}^+ = u_{\eta\eta}^- + \left(u_\xi^- - u_\xi^+\right)\chi_{\eta\eta} + w_{\eta\eta}, \tag{2.10f}$$

$$u_{\tau\tau}^+ = u_{\tau\tau}^- + \left(u_\xi^- - u_\xi^+\right)\chi_{\tau\tau} + w_{\tau\tau}, \tag{2.10g}$$

$$u_{\xi\eta}^+ = \frac{\beta^-}{\beta^+} u_{\xi\eta}^- + \left(u_\eta^+ - \frac{\beta^-}{\beta^+} u_\eta^-\right)\chi_{\eta\eta} + \left(u_\tau^+ - \frac{\beta^-}{\beta^+} u_\tau^-\right)\chi_{\eta\tau} + \frac{\beta_\eta^-}{\beta^+} u_\xi^- - \frac{\beta_\eta^+}{\beta^+} u_\xi^+ + \frac{v_\eta}{\beta^+}, \tag{2.10h}$$

$$u_{\xi\tau}^+ = \frac{\beta^-}{\beta^+} u_{\xi\tau}^- + \left(u_\eta^+ - \frac{\beta^-}{\beta^+} u_\eta^-\right)\chi_{\eta\tau} + \left(u_\tau^+ - \frac{\beta^-}{\beta^+} u_\tau^-\right)\chi_{\tau\tau} + \frac{\beta_\tau^-}{\beta^+} u_\xi^- - \frac{\beta_\tau^+}{\beta^+} u_\xi^+ + \frac{v_\tau}{\beta^+}, \tag{2.10i}$$

$$\begin{aligned} u_{\xi\xi}^+ = {} & \frac{\beta^-}{\beta^+} u_{\xi\xi}^- + \left(\frac{\beta^-}{\beta^+} - 1\right) u_{\eta\eta}^- + \left(\frac{\beta^-}{\beta^+} - 1\right) u_{\tau\tau}^- + u_\xi^+ \left(\chi_{\eta\eta} + \chi_{\tau\tau} - \frac{\beta_\xi^+}{\beta^+}\right) - u_\xi^- \left(\chi_{\eta\eta} + \chi_{\tau\tau} - \frac{\beta_\xi^-}{\beta^+}\right) + \\ & \frac{1}{\beta^+}\left(\beta_\eta^- u_\eta^- - \beta_\eta^+ u_\eta^+\right) + \frac{1}{\beta^+}\left(\beta_\tau^- u_\tau^- - \beta_\tau^+ u_\tau^+\right) + \frac{1}{\beta^+}\left([\sigma]u^- - \sigma^+[u]\right) + \frac{[f]}{\beta^+} - w_{\tau\tau} - w_{\eta\eta}. \end{aligned} \tag{2.10j}$$

Proof.

The first two interface conditions are the original jump conditions (2.2). By differentiating the first jump condition $[u] = w$ in (2.2) with respect to η and τ respectively, we get

$$[u_\xi]\chi_\eta + [u_\eta] = w_\eta, \tag{2.11}$$

$$[u_\xi]\chi_\tau + [u_\tau] = w_\tau, \tag{2.12}$$

which give (2.10c) and (2.10d) if we evaluate the equations at $(\xi, \eta, \tau) = (0,0,0)$ in the new coordinate system and use the fact $\chi_\eta(0,0) = \chi_\tau(0,0) = 0$. Differentiating (2.11) with respect to τ yields

$$\chi_\eta \frac{\partial}{\partial\tau}[u_\xi] + \chi_{\eta\tau}[u_\xi] + \chi_\tau[u_{\eta\xi}] + [u_{\eta\tau}] = w_{\eta\tau}, \tag{2.13}$$

From which we get (2.10e). Differentiating (2.11) with respect to η and differentiating (2.12) with respect to τ respectively, we obtain

$$\chi_\eta \frac{\partial}{\partial\eta}[u_\xi] + \chi_{\eta\eta}[u_\xi] + \chi_\eta[u_{\eta\xi}] + [u_{\eta\eta}] = w_{\eta\eta}, \tag{2.14}$$

$$\chi_\tau \frac{\partial}{\partial\tau}[u_\xi] + \chi_{\tau\tau}[u_\xi] + \chi_\tau[u_{\tau\xi}] + [u_{\tau\tau}] = w_{\tau\tau}, \tag{2.15}$$

from which we get (2.10f) and (2.10g). Before differentiating the jump condition of the normal derivative $[\beta u_n] = v$, in (2.2), we first express the normal vector of the interface Γ as

$$\boldsymbol{n} = \frac{(1, -\chi_\eta \ \ , -\chi_\tau)}{\sqrt{1 + \chi_\eta^2 + \chi_\tau^2}}.$$

Thus, the second interface condition $[\beta u_n] = v$ in (2.2) can be written as

$$[\beta(u_\xi - u_\eta\chi_\eta - u_\tau\chi_\tau)] = v(\eta, \tau)\sqrt{1 + \chi_\eta^2 + \chi_\tau^2}. \tag{2.17}$$

Differentiating this with respect to η, we get

$$[(\beta_\xi \chi_\eta + \beta_\eta)(u_\xi - u_\eta \chi_\eta - u_\tau \chi_\tau)] + \left[\beta(u_{\xi\xi}\chi_\eta + u_{\xi\eta} - \chi_\eta \frac{\partial}{\partial\eta} u_\eta - \chi_\tau \frac{\partial}{\partial\tau} u_\tau - u_\eta \chi_{\eta\eta} - u_\tau \chi_{\eta\tau})\right]$$
$$= v_\eta \sqrt{1 + \chi_\eta^2 + \chi_\tau^2} + v \frac{\chi_\eta \chi_{\eta\eta}}{\sqrt{1 + \chi_\eta^2 + \chi_\tau^2}}, \tag{2.18}$$

which gives (2.10h) at $(\xi, \eta, \tau) = (0,0,0)$. Similarly, differentiating (2.17) with respect to τ , we get the last interface relation (2.10j) by

$$[(\beta_\xi \chi_\tau + \beta_\tau)(u_\xi - u_\eta \chi_\eta - u_\tau \chi_\tau)] + \left[\beta(u_{\xi\xi}\chi_\tau + u_{\xi\tau} - \chi_\eta \frac{\partial}{\partial\tau} u_\eta - \chi_\tau \frac{\partial}{\partial\tau} u_\tau - u_\eta \chi_{\eta\tau} - u_\tau \chi_{\tau\tau})\right]$$
$$= v_\tau \sqrt{1 + \chi_\eta^2 + \chi_\tau^2} + v \frac{\chi_\tau \chi_{\tau\tau}}{\sqrt{1 + \chi_\eta^2 + \chi_\tau^2}},$$

which gives (2.10i) at $(\xi, \eta, \tau) = (0,0,0)$.

To get the relation for $u_{\xi\xi}^+$, we need to use the differential equation (2.8) itself, from which we can write

$$\left[\beta\left(u_{\xi\xi} + u_{\eta\eta} + u_{\tau\tau}\right) + \beta_\xi u_\xi + \beta_\eta u_\eta + \beta_\tau u_\tau - ku\right] = [f]. \tag{2.20}$$

Notice that

$$-k^- u^- + k^+ u^+ = -k^- u^- + k^+ u^- - k^+ u^- + k^+ u^+ = [k] u^- + k^+ [u]. \tag{2.21}$$

Rearranging (2.20) and using (2.21) above, we get

$$\begin{aligned}&\beta^+\left(u_{\xi\xi}^+ + u_{\eta\eta}^+ + u_{\tau\tau}^+\right) + \beta_\xi^+ u_\xi^+ + \beta_\eta^+ u_\eta^+ + \beta_\tau^+ u_\tau^+ \\ &= \beta^-\left(u_{\xi\xi}^- + u_{\eta\eta}^- + u_{\tau\tau}^-\right) + \beta_\xi^- u_\xi^- + \beta_\eta^- u_\eta^- + \beta_\tau^- u_\tau^- + [f] - k^- u^- + k^+ u^+.\end{aligned} \tag{2.22}$$

By solving $u_{\xi\xi}^+$ from the equation above, we get the last interface relation, (2.10j).

2.4 The finite difference scheme of the IIM in 3D

It is more convenient and also easier to use the zero level surface of the three-dimensional function $\varphi(x, y, z)$ to represent the interface Γ compared with other approaches. Let $\varphi(x, y, z)$ be a Lipschitz continuous function satisfying

$$\begin{aligned} &\varphi(x,y,z) < 0 \quad if \quad (x,y,z) \in \Omega^-, \\ &\varphi(x,y,z) = 0 \quad if \quad (x,y,z) \in \Gamma, \\ &\varphi(x,y,z) > 0 \quad if \quad (x,y,z) \in \Omega^+. \end{aligned} \tag{2.23}$$

Such a level set function is not unique but should be chosen as an approximation of the signed distance function to the interface Γ. We also assume that the level set function $\varphi(x, y, z)$ has up to second-order continuous partial derivatives in a neighborhood of the zero level set $\varphi = 0$. We refer the reader to [41,48] for more information about the level set method.

Let the domain Ω be a convex cubic domain in three dimension space and let $\{x_i, y_j, z_k\}$ be a uniform Cartesian grid. The level set function is defined at grid points $\varphi_{ijk} = \varphi(x_i, y_j, z_k)$. At a grid point, $\mathbf{x}_{ijk}$ we define

$$\varphi_{ijk}^{min} = \min\{\varphi_{i\pm1,j,k}, \varphi_{i,j\pm1,k}, \varphi_{i,j,k\pm1}\}, \tag{2.24}$$

$$\varphi_{ijk}^{max} = \max\{\varphi_{i\pm1,j,k}, \varphi_{i,j\pm1,k}, \varphi_{i,j,k\pm1}\}. \tag{2.25}$$

A grid point $\mathbf{x}_{ijk}$ is called an *irregular* grid point if

$$\varphi_{ijk}^{max} \varphi_{ijk}^{min} \le 0 \tag{2.26}$$

in reference to the standard central 7-point finite difference stencil. Otherwise $\mathbf{x}_{ijk}$ is called *a regular* grid point.

2.4.1 Finite difference scheme of the IIM in 3D

Similar to the IIM in two dimensions, given a Cartesian grid (x_i, y_j, z_k), $i = 0,1, \dots, L, j = 0,1, \dots, M, k = 0,1, \dots, N$, the finite difference scheme of (2.1) has the generic form:

$$\sum_{m}^{n_s} \gamma_m \, U_{i+i_m, j+j_m, k+k_m} - \sigma_{ijk} U_{ijk} = f_{ijk} + C_{ijk} \tag{2.27}$$

at any grid point (x_i, y_j, z_k) where the solution $u(x_i, y_j, z_k)$ is unknown. In the finite difference scheme above, n_s is the number of grid points involved in the finite difference stencil and $U_{ijk}$is an approximation to the solution $u(x, y, z)$ of (2.1) at (x_i, y_j, z_k). The sum over m involves a finite number of grid points neighboring (x_i, y_j, z_k). So each i_m, j_m, k_m will take values in the set $\{0, \pm1, \pm2, \dots\}$. The coefficients $\{\gamma_m\}$ and the indexes i_m, j_m, k_m depend on (i, j, k), so they should really be labeled as γ_{ijk}, etc. But for simplicity of notation, we will concentrate on a single grid point (x_i, y_j, z_k) and drop the dependency. The local truncation error at a grid point (x_i, y_j, z_k) can be defined as

$$T_{ijk} = \sum_{m}^{n_s} \gamma_m u\big(x_{i+i_m}, y_{j+j_m}, z_{k+k_m}\big) - \sigma_{ijk} u\big(x_i, y_j, z_k\big) - f_{ijk} - C_{ijk}. \tag{2.28}$$

A grid point (x_i, y_j, z_k) is called a *regular grid point* in reference to the standard 7-point finite difference stencil centered at (i, j, k) if all seven grid points are on the same side of the interface.

At regular grid point, the local truncation errors are $O(h^2)$ if the standard centered 7-point in $(n_s = 7)$ finite difference coefficients are given by

$$\gamma_{i\pm1,j,k} = \frac{\beta_{i\pm\frac{1}{2},j,k}}{h^2}, \quad \gamma_{i,j\pm1,k} = \frac{\beta_{i,j\pm\frac{1}{2},k}}{h^2}, \quad \gamma_{i,j,k\pm1} = \frac{\beta_{i,j,k\pm\frac{1}{2}}}{h^2},$$

$$\gamma_{i,j,k} = -\left(\sum\gamma_{i\pm1,j,k} + \sum\gamma_{i,j\pm1,k} + \sum\gamma_{i,j,k\pm1}\right), \tag{2.29}$$

*w*here $\beta_{i-\frac{1}{2},j,k} = \beta\left(x_i - h_x/2\,, y_j, z_k\right)$, and so forth. Here, we have assumed for simplicity that $h_x = h_y = h_z = h$. The correction term is simply $C_{ijk} = 0$ and the local truncation errors are $O(h^2)$.

If (x_i, y_j, z_k) is *an irregular grid point*, that is, the grid points in the centered 7-point stencil are from both sides of the interface, then an undetermined coefficient method is used to set up a linear system of equations for the finite difference coefficients $\{\gamma_m\}$ in (2.27) . The correction term C_{ijk} can be determined after $\{\gamma_m\}'s$ are obtained. With the assumption that the solution is piecewise smooth, a point (x_i^*, y_j^*, z_k^*) on the interface Γ near the grid point (x_i, y_j, z_k) is chosen so that Taylor expansion can be carried out from each side of the interface. Usually (x_i^*, y_j^*, z_k^*) is chosen either as the orthogonal projection of (x_i, y_j, z_k) on the interface or as the intersection of the interface on one of the axes. See Figure 2.2 below for more illustrations. Let the local coordinates of $\left(x_{i+i_m}, y_{j+j_m}, z_{k+k_m}\right)$ be (ξ_m, η_m, τ_m). The idea is to minimize the magnitude of the local truncation error T_{ijk} in (2.28) by matching the finite difference equation to the differential equation up to all second-order partial derivatives. Thus, the local truncation error would be zero if the exact solution is a piecewise quadratic function, which implies second –order convergence if the stability condition is also satisfied.

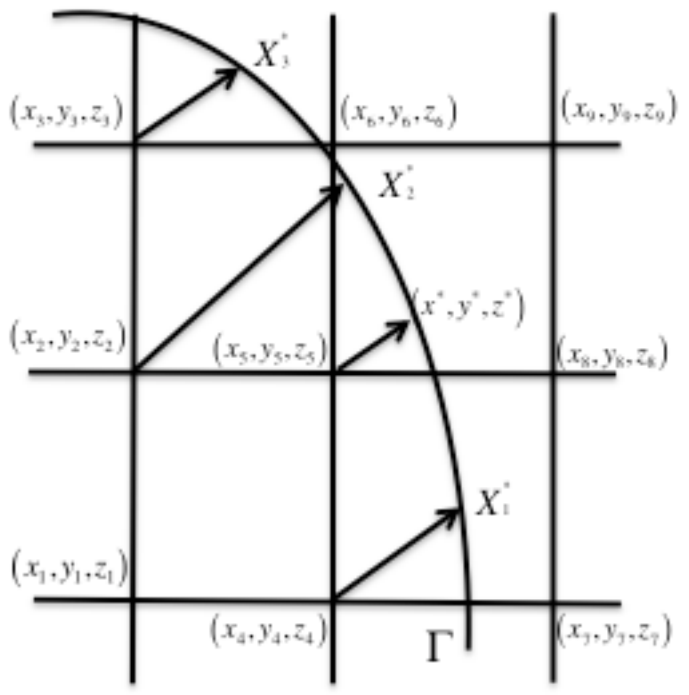

Figure 2.2 The irregular grid points and their one-sided projections on the interface Γ. The projection points are the control points on the interface Γ.

More detail with the local truncation error and correction term can be found in [33].

2.4.2 Computing the orthogonal projection in 3D Cartesian grid

To derive the finite difference equation using the IIM at an irregular grid point $\mathbf{x}_{ijk} = (x_i, y_j, z_k)$, we need to choose a point $\boldsymbol{X}_{ijk}^* = (X_i^*, Y_j^*, Z_k^*)$ on the interface that is close to $\mathbf{x}_{ijk}$. At this point, we can use the Taylor expansion from each side of the interface so that the jump conditions (2.2) and the interface relations (2.10a)-(2.10j) can be utilized. While we can choose any point (on the interface) that is close to the grid point $\mathbf{x}_{ijk}$, it is natural to choose the orthogonal projection of $\mathbf{x}_{ijk}$ on the interface. Since $\pm\nabla\varphi(\mathbf{x}_{ijk})$ is the direction where the level set function has the largest rate increase/decrease, the orthogonal projection can be determined from

$$\boldsymbol{X}_{ijk}^{*} = \mathbf{x}_{ijk} + \alpha \boldsymbol{P},$$

where $\boldsymbol{P} = \nabla\varphi/\|\nabla\varphi\|$ and $|\alpha| \sim h$ is an approximation to the distance between the grid point $\mathbf{x}_{ijk}$ and $\mathbf{X}_{ijk}^{*}$. Neglecting $O(\alpha^3)$ and higher-order terms in Taylor expansion of $\varphi\left(X_{ijk}^{*}\right) = \varphi\left(X_{ijk} + \alpha\boldsymbol{P}\right) = 0\ at\ \mathbf{x}_{ijk}$, we get a quadratic equation for α,

$$\varphi\left(\mathbf{x}_{ijk}\right) + \left|\nabla\varphi(\mathbf{x}_{ijk})\right|\alpha + \frac{1}{2}\left(\boldsymbol{P}^{T} He\left(\varphi\left(\mathbf{x}_{ijk}\right)\right)\boldsymbol{P}\right)\alpha^2 = 0,$$

where $He(\varphi\left(\mathbf{x}_{ijk}\right))$ is the Hessian matrix of φ.

$$He\left(\varphi\left(\mathbf{x}_{\boldsymbol{ijk}}\right)\right) = \begin{pmatrix} \varphi_{xx} & \varphi_{xy} & \varphi_{xz} \\ \varphi_{yx} & \varphi_{yy} & \varphi_{yz} \\ \varphi_{zx} & \varphi_{zy} & \varphi_{zz} \end{pmatrix}$$

Evaluated at $\mathbf{x}_{\boldsymbol{ijk}}$. The values of $\varphi, \varphi_x, \varphi_y, \varphi_z, \varphi_{xx}, \varphi_{xy}, \varphi_{xz}, \varphi_{yy}, \varphi_{yz}, and\ \varphi_{zz}$ are evaluated at the grid point $\mathbf{x}_{ijk} = (x_i, y_j, z_k)$ and can be approximated using the standard central finite difference formulas, for example,

$$\varphi_x \approx \frac{\varphi_{i+1,j,k} - \varphi_{i-1,j,k}}{2h_x},$$

$$\varphi_{yy} \approx \frac{\varphi_{i,j-1,k} - 2\varphi_{i,j,k} - \varphi_{i,j+1,k}}{h_y^2},$$

$$\varphi_{xy} \approx \frac{\varphi_{i-1,j-1,k} + \varphi_{i+1,j+1,k} - \varphi_{i-1,j+1,k} - \varphi_{i+1,j-1,k}}{4h_x h_y}.$$

Note that the truncation error of the quadratic approximation is $O(\alpha^3) \sim O(h^3)$. The computed projections are typically third-order accurate.

2.4.3 Setting up a local coordinate system using a level set function

Given a point $\boldsymbol{X}^* = (x^*, y^*, z^*)$ on the interface Γ, the unit normal direction of the interface at $\boldsymbol{X}^*$ is given by

$$\boldsymbol{\xi} = \frac{\nabla\varphi}{|\nabla\varphi|} = \frac{\left(\varphi_x, \varphi_y, \varphi_z\right)^T}{\sqrt{\varphi_x^2 + \varphi_y^2 + \varphi_z^2}},$$

where the partial derivatives are evaluated at (X^*, Y^*, Z^*). Note that at the interface Γ, the level set function should be chosen such that $|\nabla\varphi| \neq 0$. We prefer to choose the level set function φ as the signed distance function, which satisfies $|\nabla\varphi| = 1$ at least in two-sided neighborhood of the interface Γ. The choice of the tangential axis vectors is not unique. It is recommended that if $\varphi_x^2 + \varphi_y^2 \geq \varphi_x^2 + \varphi_z^2$, then the two tangential directions should be chosen as

$$\boldsymbol{\eta} = (\varphi_y, -\varphi_x, 0)^T / \sqrt{\varphi_x^2 + \varphi_y^2},$$

$$\boldsymbol{\tau} = \frac{\boldsymbol{s}}{|\boldsymbol{s}|},$$

where $\boldsymbol{s} = (\varphi_x\varphi_z, \varphi_y\varphi_z, -\varphi_x^2 - \varphi_y^2)^T$.

Otherwise, the two tangential directions are chosen as

$$\boldsymbol{\eta} = \frac{(\varphi_z, 0, -\varphi_x)^T}{\sqrt{\varphi_x^2 + \varphi_z^2}},$$

$$\boldsymbol{\tau} = \frac{t}{|t|},$$

where $\boldsymbol{t} = (-\varphi_x\varphi_y, \varphi_x^2 + \varphi_z^2, -\varphi_y\varphi_z)^T$.

The three unit orthogonal vectors $\boldsymbol{\xi}$, $\boldsymbol{\eta}$ and $\boldsymbol{\tau}$ form local coordinates in the normal and tangential directions.

Next in chapter 3, based on the IIM for elliptic interface problems in 3D, we will establish our 3D augmented approach. It is second order fast algorithm for elliptic interface problems with piecewise constant but discontinuous coefficients.

.

Chapter 3

The augmented approach for 3D elliptic interface problems with piecewise constant coefficients

In this chapter, we will develop a second order fast approach for elliptic interface equations with large jump in the coefficients which are piecewise constant, in both cases, continuous and discontinuous coefficients.

3.1 Introduction

The problem we are interested in solving is of the form:

Problem Formulation (I)

$$\nabla.(\beta\nabla u) = f, \qquad \text{in} \quad \Omega, \tag{3.1}$$

$$[u] = w, \qquad \text{on} \quad \Gamma, \tag{3.2a}$$
$$[\beta u_n] = q, \qquad \text{on} \quad \Gamma, \tag{3.2b}$$

with boundary conditions on $\partial\Omega$. There are two main concerns in solving problem (I) numerically. One is how to discretize it to certain accuracy. As we pointed out in chapter 1, there are a few numerical methods presented in the past few years. Most of these methods can be second order accurate in L_1 or L_2 norm, but not in L_∞ norm.

The other concern is how to solve the resulting linear system efficiently. Usually the number of iterations depends on the mesh size. Also, if the jump in the coefficient β is large, then the resulting linear system is ill-conditioned, and thus the number of iterations in solving such a linear system is large and may also be proportional to the jump in the coefficient.

Based on integral equations, some fast solvers are available for Poisson equation with piecewise constant coefficient and other problems [20,35,37,38]. In these methods, integral equations are set up at some points on the interface and boundaries for unknown source strength, and the solution then can be found using fast boundary integral techniques. Non-homogeneous source terms can be decomposed as two homogeneous problems. For example, Mayo and Greenbaum [36,37] have derived an integral equation for elliptic interface problems with piecewise constant coefficients. By solving the integral equation, they can solve such interface problems to second order accuracy in L_∞ norm using the techniques developed by Mayo in [35,36] for solving Poisson and biharmonic equations on irregular regions. Basically, 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. The total cost includes solving the integral equation and regular Poisson equation using a fast solver, so this gives a fast algorithm. The possibility of extension to variable coefficient β is mentioned in [36].

In this chapter, we develop a fast approach for elliptic interface equations with piecewise constant but discontinuous coefficients. The idea is to precondition the original elliptic equation before using the immersed interface method IIM. In order to take advantage of existing fast Poisson solvers on cubic domains, we introduce a new problem by rewriting the PDE and introducing an intermediate unknown jump in u_n. The new problem looks like a Poisson equation, which can be discretized by using the standard seven point stencil with some modification to the right-hand side. Then some existing fast Poisson solver can be called directly. Basically, this approach is equivalent to using a second order finite difference scheme to approximate the Poisson equation in Ω^+ and Ω^-, and a second order discretization for the Neumann-like interface condition

$$\beta^+ u_n^+ - \beta^- u_n^- = q.$$

Thus from the error analysis for elliptic equations with Neumann boundary conditions, for example, see [39], we would have second order accurate solution at all grid points including those near or on the interface. The GMRES method is employed to solve the Schur complement system derived from the discretization. This approach appears to be very promising not only because it is second order accurate, but also because the number of iterations in solving the Schur complement system is nearly independent of both the mesh size and the β jump coefficients.

Now, we begin to describe the approach in detail.

3.2 The augmented IIM approach description

Now, we will describe our approach and strategies in detail as pointed out earlier in Chapter 1.

3.2.1 Preconditioning the PDE to an equivalent problem

By dividing the coefficient β in each sub-domain of Ω, then we get that equation (3.1) can be rewritten into the following differential equations

$$\begin{aligned} \Delta u + \frac{\nabla \beta^+}{\beta^+} . \nabla u &= \frac{f}{\beta^+}, \qquad in\ \Omega^+, \\ \Delta u + \frac{\nabla \beta^-}{\beta^-} . \nabla u &= \frac{f}{\beta^-}, \qquad in\ \Omega^-. \end{aligned} \tag{3.3}$$

And since β is piecewise constant, we get

$$\Delta u = \frac{f}{\beta^+}, \quad in \ \ \Omega^+,$$

(3.4)

$$\Delta u = \frac{f}{\beta^-}, \quad in \ \ \Omega^-.$$

The Poisson equation can be solved readily with a fast 3D Poisson solver if we know the jump in the solution $[u] = w$ and the jump in the normal derivative $[u_n]$, but unfortunately it is given to us the flux jump condition (3.2b) which is $[\beta u_n] = q$ instead of $[u_n]$.

Notice that we can not divide β from the flux jump condition (3.2b) because β is discontinuous. In [29], a fast method for two dimensional problems is proposed, but in this thesis, we describe our augmented method for three dimensional problems. As described in [29], the idea is to augment the unknown $[u_n]= g$ and equation (3.2b) to (3.4).

Another observation here is that: let u^* be the solution of problem (I). Define

$$g^* = [u_n^*],$$

Then u^* will also be the solution of (3.3) with the jump condition $[u]= w$ and $[u_n]= g^*$. Therefore, solving problem (I) is equivalent to finding the corresponding g^* and then the solution, u_{g^*}, of the following problem with $g = g^*$.

Problem Formulation (II).

$$\Delta u + \frac{\nabla\beta^+}{\beta^+}.\nabla u = \frac{f}{\beta^+}, \quad in \ \Omega^-, \tag{3.5}$$

$$\Delta u + \frac{\nabla\beta^-}{\beta^-}.\nabla u = \frac{f}{\beta^-}, \quad in \ \Omega^-, \tag{3.6}$$

$$[u] = w, \quad on \ \ \Gamma, \tag{3.7}$$

$$[u_n] = g, \quad on \ \ \Gamma. \tag{3.8}$$

with boundary conditions on $\partial\Omega$. The key is how to find g^* efficiently. Basically, we choose an initial guess and then iteratively update it until the flux jump condition in (3.2b) is satisfied.

Notice that g^* is only defined along the interface Γ, so it two-dimensional in a three-dimensional space. Problem (II) is much easier to solve because one jump condition is given in $[u_n]$ instead of in $[\beta u_n]$.

In this thesis, we are especially interested in the case that β is piecewise constant, so the corresponding problem becomes a Poisson equation with discontinuous source term and given jump conditions. We can then use the standard seven point stencil to discretize the left-hand side of (3.6), but just modify the right-hand side to get a second order finite difference scheme, see [26,27] for the detail. Thus we can take advantage of fast Poisson solvers for the discrete system.

Here we want to compute u_{g^*} to second order accuracy. We also hope that the total cost in computing g^* and u_{g^*} is less than in computing u_{g^*} through the original problem. The key to success is to compute g^* efficiently. Now we begin to describe our approach to determine g^*. Once g^* is found, we just need one more fast Poisson solver call to get the solution u^*. As we briefed earlier in Chapter 1, only $O(N^3 log N^3)$ arithmetic operations for a mesh of N grids points are required.

3.2.2 Representation of the interface and jump conditions

We use the level set approach to represent the interface. For any quantity $\boldsymbol{p}$ defined along the interface Γ, we represent it at some specified discrete points on the interface. We call these points *Control Points.* Then any information of $\boldsymbol{p}$ is evaluated by its values at these control points, see § 3.2.6.

Obviously, one simple approach is to choose projections of irregular grid points as control points. But it is quite possible that different irregular grid points may share the common projections or some projections may be very close to each other. To avoid this situation, only projections corresponding to the irregular grid points in one side of the interface are chosen as control points. Usually, the number of control points is about half of that of total projections.

Let the control points be $\boldsymbol{X_k} = (X_k, Y_k, Z_k), k = 1,2, \dots, n_c$, where n_c is the number of the control points. Then any quantity defined on the interface can be discretized. For example, we denote the discrete vector forms of w, q and g by

$$W = (w_1, w_2, \dots, w_{n_c})^T,$$
$$Q = (q_1, q_2, \dots, q_{n_c})^T,$$
$$G = (g_1, g_2, \dots, g_{n_c})^T.$$

where

$$w_k \approx w(\boldsymbol{X_k}) = w(X_k, Y_k, Z_k),$$
$$q_k \approx q(\boldsymbol{X_k}) = q(X_k, Y_k, Z_k),$$
$$g_k \approx g(\boldsymbol{X_k}) = g(X_k, Y_k, Z_k).$$

3.2.3 Discretization

As we mentioned earlier in Chapter 2, the uniform Cartesian grid on the cube $[a_1, b_1]\times[a_2, b_2]\times[a_3, b_3]$ where Problem (I) is defined and given by:

$$x_i = a_1 + ih, \qquad y_j = a_2 + jh, \qquad z_k = a_3 + kh, \qquad 0 \le i \le l, \qquad 0 \le j \le m, \qquad 0 \le k \le n.$$

Here, for convenience, we assume that the mesh size h is given as $h = (b_1 - a_1)/l = (b_2 - a_{2)}/m = (b_3 - a_3)/n$. From the IIM, it is known that the discrete form of (3.6) can be written as

$$L_h u_{ijk} = \frac{f_{ijk}}{\beta_{ijk}} + C_{ijk}, \qquad 0 \le i \le l,\ \ 0 \le j \le m,\ \ 0 \le k \le n, \tag{3.9}$$

where

$$L_{ijk} u_{ijk} \stackrel{\text{def}}{=} \frac{u_{i-1,j,k} + u_{i+1,j,k} + u_{i,j-1,k} + u_{i,j+1,k} + u_{i,j,k-1} + u_{i,j,k+1} - 6u_{ijk}}{h^2}, \tag{3.10}$$

is the discrete Laplace operator using the standard seven point stencil. Note that if a grid point (x_i, y_j, z_k) happens to be on the interface, then f_{ijk} and β_{ijk} are defined as the limiting values from a pre-chosen side of the interface. For regular grid point, the correction term C_{ijk} is zero. For irregular grid points, C_{ijk} is computed with the IIM. Then, a fast Poisson solver, for example, a fast Fourier transformation (FFT), or a multigrid solver can be applied to solve (3.9).

3.2.4 Updating the jump in the normal derivative

As presented earlier, the key idea of the fast approach is to guess a jump in u_n initially and then update it gradually until we have that the original jump condition in βu_n is satisfied. Hence the crucial part of the fast approach is how to update $[u_n]$ efficiently.

By choosing an initial guess on $[u_n]$, say g_1, and solving Problem (II) for u_1, we actually have u_{ijk}, the approximate value of u at grid points (x_i, y_j, z_k), $0 \le i \le l, 0 \le j \le m, 0 \le k \le n$. Then based upon these values (defined on grid points, sometimes we call it a grid function), for each control point $\boldsymbol{X}_k$, $k = 1,2, \dots, n_c$, we may find the limiting values of u_n

from both sides of the interface respectively, say $u_n^-(\boldsymbol{X}_k)$ and $u_n^+(\boldsymbol{X_k}),\ k = 1,2,\ldots\ ,n_c$. This can be done by an interpolation scheme, and we will cover this later. Let

$$U_n^- = (u_n^-(X_1), u_n^-(X_2), \ldots, u_n^-(X_{n_c}))^T,$$

$$U_n^+ = (u_n^-(X_1), u_n^-(X_2), \ldots, u_n^+(X_{n_c}))^T.$$

The ideal situation would be that we already got the solution of Problem (I). In other words, U_n^- and U_n^+ already satisfy the discrete form of (3.4)

$$\beta^+ U_n^+ - \beta^- U_n^- - Q = 0. \tag{4.11}$$

Unfortunately, usually this is not true for an arbitrary initial guess.

If U_n^- and U_n^+ are exact, that is (3.14) is satisfied, then we can solve $[U_n] = U_n^+ - U_n^-$ in terms of Q, β^-, β^+, and one of $U_n^-\ and\ U_n^+$. It is straightforward to get

$$[U_n] = \frac{Q - (\beta^+ - \beta^-)U_n^-}{\beta^+},$$

or

$$[U_n] = \frac{Q - (\beta^+ - \beta^-)U_n^+}{\beta^-}.$$

And these lead to the following updating scheme

$$G_{new} = \frac{Q-(\beta^{+}-\beta^{-})U_n^{-}}{\beta^{+}}, \tag{3.12}$$

or

$$G_{new} = \frac{Q-(\beta^{+}-\beta^{-})U_n^{+}}{\beta^{-}}. \tag{3.13}$$

We did some numerical experiments by using this updating approach. It turned out that in most cases the iteration converged. However, the convergence rate was too slow, and the number of iterations seemed to be proportional to $1/h$. Now we describe the augmented approach for the 3D IIM.

The solution U of Problem (II) depends on G and W continuously. When $W = 0,\ G = 0$, the discrete linear system for Problem (II) is

$$AU = F,$$

which is the standard discretization of a usual Poisson problem. For non-homogeneous W or G, the discrete linear system of problem (II), in matrix-vector form is

$$AU + \Psi(W,G) = F, \tag{3.14}$$

where $\Psi(W,G)$ is a mapping from W and G to $C_{ijk}'s$ in (3.9). We also know that $\Psi(W,G)$ depend on the first and second derivatives of w, and the first derivatives of g, where the differentiation is carried out along the interface. At this time we do not know whether such a mapping is linear or not. However in the discrete case, as we will see later, all the derivatives are obtained by interpolation values of w or g on those control points. Therefore, $\Psi(W,G)$ is indeed a linear mapping and can be written as

$$\Psi(W, G) = BG - B_1 W, \tag{3.15}$$

where B and B_1 are two matrices with real entries. So (3.17) becomes

$$AU + BG = F + B_1 W = \bar{F}, \tag{3.16}$$

where $\bar{F}$ is defined as $F + B_1 W$.

The solution U of the equation above certainly depends on G and W we are interested in finding G^* which satisfies the discrete form of (3.2b)

$$\beta^+ U_n^+(G^*) - \beta^- U_n^-(G^*) - Q = 0. \tag{3.17}$$

Later on, we will discuss how to use the known jump G, and sometimes also Q, to interpolate U to get U_n^- and U_n^+ in detail. As we will see, U_n^- and U_n^+ depend on U, G, and Q linearly, which implies

$$\begin{aligned} \beta^+ U_n^+ - \beta^- U_n^- - Q &= EU + DG + \bar{P}Q - Q \\ &= EU + DG - PQ. \end{aligned} \tag{3.18}$$

where $E, D, \bar{P}, and\ P$ are some matrices, and $P = I - \bar{P}$. Combining (3.16) and (3.18), we obtain the system of linear equations for U and G

$$\begin{pmatrix} A & B \\ E & D \end{pmatrix} \begin{pmatrix} U \\ G \end{pmatrix} = \begin{pmatrix} \bar{F} \\ PQ \end{pmatrix}. \tag{3.19}$$

Now the question is how to solve (3.19) efficiently. We will solve for G and U in turn using the most updated information.

Solving for U is one fast Poisson solver call if β is piecewise constant. The question is how to solve for G efficiently. Eliminating U from (3.19) gives us a linear system for G

$$(D - EA^{-1}B)G = PQ - EA^{-1}\bar{F} = \bar{Q}, \tag{3.20}$$

where $\bar{Q}$ is defined as $PQ - EA^{-1}\bar{F}$. This is an $n_c \times n_c$ linear system for G, a much smaller system compared to the one for U. The coefficient matrix is the Schur complement of D in (3.19). In practice, the matrices A, B, E, D, P and the vectors $\bar{Q}, \bar{F}$ are never explicitly formed. They are merely used for theoretical purposes. Therefore an iterative method is preferred. Especially, note that the Schur complement is not symmetric, then GMRES iterative method will be employed to solve the Schur complement system.

Also note that if β is continuous, the coefficient matrix of (3.20) is invertible since $E \equiv 0$ and $D \equiv I$.

3.2.5 The Least squares approach

When we apply the GMRES method to solve the Schur complement system (3.20), we need to compute U_n^- and U_n^+ with the knowledge of U. This turns out to be a crucial step in solving the system of linear equations. Below we will describe a least square approach to interpolate U_n^- and U_n^+.

Let u be a piecewise smooth function, with discontinuities only along the interface. For a given point $\boldsymbol{X} = (X, Y, Z)$ on the interface, we want to interpolate $u(x_i, y_j, z_k)$, $0 \le i \le l$, $0 \le j \le m$, $0 \le k \le n$, to get the normal derivatives $u_n^-(\boldsymbol{X})$ and $u_n^+(\boldsymbol{X})$.

The approach is inspired by Peskin's method [43] in interpolating a velocity field to get the velocity of the interface using a discrete δ-function. The continuous and discrete forms are the following

$$u(X) = \iiint u(x,y,z)\delta(X-x)\delta(y-Y)\delta(Z-z)dxdydz, \tag{3.21}$$

$$u(X) \approx h^3 \textstyle\sum_{ijk} u_{ijk}\delta_h(X-x_i)\delta_h\left(Y-y_j\right)\delta_h(Z-z_k), \tag{3.22}$$

where $\boldsymbol{X} = (X,Y,Z)$ is a point on the interface and δ_h is a discrete Dirac δ-function. A commonly used one is

$$\delta_h(x) = \begin{cases} \dfrac{1}{4h}(1+\cos(\pi x/2h)), & if \ \ |x| < 2h, \\ 0, & if \ \ |x| \geq 2h. \end{cases} \tag{3.23}$$

Notice that $\delta_h(x)$ is a smooth function of x. Peskin's approach is very robust and only a few neighboring grid points near $\boldsymbol{X}$ are involved. However, this approach is only first order accurate and may smear out the solution near the interface.

Our interpolation formula for $u_n^-(\boldsymbol{X})$, for example, can be written in the following form

$$u_n^-(\boldsymbol{X}) \approx \sum_{(i,j,k)\in N} \gamma_{ijk}\, u_{ijk} - C, \tag{3.24}$$

where N denotes a set of neighboring grid points near $\boldsymbol{X}$, and C is a correction term which can be determined once $\gamma_{ijk}{}'s$ are known. Usually, we choose N starting with those grid points closest to $\boldsymbol{X}$. Therefore, expression (3.24) is robust and depends on the grid function u_{ijk} continuously, one very attractive property of Peskin's formula (3.25). In addition to the advantages of Peskin's approach, we also have flexibility in choosing the coefficient $\gamma_{ijk}{}'s$ and the correction term C to achieve second order accuracy. See[13].

Now we discuss how to use the IIM method to determine the coefficients $\gamma_{ijk}{}'s$ and the correction term C. They are different from point to point on the interface.

We use the same idea as used in the IIM method. Since one jump condition is given in the normal derivative of the solution, we use the local coordinates at $\boldsymbol{X} = (X, Y, Z)$

$$\begin{pmatrix} \xi \\ \eta \\ \tau \end{pmatrix} = A \begin{pmatrix} x - X \\ y - Y \\ z - Z \end{pmatrix}, \tag{3.25}$$

where A is defined in (2.4) in. Recall that under such new coordinates, the interface can be parameterized by

$$\xi = \chi(\eta\,, \tau) \quad \text{with} \quad \chi(0,0) = 0, \quad \chi_\eta(0,0) = 0, \ \chi_\tau(0,0) = 0, \tag{3.26}$$

provided the interface is smooth at $\boldsymbol{X} = (X, Y, Z)$

It is easy to check that when β is piecewise constant, the interface relation (2.10a)-(2.10j) from Chapter 2 for Problem (II) can be reduced to

$$\begin{aligned}
u^{+} &= u^{-} + w, \\
u_{\xi}^{+} &= u_{\xi}^{-} + g, \\
u_{\eta}^{+} &= u_{\eta}^{-} + w_{\eta}, \\
u_{\tau}^{+} &= u_{\tau}^{-} + w_{\tau}, \\
u_{\eta\tau}^{+} &= u_{\eta\tau}^{-} - g\chi_{\eta\tau} + w_{\eta\tau}, \\
u_{\eta\eta}^{+} &= u_{\eta\eta}^{-} - g\chi_{\eta\eta} + w_{\eta\eta}, \\
u_{\tau\tau}^{+} &= u_{\tau\tau}^{-} - g\chi_{\tau\tau} + w_{\tau\tau}, \\
u_{\xi\eta}^{+} &= u_{\xi\eta}^{-} + w_{\eta}\chi_{\eta\eta} + w_{\tau}\chi_{\eta\tau} + g_{\eta}, \\
u_{\xi\tau}^{+} &= u_{\xi\tau}^{-} + w_{\eta}\chi_{\eta\tau} + w_{\tau}\chi_{\tau\tau} + g_{\tau} \\
u_{\xi\xi}^{+} &= u_{\xi\xi}^{-} + g\left(\chi_{\eta\eta} + \chi_{\tau\tau}\right) + \left[\frac{f}{\beta}\right] - w_{\eta\eta} - w_{\tau\tau}.
\end{aligned} \tag{3.27}$$

Let (ξ_i, η_j, τ_k) be the $\xi - \eta - \tau$ coordinates of $(x_i,\, y_j,\, z_k)$, then by Taylor series expansion as we did in chapter 2,then we get

$$u(\xi_i, \eta_j, \tau_k) \approx u^{\pm} + u_{\xi}^{\pm}\xi_i + u_{\eta}^{\pm}\eta_j + u_{\tau}^{\pm}\tau_k + \tfrac{1}{2}u_{\xi\xi}^{\pm}\xi_i^2 + \tfrac{1}{2}u_{\eta\eta}^{\pm}\eta_j^2 + \tfrac{1}{2}u_{\tau\tau}^{\pm}\tau_k^2 + u_{\xi\eta}^{\pm}\xi_i\eta_j + u_{\xi\tau}^{\pm}\xi_i\tau_j + u_{\eta\tau}^{\pm}\eta_j\tau_k, \tag{3.28}$$

where + and $-$ sign depends on whether (ξ_i, η_j, τ_k) lies in the + or - side of the interface Γ. Expressing + values by - values and collecting like terms, we get

$$u_n^{-}(\boldsymbol{X}) \approx a_1u^{-} + a_2u^{+} + a_3u_{\xi}^{-} + a_4u_{\xi}^{+} + a_5u_{\eta}^{-} + a_6u_{\eta}^{+} + a_7u_{\tau}^{-} + a_8u_{\tau}^{+} + a_9u_{\xi\xi}^{-} + a_{10}u_{\xi\xi}^{+} + a_{11}u_{\eta\eta}^{-} + a_{12}u_{\eta\eta}^{+} + a_{13}u_{\tau\tau}^{-} + a_{14}u_{\tau\tau}^{+} + a_{15}u_{\xi\eta}^{-} + a_{16}u_{\xi\eta}^{+} + a_{17}u_{\xi\tau}^{-} + a_{18}u_{\xi\tau}^{+} + a_{19}u_{\eta\tau}^{-} + a_{20}u_{\eta\tau}^{+} - C + O(h^3 max|\gamma_{ijk}|), \tag{3.29}$$

where the coefficient $a_k's$ are given by (2.36). After using the interface relations in (3.27), we get

$$u_n^-(\boldsymbol{X}) \approx (a_1 + a_2)u^- + (a_3 + a_4)u_\xi^- + (a_5 + a_6)u_\eta^- + (a_7 + a_8)u_\tau^- + (a_9 + a_{10})u_{\xi\xi}^- + (a_{11} + a_{12})u_{\eta\eta}^- + (a_{13} + a_{14})u_{\tau\tau}^- + (a_{15} + a_{16})u_{\xi\eta}^- + (a_{17} + a_{18})u_{\xi\tau}^- + (a_{19} + a_{20})u_{\eta\tau}^- + a_2[u] + a_4[u_\xi] + a_6[u_\eta] + a_8[u_\tau] + a_{10}[u_{\xi\xi}] + a_{12}[u_{\eta\eta}] + a_{14}[u_{\tau\tau}] + a_{16}[u_{\xi\eta}] + a_{18}[u_{\xi\tau}] + a_{20}[u_{\eta\tau}] - C + O(h^3 \max|\gamma_{ijk}|). \tag{3.30}$$

On the other hand, we know $u_n^- = u_\xi^-$. Therefore, we have the system of linear equation for $\gamma_{ijk}{'s}$

$$\begin{cases} a_1 + a_2 = 0 \\ a_3 + a_4 = 1 \\ a_5 + a_6 = 0 \\ a_7 + a_8 = 0 \\ a_9 + a_{10} = 0 \\ a_{11} + a_{12} = 0 \\ a_{13} + a_{14} = 0 \\ a_{15} + a_{16} = 0 \\ a_{17} + a_{18} = 0 \\ a_{19} + a_{20} = 0 \end{cases} \tag{3.31}$$

If the system of linear equations (3.31) has a solution, then we can obtain a second order approximate to the normal derivative $u_n^-(\boldsymbol{X})$ by choosing an appropriate correction term C. The above linear system has ten equations. So the set of neighboring grid points N should be large enough such that at least 10 grid points are included. Usually we take more than 10 grid points and the above linear system becomes an underdetermined system which has an infinite number of solutions.

When we get the coefficient γ_{ijk}'s we can compute the a_k's From the a_k's and (3.30), we can determine the correction term C easily by

$$C = a_2[u] + a_4[u_\xi] + a_6[u_\eta] + a_8[u_\tau] + a_{10}[u_{\xi\xi}] + a_{12}[u_{\eta\eta}] + a_{14}[u_{\tau\tau}] + a_{16}[u_{\xi\eta}] + a_{18}[u_{\xi\tau}] + a_{20}[u_{\eta\tau}]$$

$$= a_2 w + a_4 g + a_6 w_\tau + a_8 w_\tau + a_{10}\big(g(\chi_{\eta\eta} + \chi_{\tau\tau}) + -w_{\eta\eta} - w_{\tau\tau}\big) + a_{12}(w_{\eta\eta} - g\chi_{\eta\eta}) + a_{14}(w_{\tau\tau} - g\chi_{\tau\tau}) + a_{16}(w_\eta \chi_{\eta\eta} + w_\tau \chi_{\eta\tau+} g_\eta) + a_{18}(w_\eta \chi_{\eta\tau} + w_\tau \chi_{\tau\tau} + g_\tau) + a_{20}(w_{\eta\tau} - g\chi_{\eta\tau}). \tag{3.32}$$

Therefore we are able to compute $u_n^-(\boldsymbol{X})$ to second order accuracy. Similarly we can derive a formula for $u_n^+(\boldsymbol{X})$ in exactly the same way, i.e., we may use the following interpolation formula

$$u_n^+(\boldsymbol{X}) \approx \sum \tilde{\gamma}_{ijk} u_{ijk} - \tilde{C}. \tag{3.33}$$

However, with the jump condition $u_n^+(\boldsymbol{X}) = u_n^-(\boldsymbol{X}) + g(\boldsymbol{X})$, we can write down a second order interpolation scheme for $u_n^+(\boldsymbol{X})$ immediately

$$u_n^+(\boldsymbol{X}) \approx \sum_{(i,j,k)\in N} \gamma_{ijk} u_{ijk} - C + g(\boldsymbol{X}), \tag{3.34}$$

where γ_{ijk}'s is the solution we computed for $u_n^-(\boldsymbol{X})$.

The above least squares technique has several nice properties. First of all, it has second order accuracy with local support. Second, it is robust. The interpolation formulas (3.24) and (3.34) depend continuously on the location of the point $\boldsymbol{X}$ and the grid points involved, and so does the truncation error for these two interpolation schemes. In other words, we have a smooth error distribution. This is very important for moving interface problems where we do not want to introduce any non-physical oscillations.

With the augmented techniques described above, we are able to solve Problem (I) to second order accuracy. In each iteration, we need to solve a Poisson equation with a modified right-hand side. A fast Poisson solver using the FFT method, the cyclic reduction, etc, can then be used. Also we need to solve a Schur complement system. The GMRES method can be used and the number if iterations depends on the condition number of the Schur complement system, if we make use of both (3.24) and (3.34) to compute $u_n^-(\boldsymbol{X})$ and $u_n^+(\boldsymbol{X})$ the condition number seems to be proportional to $1/h$. Therefore, the number of iterations will grow linearly as we increase the number of grid points. This is what we do not want to see in the fast augmented IIM approach.

A simple modification in the way of computing U_n^- and U_n^+ seems to improve the condition number of the Schur complement system. The idea is simple. We have the jump condition $[\beta u_n] = q$, which implies that if U_n^- and U_n^+ are exact, then

$$\beta^+ U_n^+ - \beta^- U_n^- = Q.$$

We can solve for U_n^- or U_n^+ in terms of Q, β^-, β^+ and $[U_n]$ to have

$$U_n^- = \frac{Q - \beta^+ [U_n]}{\beta^+ - \beta^-} \tag{3.35}$$

or

$$U_n^+ = \frac{Q - \beta^- [U_n]}{\beta^+ - \beta^-} \tag{3.36}$$

If we independently compute U_n^- and U_n^+ from (3.24) and (3.34) respectively, due to errors, usually they may not satisfy the flux jump condition. Therefore, in practice we use one of the formulas (3.24) and (3.34) to approximate U_n^- or U_n^+, and then use (3.36) or (3.35) to approximate U_n^+ or U_n^- to force the solution to satisfy the flux jump condition. This is an acceleration process or a preconditioner for the Schur complement system.

Whether we use the pair (3.24) and (3.36) or the other, (3.34) and (3.35), has only a little effect on accuracy of the computed solution and the number of iterations. In our numerical experiment, we have been using the following criteria to choose the desired pair

$$If\ \beta^+ < \beta^-: \begin{cases} Interpolation\ for\ \ U_n^+\ \ by\ \ (3.34), \\ U_n^- \ \ = \dfrac{Q - \beta^+ G}{\beta^+ - \beta^-}. \end{cases}$$

$$If\ \beta^+ > \beta^-: \begin{cases} Interpolation\ for\ \ U_n^-\ \ by\ \ (3.24), \\ U_n^+ \ \ = \frac{Q - \beta^- G}{\beta^+ - \beta^-}. \end{cases}$$

3.2.6 A least squares approach for computing an interface quantity from its values at control points

To avoid unnecessary large and ill-conditioned Schur complement system, not all projection points are chosen as control points. Therefore, when we solve the Schur complement system, we only have values of the update jump at those control points. However, to apply the immersed interface method to Problem (II), we need values of the jump and/or derivatives of the jump at all projection points. Basically, this leads us to ask, given discrete values $p_k,\ k = 1,2,\dots,n_c$ of a quantity p at control points $\boldsymbol{X}_k, k = 1,2,\dots,n_c$, how to find its value or derivatives at a projection point $\boldsymbol{X} = (X,Y,Z)$ on the interface. Again a least square approach can be used, say,

$$P(X) = \sum_{k=1}^{n_p} \alpha_k p_k, \tag{3.37}$$

$$p_{\varepsilon}(X) = \sum_{k=1}^{n_p} \lambda_k p_k, \tag{3.38}$$

$$p_{\eta}(X) = \sum_{k=1}^{n_p} \alpha_k p_k, \tag{3.39}$$

$$p_{\tau}(X) = \sum_{k=1}^{n_p} \mu_k p_k, \tag{3.40}$$

where the summation is over a set of n_p neighboring control points near the point $\boldsymbol{X}$. Also the coefficient $\alpha_K{}'$s, λ_k's ,$\sigma_k{}'s$ and $\mu_k's$ can be found in the same way as discussed in [33].

3.2.7 Invertibility of the Schur complement system

As mentioned earlier in § 3.2.4, if β is continuous, the coefficient matrix of (3.20) is invertible since $E \equiv 0$ and $D = I$. For general cases, we can show that the coefficient matrix $D - EA^{-1}B$ is also invertible if h is small enough.

We know the system of linear equations for the jump in the normal derivative G^* is implicitly defined in the discrete form of the flux jump condition

$$\beta^+ U_n^+ - \beta^- U_n^- - Q = 0. \tag{3.41}$$

With the least square interpolation (3.24) and (3.34) described earlier, the component of the equation above at a control point is approximated by

$$(\beta^+ - \beta^-)\ \sum_{(i,j,k)\epsilon N} \gamma_{ijk} u_{ijk} + (\beta^+ - (\beta^+ - \beta^-)(a_4 + a_{10}(\chi_{\eta\eta} + \chi_{\tau\tau}) - a_{12}\chi_{\eta\eta} - a_{14}\chi_{\tau\tau} - a_{20}\chi_{\eta\tau}))\ g + a_{16} g_\eta + a_{18} g_\tau - q - (\beta^+ - \beta^-)\bar{C} = 0, \tag{3.42}$$

where

$$\bar{C} = a_2 w + a_6 w_\eta + a_8 w_\tau + a_{10}(\ [f/\beta]\ - w_{\eta\eta} - w_{\tau\tau})$$

$$+a_{12}w_{\eta\eta} + a_{14}(w_{\tau\tau} - gx_{\tau\tau}) + a_{16}(w_{\eta}\chi_{\eta\eta} + w_{\tau}\chi_{\eta\tau})$$
$$+a_{18}\left(w_{\eta}\chi_{\eta\tau} + w_{\tau}\chi_{\tau\tau}\right) + a_{20}w_{\eta\tau}. \tag{3.43}$$

In vector form, it is the second equation in (3.22)

$$EU + DG = PQ. \tag{3.44}$$

If $\beta^+ = \beta^-$, then we have the unique solution for $G, G = Q/\beta^+$. Assuming now $\beta^+ \neq \beta^-$, we prove the following theorem on the invertibility of the Schur complement system (3.20).

Theorem 3.1

Assume that we use the least square interpolation formula (3.23) to compute u_n^-, and the equation (3.34) to compute u_n^+ . If h is small enough, then $\mathrm{D} - \mathrm{EA}^{-1}\mathrm{B}$ is invertible.

Proof.

It is enough to consider the homogeneous case

$$w = 0,\ \ f = 0,\ \ q = 0, \text{and } u = 0, \quad \text{on} \quad \partial\Omega.$$

In this case, $\bar{Q} = -EA^{-1}\,\bar{F} = 0$. If the theorem is not true, then there is a $G^* \neq 0$ such that $(D - EA^{-1}\,B)G^* = 0$. Let $U^* = -A^{-1}\,BG^*$, which is the discrete solution of Problem (II) with $[u_n] = g^*$, a continuous extension of G^* along the interface. Since$(D - EA^{-1}B)G^*$is second order approximation of $\beta^+u_n^+ - \beta^-u_n^-$, we conclude $[\beta u^*] = 0$ when h is small enough. So u^* is also the solution of Problem (I). However $u \equiv 0$ is an obvious solution of Problem (I). By the uniqueness of the solution of the interface problem, we must have $u^* \equiv 0$ and $g^* \equiv 0$. This contradicts the assumption that $G^* = 0$.

Next, in Chapter 4 we will provide some numerical experiments of the augmented approach based on the fast algorithm described earlier in this Chapter.

Chapter 4

Numerical experiments

4.1 Numerical examples

We have done some numerical experiments here of the 3D fast IIM approach with different jumps which show second order accuracy of the solution and more importantly also, discuss the number of iterations for the GMRES iterative method.

The computations are done by using **Dell Precision 690 Workstation** running **RHEL4, OS: RedHat Enterprise Linux, ws release 4 RHEL4, CPU: 1 XEON 5160, 2 cores (HT4 cores), memory 32GB. We used the gfortran compilier.** The initial guess for g is always chosen as 0, n_p as 50 and n_{u_n} as 50. The computational domain is [-1,1]×[-1,1]×[-1,1] unless otherwise specified. We also used $l = m = n$ in all computations. The tolerance for the GMRES iteration is taken as 10^{-5}.

We used the program **hw3crt.f** (Fishpack)[2] as the 3D fast Poisson solver, and the program **ssvdc.f** (Linpack) to perform the singular value decomposition **(SVD)** which is then used to solve the undetermined linear system. The present version of **hw3crt.f** solves the standard seven point finite difference approximation to the Helmholtz equation $\Delta u + ku = f$ in Cartesian coordinates.

Example 4.1

In this example we consider a problem with a piecewise constant coefficient β and a discontinuous source term f. The interface is a sphere $x^2 + y^2 + z^2 = 1/4$. The differential equation is

$$(\beta u_x)_x + \left(\beta u_y\right)_y + (\beta u_z)_z = f,$$

with

$$\beta(x,y,z) = \begin{cases} \beta^+, & if\ r < \dfrac{1}{2}, \\ \beta^-, & if\ r \geq \dfrac{1}{2}. \end{cases}$$

$$f(x,y,z) = \begin{cases} -6\beta^-, & if\ \ r < \dfrac{1}{2}, \\ 6\beta^+, & if\ \ r \geq \dfrac{1}{2}. \end{cases}$$

Dirichlet boundary conditions and the jump conditions (3.3) and (3.4) are determined from the exact solution and the level set function:

$$u(x,y,z) = \begin{cases} -r^2, & if\ r < \dfrac{1}{2}, \\ r^2, & if\ r \geq \dfrac{1}{2}. \end{cases}$$

i.e .,

$[u] = 2r_0^2 = 1/2,$

$[\beta u_n] = (\beta^+ + \beta^-),$

where $r = \sqrt{x^2 + y^2 + z^2}$ and on Γ, $r = r_0 = 1/2$.

Note that there are jumps in u *and* βu_n.

We tested three different cases, no jump case, samall jump case, and a big jump case. The no jump case is with $\beta^- = \beta^+ = 1$, the small jump case is with $\beta^- = 1,\ \beta^+ = 2$ and the big jump case is with $\beta^- = 1, \beta^+ = 2000.$ We see that the augmented approach does accurately give the jumps in the solution and in the normal derivative of the solution, without smearing out the solution.

Table (4.)-(4.3) show the results of a grid refinement analysis, where $l = m = n$ is the number of uniform grid points in the $x, y,$ and z directions, respectively. The maximum relative error over all grid points (the infinity norm) is defined as

$$\|E_n\|_\infty = \frac{\max_{i,j,k}|u(x_i,y_j,z_k) - u_{ijk}|}{\max_{i,j,k}|u(x_i,y_j,z_k)|}, \tag{4.1}$$

where $u_{i,j,k}$ is the computed approximation of $u(x_i, y_j, z_k)$. We also display the ratio of two successive errors and order of accuracy, respectively, as

$$\text{Ratio} = \|E_n\|/\|E_{2n}\|, \qquad \text{order} = \log\,(\|E_n\|/\|E_{2n}\|)/\log 2 \tag{4.2}$$

For a first order method, the ratio approaches to 2, and for a second order method, the ratio approaches to 4. We will use the same notation for other examples in this thesis. We see that an average ratio of 4 indicates that the augmented approach is a second order accuracy.

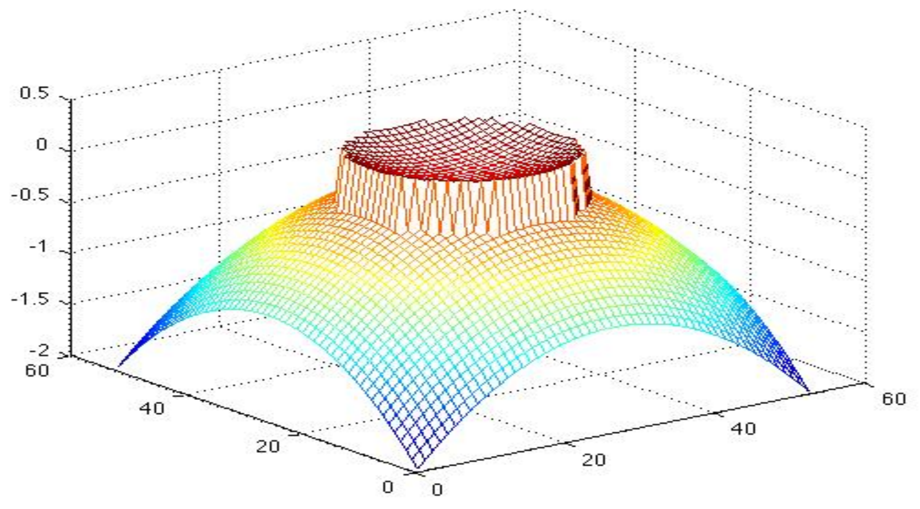

Figure 4.1 Plot of a slice of the computed solution $-u(x, y, 0)$ for example (4.1) with $\beta^{+} = 2000$, $\beta^{-} = 1$, and $l = m = n$ =52.

In Figure 4.1. The mesh size is $h = 1/26$. Both the solution and the flux $[\beta u_n]$ are discontinuous across the interface Γ. The source term f is discontinuous across the interface as well. The interface is a sphere and the computational domain is a unit cube [-1,1]×[-1,1]×[-1,1]. The plot of the solution is composed of two pieces. We see that our method does accurately give the jumps in the solution and in the normal derivative of the solution, without smearing out the solution. The discontinuity in the solution and the flux is captured sharply by our numerical method.

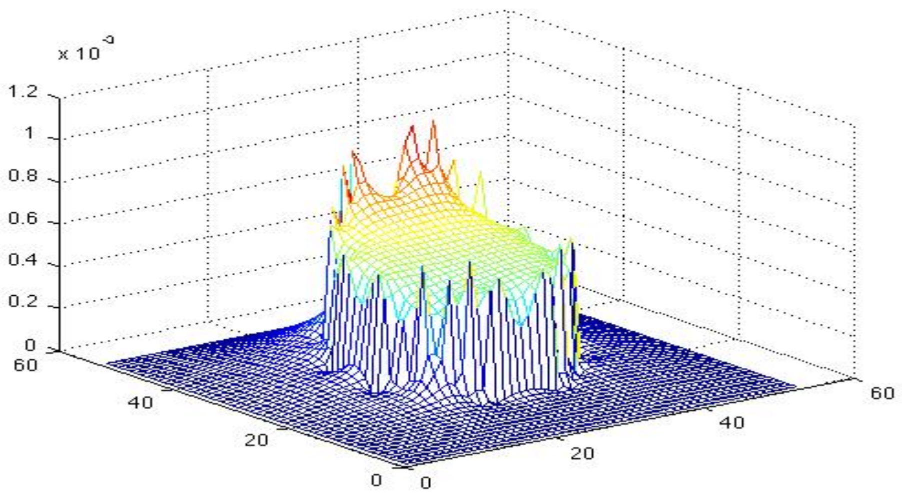

Figure 4.2 Error plot of the slice of the computed solution for example (4.1) with $\beta^+ = 2000, \beta^- = 1$, and $l = m = n = 52$.

In Figure 4.2. The mesh size is $h = 1/26$. The largest error usually occurs at those points which are close to the part of the interface which has large curvature. The errors of the solution obtained by our approach are usually more evenly distributed. The largest error in magnitude is about 0.8×10^{-3}.

Table 4.1: **The grid refinement analysis for example 4.1. Using Dell Precision Workstation 690**

n	$\beta^+ = 1$		$\beta^+ = 2$		$\beta^+ = 10$		$\beta^+ = 2000$	
	$\|\|E_n\|\|_\infty$	Ratio(order)	$\|\|E_n\|\|_\infty$	ratio(order)	$\|\|E_n\|\|_\infty$	ratio(order)	$\|\|E_n\|\|_\infty$	ratio(order)
26	0.1558E-2		0.1460E-2		0.1363E-2		0.1325E-2	
52	0.4162E-3	3.743(1.90)	0.3773E-3	3.89(1.96)	0.3478E-3	3.92(1.97)	0.3414E-3	3.88(1.96)
104	0.9919E-4	4.195(2.07)	0.8544E-4	4.42(2.14)	0.8130E-4	4.28(2.10)	0.8103E-4	4.21(2.07)

The coefficient β^- in Ω^- is 1.

Table 4.1 above shows the results of a grid refinement study with errors in the infinity norm defined over all grid points. The first column is the number of uniform grid points in the x, y and z directions. The third column is the ratio/order of convergence as defined in (4.2). We can see clearly an average of 4 which confirms second order accuracy of our method. The tolerance for the GMRES iterations is taken as 10^{-5}.

Example 4.2

In this example we consider a problem with a piecewise constant coefficient β, but variable and discontinuous source term f. The interface is a sphere $x^2 + y^2 + z^2 = 1/4$ and the differential equation is

$$(\beta u_x)_x + \left(\beta u_y\right)_y + (\beta u_z)_z = f,$$

with

$$\beta(x,y,z) = \begin{cases} \beta^- & if\ r < \frac{1}{2} \\ \beta^+ & if\ r \geq \frac{1}{2} \end{cases}$$

$$f(x,y,z) = \begin{cases} 6, & if\ \ r < \frac{1}{2}, \\ 20r^2 + \frac{\log e}{r^2}, & if\ r \geq \frac{1}{2}. \end{cases}$$

The Dirichlet boundary conditions and the jump conditions (3.3) and (3.4) are determined from the exact solution and the level set function:

$$u(x,y,z) = \begin{cases} \dfrac{r^2}{\beta^-}, & if\ \ r < \dfrac{1}{2} \\ \dfrac{r^4 + \log(2r)}{\beta^+} + \dfrac{(\frac{1}{2})^2}{\beta^-} - \dfrac{(\frac{1}{2})^4}{\beta^+}, if & r \geq \dfrac{1}{2} \end{cases}$$

i.e.,

$[u] = 0, \qquad [\beta u_n] = 4r_0^3 + \frac{log\ e}{r_0} - 2r_0,$

where $r = \sqrt{x^2 + y^2 + z^2}$ and on Γ, $r = r_0 = 1/2.$. *Note that there is no jump in* u *in this example, but in the normal derivative there is.*

The jump in the coefficient β depends on the choice of the constants β^+ and β^-. Again, We tested the different cases, no jump, small jump, and big case. Unlike in example 4.1, the solution in this example is continuous.

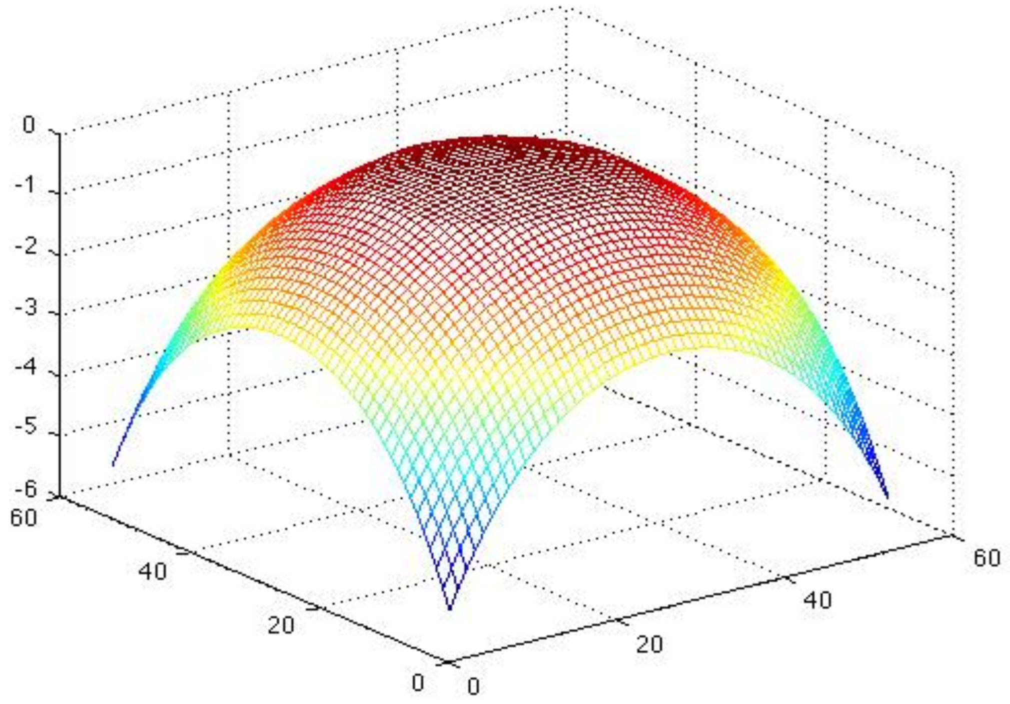

Figure 4.3 Plot of a slice of the computed solution $-u(x, y, 0)$ for example (4.2) with $\beta^+ = 1,\ \beta^- = 1$, and $l = m = n = 52$.

In Figure 4.3 The mesh size is $h = 1/26$. .The solution is continuous, but the flux $[\beta u_n]$ is not. The source term f is discontinuous across the interface. The interface is a sphere and the computational domain is a unit cube [-1,1]×[-1,1]×[-1,1]. The plot of the solution is composed as one piece. We see that our method does accurately give the jumps in the solution and in the normal derivative of the solution, without smearing out the solution. The discontinuity in the flux is captured sharply by our numerical method.

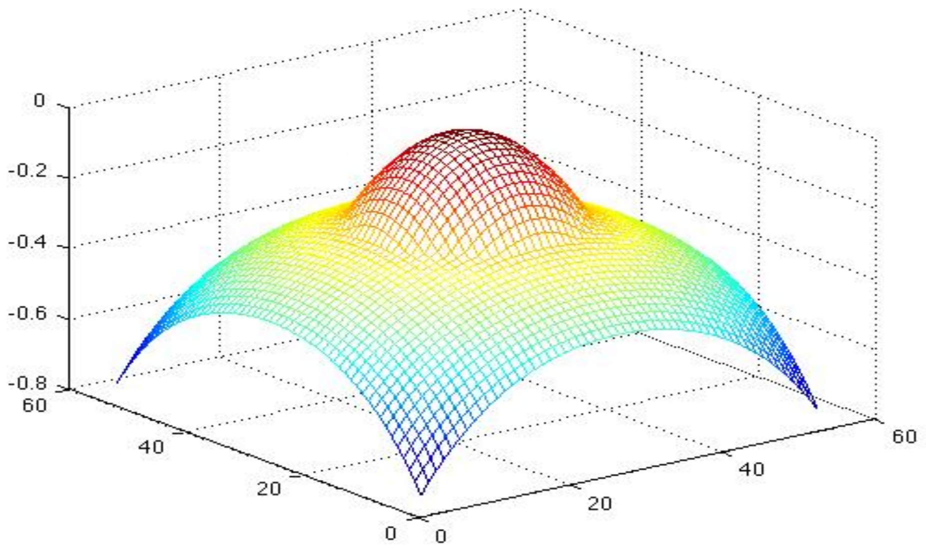

Figure 4.4 Plot of a slice of the computed solution $-u(x, y, 0)$ for example (4.2) with $\beta^+ = 10,\ \beta^- = 1$, and $l = m = n = 52$.

In Figure 4.4 The mesh size is $h = 1/26$. .The solution is continuous, but the flux $[\beta u_n]$ is not. The source term f is discontinuous across the interface. The interface is a sphere and the computational domain is a unit cube [-1,1]×[-1,1]×[-1,1]. The plot of the solution is composed as one piece. We see that our method does accurately give the jumps in the solution and in the normal derivative of the solution, without smearing out the solution. The discontinuity in the flux is captured sharply by our numerical method.

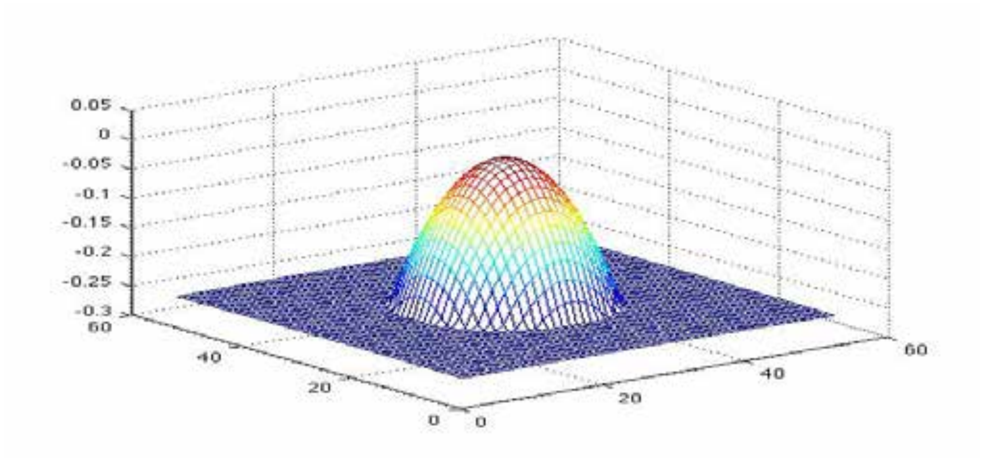

Figure 4.5 Plot of a slice of the computed solution $-u(x, y, 0)$ for example (4.2) with $\beta^+ = 2000,\ \beta^- = 1$, and $l = m = n = 52$.

In Figure 4.5 The mesh size is $h = 1/26$. .The solution is continuous, but the flux $[\beta u_n]$ is not. The source term f is discontinuous across the interface. The interface is a sphere and the computational domain is a unit cube [-1,1]×[-1,1]×[-1,1]. The plot of the solution is composed as one piece. We see that our method does accurately give the jumps in the solution and in the normal derivative of the solution, without smearing out the solution. The discontinuity in the flux is captured sharply by our numerical method.

.

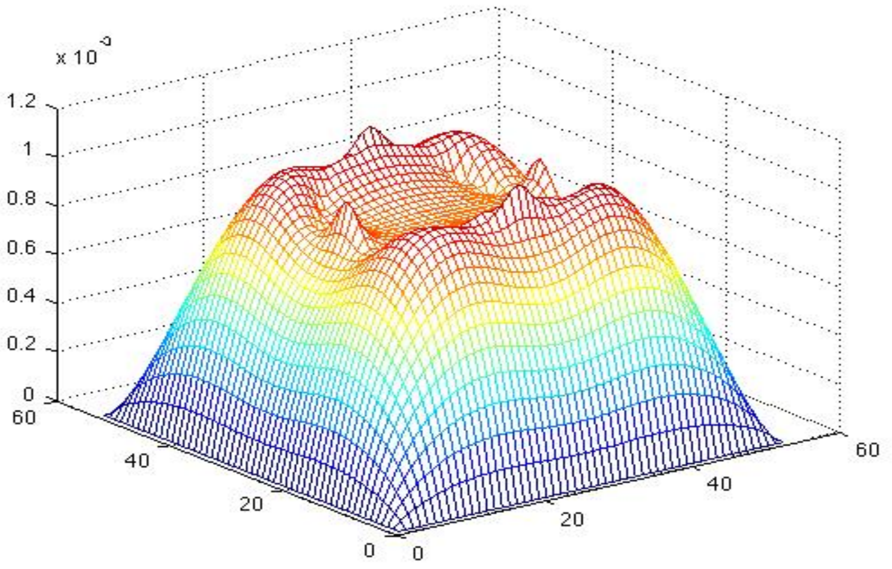

Figure 4.6 Error plot of the slice of the computed solution for example (4.2) with $\beta^{+} = 1, \beta^{-} = 1$, and $l = m = n = 52$.

Figure 4.6 is a plot of the error in the infinity norm of the slice of the computed solution. The mesh size is $h = 1/26$. The largest error usually occurs at those points which are close to the part of the interface which has large curvature. The errors of the solution obtained by our approach are usually more evenly distributed. The largest error in magnitude is about 0.8×10^{-3}.

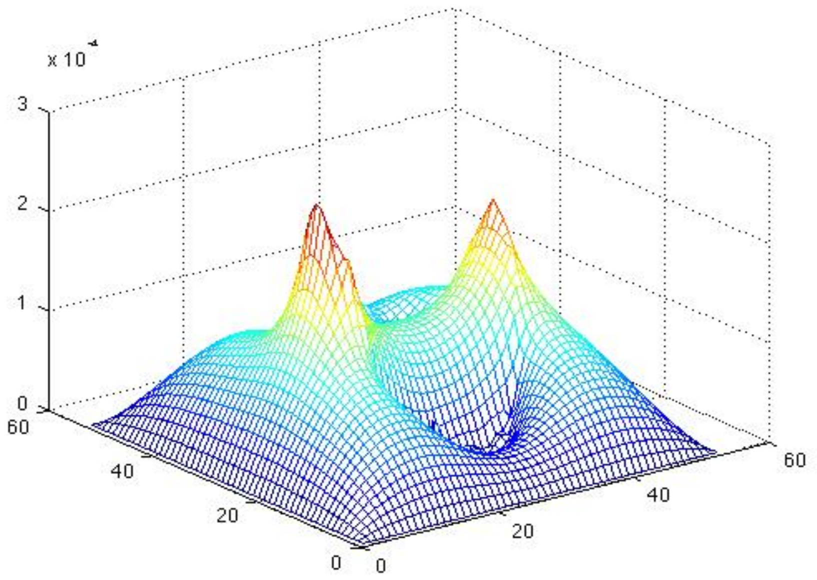

Figure 4.7 Error plot of the slice of the computed solution for example (4.1) with $\beta^+ = 10, \beta^- = 1$, and $l = m = n = 52$.

Figure 4.7 is a plot of the error in the infinity norm of the slice of the computed solution. The mesh size is $h = 1/26$. The largest error usually occurs at those points which are close to the part of the interface which has large curvature. The errors of the solution obtained by our approach are usually more evenly distributed. The largest error in magnitude is about 1.3×10^{-4}.

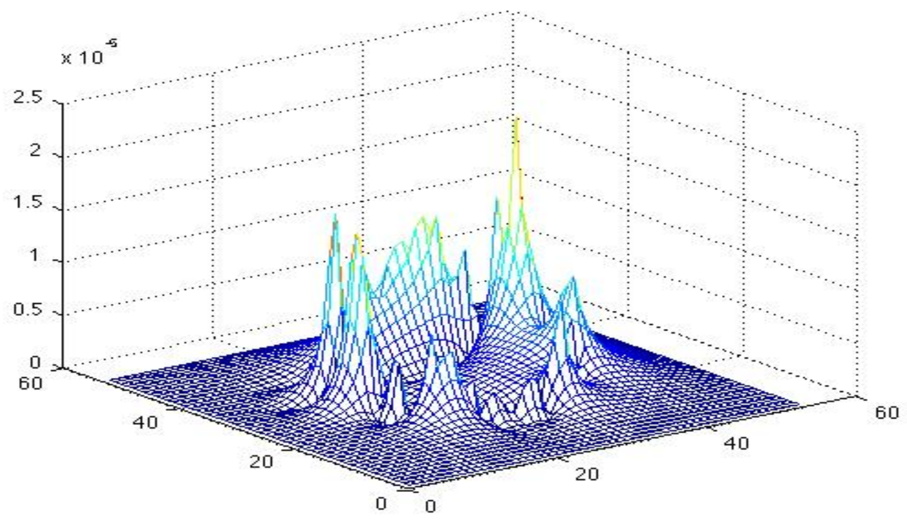

Figure 4.8 Error plot of the slice of the computed solution for example (4.1) with $\beta^+ = 2000, \beta^- = 1$, and $l = m = n = 52$.

Figure 4.8 is a plot of the error in the infinity norm of the slice of the computed solution. In this Figure where $\beta^+ = 2000, \beta^- = 1$, we see that the error in the solution drops much more rapidly. This is because the solution in Ω^+ approaches a constant as β^+ becomes large, and it is quadratic in Ω^-**.** The mesh size is $h = 1/26$. The largest error in magnitude is about 1.5×10^{-5}.

Table 4.2**: The grid refinement analysis for example (4.2). Using Dell Precision Workstation 690.**

n	$\beta^+ = 1$		$\beta^+ = 2$		$\beta^+ = 10$		$\beta^+ = 2000$	
	$\|\|E_n\|\|_\infty$	Ratio(order)	$\|\|E_n\|\|_\infty$	ratio(order)	$\|\|E_n\|\|_\infty$	ratio(order)	$\|\|E_n\|\|_\infty$	ratio(order)
26	0.5531E-3		0.5806E-3		0.1083E-2		0.4874E-2	
52	0.1475E-3	3.74(1.90)	0.1596E-3	3.64(1.86)	0.2330E-3	4.65(2.22)	0.1314E-2	3.71(1.90)
104	0.3826E-4	3.86(1.95)	0.3792E-4	4.21(2.07)	0.6030E-4	3.85(1.94)	0.3382E-3	3.89(1.96)

The coefficient β^-in $\Omega^-$is 1.

Table 4.2 above shows the results of a grid refinement study with errors in the infinity norm when $l = m = n = 52$ as shown in Figure (4.6)-(4.8). Again second order convergence is verified. The tolerance for the GMRES iterations is taken as 10^{-5}.

Example 4.3

In this example we consider a problem with a piecewise constant coefficient β and a discontinuous source term f. The interface is a sphere $x^2 + y^2 + z^2 = 1/4$. The differential equation is

$$(\beta u_x)_x + \left(\beta u_y\right)_y + (\beta u_z)_z = f,$$

with

$$\beta(x,y,z) = \begin{cases} \beta^+, & if \;\; r < \dfrac{1}{2}, \\ \beta^-, & if \;\; r \geq \dfrac{1}{2}. \end{cases}$$

$$f(x,y,z) = \begin{cases} -200\beta^- r^2, & if \quad r < \dfrac{1}{2}, \\ 20\beta^+ r^2, & if \quad r \geq \dfrac{1}{2}. \end{cases}$$

Dirichlet boundary conditions and the jump conditions (3.3) and (3.4) are determined from the exact solution and the level set function:

$$u(x,y,z) = \begin{cases} -10r^4, & if \;\; r < \dfrac{1}{2}, \\ r^4, & if \;\; r \geq \dfrac{1}{2}. \end{cases}$$

i.e .,

$[u] = 11r_0^4 = 11/16,$

$[\beta u_n] = 4(10\beta^- + \beta^+)/2,$

where $r = \sqrt{x^2 + y^2 + z^2}$ and on $\Gamma,\; r = r_0 = 1/2.$

Note that there are jumps in u *and* βu_n.

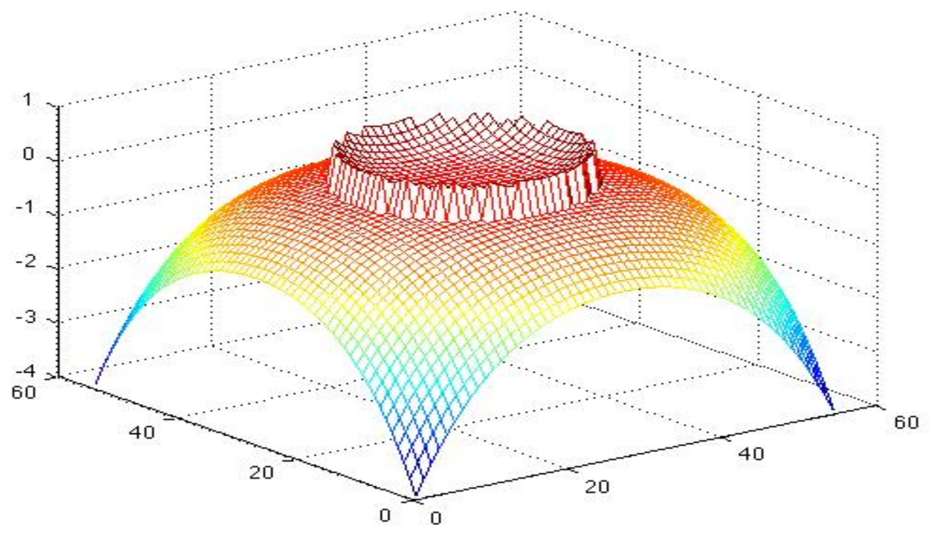

Figure 4.9 Plot of a slice of the computed solution $-u(x, y, 0)$ for example (4.3) with $\beta^+ = 2000$, $\beta^- = 1$, and $l = m = n$ =52.

In Figure 4.9 The mesh size is $h = 1/26$. Both the solution and the flux $[\beta u_n]$ are discontinuous across the interface Γ. The source term f is discontinuous across the interface as well. The interface is a sphere and the computational domain is a unit cube [-1,1]×[-1,1]×[-1,1]. The plot of the solution is composed of two pieces. We see that our method does accurately give the jumps in the solution and in the normal derivative of the solution, without smearing out the solution. The discontinuity in the solution and the flux is captured sharply by our numerical method.

.

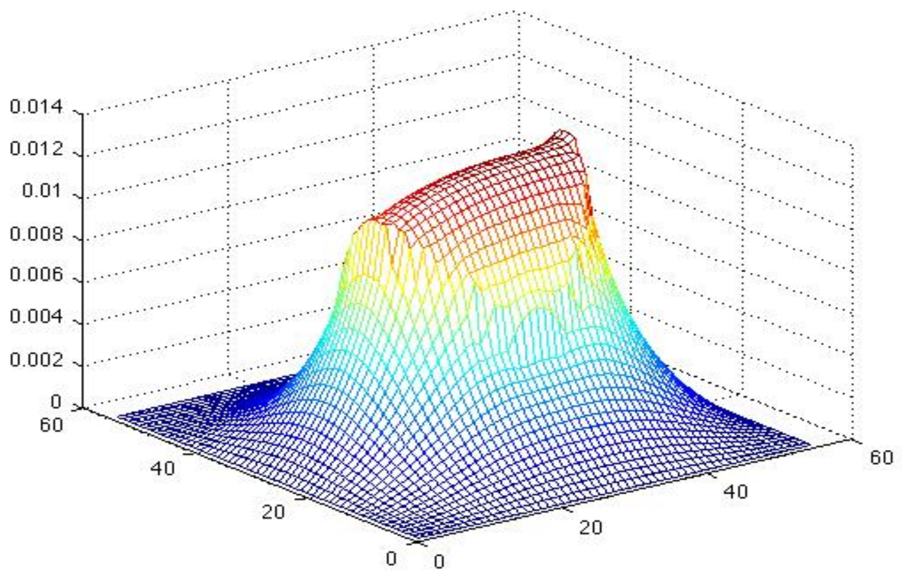

Figure 4.10 Error plot of the slice of the computed solution for example (4.3) with $\beta^+ = 2000, \beta^- = 1$, and $l = m = n = 52$.

Figure 4.10 is a plot of the error in the infinity norm of the slice of the computed solution. The mesh size is $h = 1/26$. The largest error usually occurs at those points which are close to the part of the interface which has large curvature. The errors of the solution obtained by our approach are usually more evenly distributed. The largest error in magnitude is about 0.9×10^{-2}.

Table 4.3**: The grid refinement analysis for example 4.3. Using Dell Precision Workstation 690.**

n	$\beta^+ = 1$		$\beta^+ = 2$		$\beta^+ = 10$		$\beta^+ = 2000$	
	$\|\|E_n\|\|_\infty$	Ratio(order)	$\|\|E_n\|\|_\infty$	ratio(order)	$\|\|E_n\|\|_\infty$	ratio(order)	$\|\|E_n\|\|_\infty$	ratio(order)
26	0.2180E-2		0.2849E-2		0.4672E-2		0.5234E-2	
52	0.5686E-3	3.83(1.93)	0.6499E-3	4.38(2.13)	0.1121E-2	4.17(2.06)	0.1245E-2	4.20(2.07)
104	0.1229E-3	4.63(2.21)	0.1729E-3	3.76(1.91)	0.2997E-3	3.74(1.90)	0.3456E-3	3.60(1.85)

The coefficient β^{-}in $\Omega^{-}$is 1.

Table 4.3 above shows the results of a grid refinement study with errors in the infinity norm when $l = m = n = 52$ as shown in Figure 4.10. Again. we can see again clearly the average ratio is close to 4 indicating second order accuracy of our method. The tolerance for the GMRES iterations is taken as 10^{-5}.

4.2 The augmented approach efficiency analysis

From the numerical tests we have already seen that the augmented approach is second order accurate and can deal with large enough mesh size and large enough jumps in the coefficient. We also want to know whether the number of iterations is dependent on the mesh size or the jump in the coefficient β.

Tables (4.4)-(4.6) list some statistics for the above examples, where N_{irreg} denotes the number of irregular points, N_{cntr} denotes the number of control points and N_{iter} is the number of iterations (i.e., the number of calls to the fast Poisson solver). It can be seen clearly for all cases, that our method is fast in terms of number of iterations and in terms of time (in seconds) considering the very large condition number of the system of equations from a direct discretization even for a regular domain such as a cube, also the number of iterations does not increase by a big jump greatly. If there is no jump in β, the GMRES method converges in one step.

However, the number of iterations seems to depend on the jump in the coefficient β. This is different from two-dimensional cases in [29].

Table 4.4**: CPU time for example 4.1. Using Dell Precision Workstation 690**

n	N_{irreg}	N_{cntr}	$\beta^+ = 1$		$\beta^+ = 2$		$\beta^+ = 10$		$\beta^+ = 2000$	
			CPU(s)	N_{iter}	CPU(s)	N_{iter}	CPU(s)	N_{iter}	CPU(s)	N_{iter}
26	920	506	1.8202	5	2.3821	10	2.7173	13	3.1565	17
52	3528	1820	7.6128	5	10.2216	10	12.3317	14	14.9292	19
104	14048	7172	35.979	5	49.8391	10	61.0546	14	77.6706	20

The coefficient $\boldsymbol{\beta^-}$in $\boldsymbol{\Omega^-}$is 1.

In Table 4.4 we can see that our method is fast in terms of number of iterations and in terms of time (in seconds) considering the very large condition number of the system of equations from a direct discretization even for a regular domain such as a cube. However, the number of iteration depends on the mesh size and the jump in the coefficient β, but that does not increase the CPU time greatly.

Table 4.5**: CPU time for example 4.2. Using Dell Precision Workstation 690**

n	N_{irreg}	N_{cntr}	$\beta^+ = 1$		$\beta^+ = 2$		$\beta^+ = 10$		$\beta^+ = 2000$	
			CPU(s)	N_{iter}	CPU(s)	N_{iter}	CPU(s)	N_{iter}	CPU(s)	N_{iter}
26	920	506	1.8148	5	2.2664	9	2.6036	12	3.1553	17
52	3528	1820	7.6364	5	9.7203	9	11.8205	13	14.9187	19
104	14048	7172	35.924	5	47.0968	9	58.3436	13	77.8693	20

The coefficient $\boldsymbol{\beta^-}$in $\boldsymbol{\Omega^-}$is 1.

In Table 4.5 above, again we can see that the number of iteration depends on the mesh size and the jump in the coefficient β, but that does not increase the CPU time greatly.

Table 4.6: **CPU time for example 4.3. Using Dell Precision Workstation 690**

n	N_{irreg}	N_{cntr}	$\beta^+ = 1$		$\beta^+ = 2$		$\beta^+ = 10$		$\beta^+ = 2000$	
			CPU(s)	N_{iter}	CPU(s)	N_{iter}	CPU(s)	N_{iter}	CPU(s)	N_{iter}
26	920	506	1.8125	5	2.3849	10	2.7095	13	3.2683	18
52	3528	1820	7.5889	5	10.236	10	12.8407	15	15.4590	20
104	14048	7172	35.868	5	49.854	10	64.0625	15	80.5074	21

The coefficient β^-in $\Omega^-$is 1.

In Table 4.6 again we can see that the number of iteration depends on the mesh size and the jump in the coefficient β, but that does not increase the CPU time greatly.

Chapter 5

Applications of the fast 3D solver

5.1 The augmented method for Holmholtz / Poisson equations on irregular domains

Sometimes a problem on an irregular domain can be handled more easily as an interface problem by embedding the domain into a cubic domain and then solving the equation on a Cartesian grid in the cube. The original boundary then becomes an interface. As an example, suppose we want to solve a three-dimensional elliptic equation on an irregular domain Ω. We can embed the domain in a larger cubic domain R. For example, we could solve the following Laplace equation with a Dirichlet boundary condition

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

By extending it to the problem

$$\begin{aligned} &\Delta u=\iint F(\boldsymbol{X})\delta(x-X)\delta(y-Y)\delta(z-Z)dS, && \text{in} \quad R, \\ &\mathrm{u}=0, && \text{on} \quad \partial\mathrm{R}. \end{aligned} \tag{5.1}$$

The problem is then to determine F(**X**) so that the boundary condition $u=v \; on \; \partial\Omega$ 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 [7,15,35,45] and methods based on solving integral equations along $\partial\Omega$. With the idea of the augmented approach, we can also develop an embedding technique to solve elliptic equations on complicated regions with Dirichlet, Neumann, or Robin boundary conditions as will be indicated in chapter 6 as future work. In this Chapter, we show how the augmented approach can be utilized to solve exterior / interior Poisson equations on irregular domains.

First, consider 3D interface Poisson equation with Dirichlet boundary condition

$$\begin{aligned} &\Delta u + ku = f, && \text{in} \quad \Omega, \\ &u = v, && \text{on} \quad \partial\Omega. \end{aligned} \tag{5.2}$$

where Ω is a cubic volume with an arbitrary closed void region, $\partial\Omega$ is the interior boundary of Ω and ∂R is the exterior boundary of Ω, see Figure (5.1) below for an illustration.

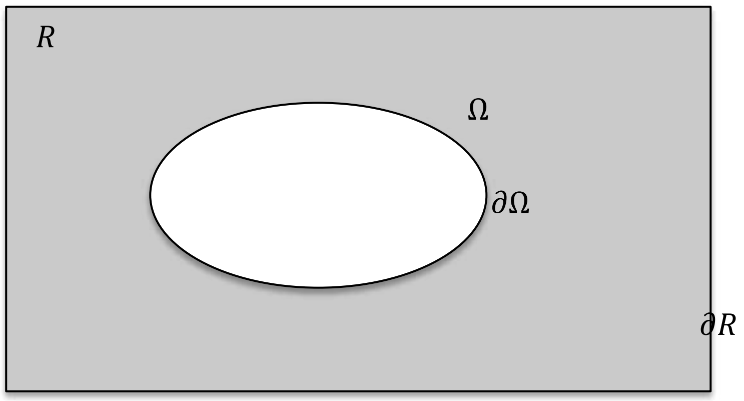

Figure 5.1 : A diagram for exterior Poisson problem.

From the embedding or the fictitious techniques, we can imagine the irregular region as an embedded region into a cube R. We may treat the original problem as an interface problem and use the IIM to solve it. However, recall that to make the resulting interface problem well-posed, usually we need to know not only the jump in the solution [u], but also the jump in the normal derivative of the solution $[u_n]$, across the boundary (now it becomes the interface). We may simply use the Dirichlet boundary condition as the jump condition in the solution, but unfortunately there seems no way to know the exact jump in the normal derivative of the solution. Thus, it will be assumed to be an augmented variable.

Based on the same idea as used in the augmented approach, we can solve the above problem by choosing an initial guess on $[u_n]$, and then updating it until the original boundary condition is satisfied. Below we begin to describe this approach in more detail.

We extend the source term in the Poisson equation by zero outside Ω. On the irregular boundary $\partial\Omega$, we allow a finite jump $[u]$ in the solution itself. One particular choice is just to use the original boundary condition v as $[u]$ and let $u = 0$ on the boundary ∂R of the cube R. As for the jump in the normal derivative of the solution $[u_n]$, we may use an initial guess, say, $[u_n] = g$ (usually 0). This extension leads to the following interface problem

$$\begin{aligned} &\Delta u + ku = \begin{cases} f, & if \;\; (x,y,z) \in \Omega, \\ 0, & if \;\; (x,y,z) \notin \Omega, \end{cases} \\ &[u] = v, \qquad \text{on} \quad \partial\Omega, \\ &[u_n] = g, \qquad \text{on} \quad \partial\Omega, \\ &u = 0, \qquad \text{on} \quad \partial\text{R}. \end{aligned} \tag{5.3}$$

We then use the GMRES iteration to update g until the original boundary condition is satisfied, i.e.,

$$u^- = v, \quad on \;\; \partial\Omega, \tag{5.4}$$

where u^- is the limiting value of the solution on the boundary from the inside of $\partial\Omega$.

Now the augmented approach can be used and only some minor changes are needed. First, instead of using $[\beta u_n] = g$ as the convergence-checking rule, we use $u^- = v$. Therefore, instead of interpolating $u_n^+ \; and \; u_n^-$ with the knowledge of u_{ijk}'s, we need to find u^+ and u^-. The same least square approach can still be used here. For example, our interpolation formula for u^-can be written in the following:

$$u^- \approx \sum_{i,j,k} \gamma_{ijk} u_{ijk} - Q. \tag{5.5}$$

The same idea also applies to interior Poisson problem with Dirichlet boundary conditions as follows

$$\begin{aligned} &\Delta u + ku = f, && \text{in} \quad \partial\Omega, \\ &\mathrm{u} = \mathrm{v}, && \text{on} \quad \partial\Omega, \\ &u = u_0, && \text{on} \quad \partial\mathrm{R}. \end{aligned} \tag{5.6}$$

where Ω is an arbitrary closed region in 3D space, see Figure (5.2) for an illustrations.

By extending the source term in the Poisson equation by zero outside Ω and forming interface conditions across $\partial\Omega$, we get the following interface problem

$$\begin{aligned} &\Delta u + ku = \begin{cases} f, & if \; (x,y,z) \in \Omega \\ 0, & if \; (x,y,z) \notin \Omega \end{cases} \\ &[u] = v, && \text{on} \quad \partial\Omega, \\ &[u_n] = g, && \text{on} \quad \partial\Omega, \\ &u = u_0, && \text{on} \quad \partial\mathrm{R}. \end{aligned} \tag{5.7}$$

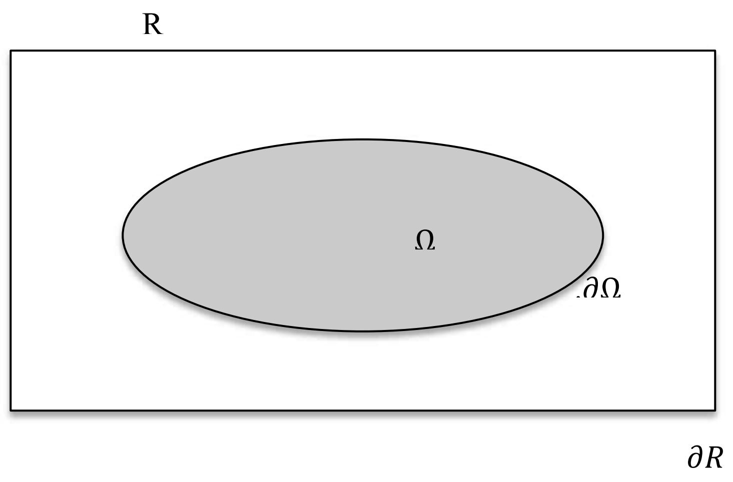

Figure 5.2: A diagram for interior Poisson problem.

Similarly, we then use the GMRES iteration to update g until the original interior boundary condition is satisfied, i.e.,

$$u^{+} = v, \qquad \text{on} \quad \partial\Omega. \tag{5.8}$$

where u^{+} is the limiting value of the solution on the interior boundary from the outside of $\partial\Omega$, which can still be obtained by the same least square approach as in § 3.2.5 , i.e.,

$$u^{+} \approx \sum_{i,j,k} \gamma_{i,j,k} u_{i,j,k} - Q. \tag{5.9}$$

We have tested the fictitious or embedding techniques by solving some exterior or interior Poisson equations with Dirichlet boundary conditions. As mentioned earlier, the computations are done by using **Dell Precision 690 Workstation** running **RHEL4, OS: RedHat Enterprise Linux, ws release 4 RHEL4, CPU: 1 XEON 5160, 2 cores (HT4 cores), memory 32GB.** and by using **gfortran** compiler. The tolerance for the GMRES is taken as 10^{-5}.

5.1 Numerical experiments

We will provide two examples in solving three-dimensional Poisson equations on exterior and interior irregular domains, respectively, to show the efficiency of the augmented approach. We want to show second order accuracy of the solution, and more importantly also, show that the number of iterations is nearly independent of the mesh size in this chapter.

Example 5.1

In this example, the domain is the exterior of the sphere $x^2 + y^2 + z^2 = 1/4$. The differential equation is

$$u_{xx} + u_{yy} + u_{zz} = 20r^2 + \frac{\log(e)}{r^2}$$

Dirichlet boundary condition is chosen from the following exact solution and level set function:

$$u(x, y, z) = r^4 + \log(2r) + \frac{3}{16},$$

where $r = \sqrt{x^2 + y^2 + z^2}$. and on Γ, $r = r_0 = 1/2$

Below, Figure 5.3 (outside of the sphere) shows a slice of the computed solution: $-u(x, y, 0)$. The sphere is embedded into a unit cube [-1,1]×[-1,1]×[-1,1].

Again we did the simulations, but on a 104×104×104 grids. The mesh size is $h = 1/52$.

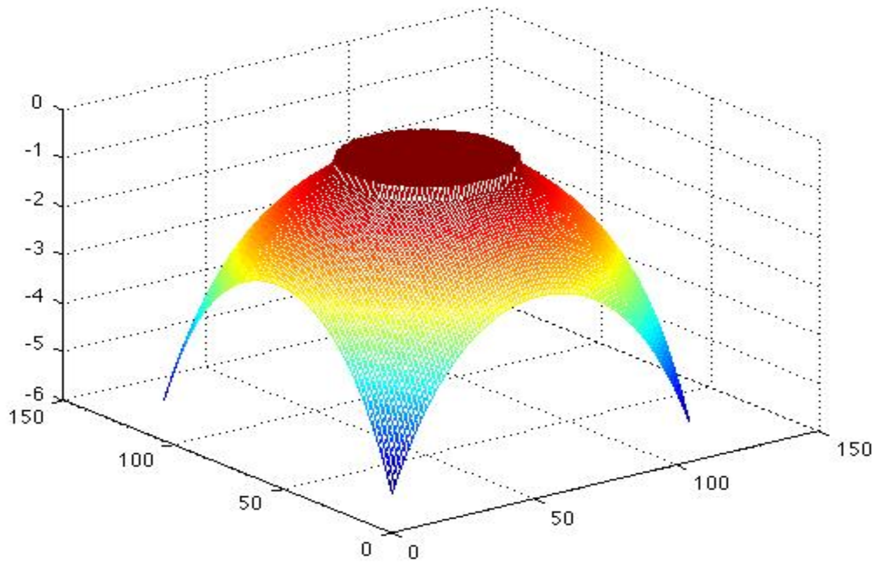

Figure 5.3 : Plot of a slice of the computed solution $-u(x, y, 0)$ for example (5.1) with $l = m = n = 104$.

In Figure 5.3(outside of the sphere) shows a slice of the computed solution : $-u(x, y, 0)$ on a 104×104×104 grids. The mesh size is $h = 1/52$. Both the solution and the flux $[\beta u_n]$ are discontinuous across the interface Γ. The source term f is discontinuous across the interface as well. The mesh size is $h = 1/52$. The interface is a sphere and the computational domain is a unit cube [-1,1]×[-1,1]×[-1,1]. The plot of the solution is composed of two pieces. We see that our method does accurately give the jumps in the solution and in the normal derivative of the solution, without smearing out the solution. The discontinuity in the solution and the flux is captured sharply by our numerical method.

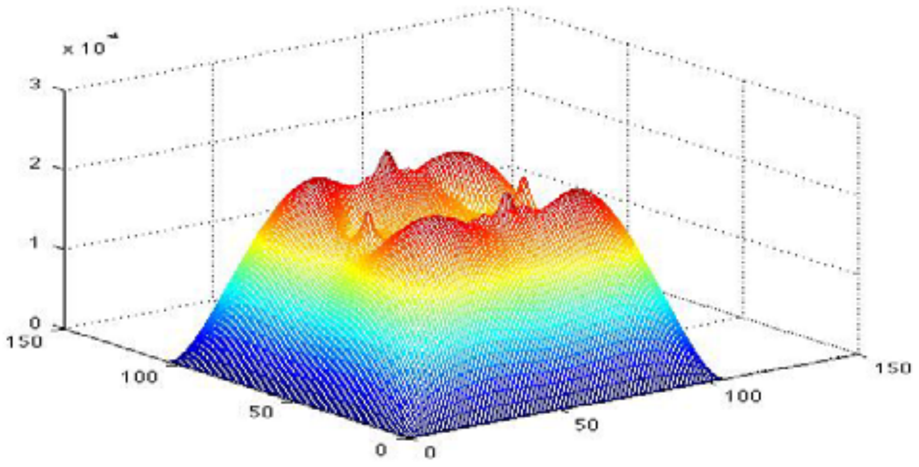

Figure 5.4 : Error plot of the slice of the computed solution for example (5.1) with $l = m = n = 104$.

Figure 5.4 is a plot of the error in the infinity norm of the slice of the computed solution plotted in Figure 5.3 on a 104×104×104 grids. The mesh size is $h = 1/52$. The largest error in magnitude is about 1.2×10^{-4}. The number of iterations for solving the Schur complement system using a GMRES is almost independent the mesh size h.

Example 5.2

The differential equation is

$$u_{xx} + u_{yy} + u_{zz} = f,$$

in the interior of the sphere $x^2 + y^2 + z^2 = 1/4$

with

$$f(x,y,z) = \begin{cases} -200r^2, & if\ r < 1/2, \\ 20r^2, & if\ r \geq 1/2. \end{cases}$$

Dirichlet boundary condition is chosen from the exact solution and level set function:

$$u(x, y, z) = -10r^4.$$

where $r = \sqrt{x^2 + y^2 + z^2}$ and on Γ, $r = r_0 = 1/2$.

Again the domain is embedded into the unit cube [-1,1]×[-1,1]×[-1,1].

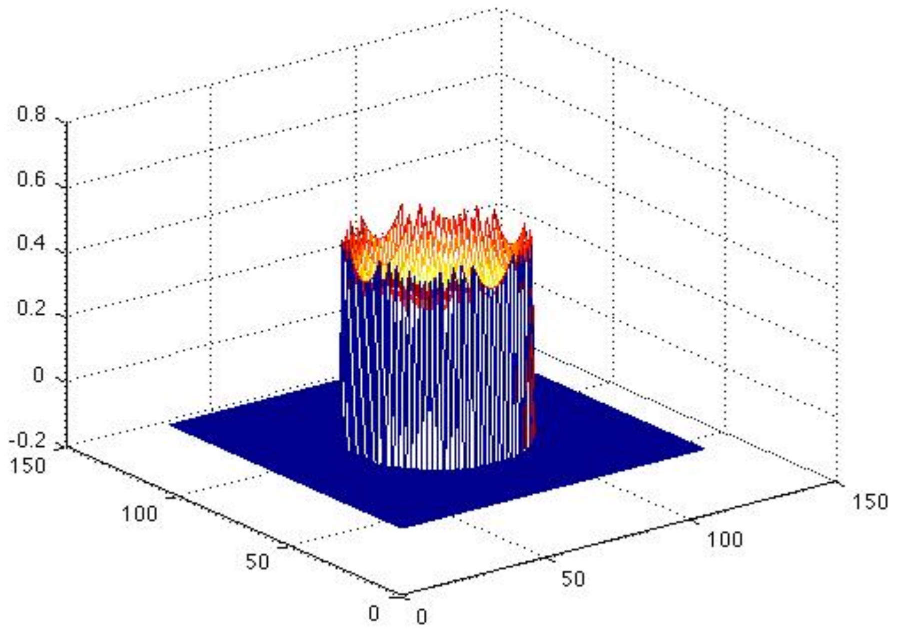

Figure 5.5: Plot of a slice of the computed solution $-u(x, y, 0)$ for example (5.2) with $l = m = n$ =104.

Figure 5.5 (inside on the top) shows a slice of the computed solution: $-u(x, y, 0)$ on a 104×104×104 grids. The mesh size is $h = 1/52$. Both the solution and the flux $[\beta u_n]$ are discontinuous across the interface Γ. The source term f is discontinuous across the interface as well. The mesh size is $h = 1/52$. The interface is a sphere and the computational domain is a unit cube [-1,1]×[-1,1]×[-1,1]. The plot of the solution is composed of two pieces. We see that our method does accurately give the jumps in the solution and in the normal derivative of the solution, without smearing out the solution. The discontinuity in the solution and the flux is captured sharply by our numerical method.

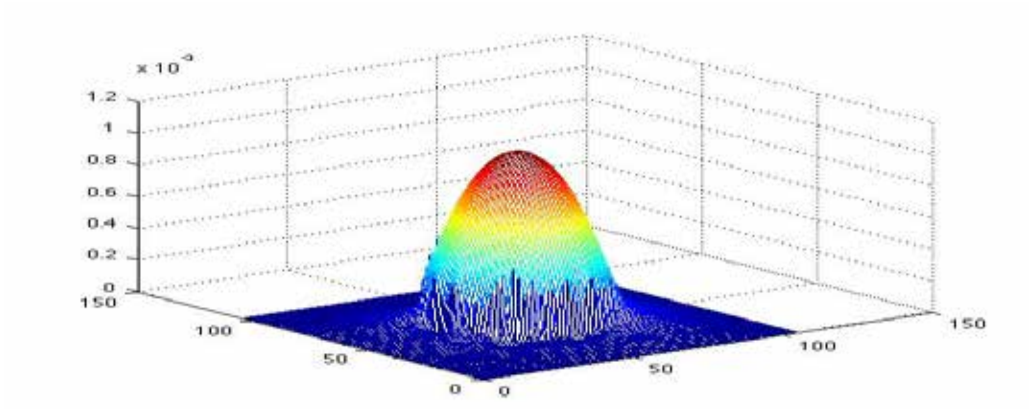

Figure 5.6: Error plot of the slice of the computed solution for example (5.2) with $l = m = n = 104$.

Figure 5.6 is a plot of the error in the infinity norm of the slice of the computed solution on a 104×104×104 grids. The mesh size is $h = 1/52$.The largest error in magnitude is about 0.5×10^{-3}. The number of iterations for solving the Schur complement system using a GMRES is almost independent the mesh size h.

Table (5.1)-(5.2) show the results of a grid refinement analysis, where $l = m = n$ is the number of uniform grid points in the $x, y,$ and z direction, respectively. The infinity norm over all grid points and order of convergence are defined as in (4.1) and (4.2) respectively.

show that the approach is second order accurate and we notice that the number of calls to the fast Poisson solver on the cubic domain is independent of the mesh size similar to the case of two space dimensions proposed in [29] although it may depend on the geometry of the domain. They also show the error in the infinity norm and other information. In those tables, N_{irreg} and N_{cntr} are the number of total irregular grid points and the number of control points respectively; N_{iter} is the number of iterations of the GMRES method or the number the calls of the 3D fast Poisson solver.

Table 5.1**: the grid refinement analysis of Example 5.1. Using Dell Precision Workstation 690**

n	N_{irreg}	N_{cntr}	CPU(s)	N_{iter}	$\|\|E_n\|\|_\infty$	Ratio(order)
26	920	506	2.1637	5	0.3898E-04	
52	3528	1820	8.6127	5	0.9849E-04	3.9577(1.98)
104	14048	7172	37.1104	5	0.2419E-04	4.0715(2.03)

Table 5.1 above shows the results of a grid refinement study with errors in the infinity norm and other information. We can see that the method still has second order accuracy when we use the embedding technique. We see that only a few iterations (only 5) are needed and the number is independent of the mesh size. The CPU time column (in seconds) shows that our method is very fast considering the very large condition number of the system of equations from a direct discretization even for a regular domain such as a cube. Furthermore, in this Table we can see that the number of iterations (only 5) is independent of the mesh size

as in the case of two space dimensions. The CPU time does not increase much. Thus, our method here is fast in terms of number of GMRES iterations and in terms of CPU time.

Table 5.2**: the grid refinement analysis for Example 5.2. Using Dell Precision Workstation 690**

n	N_{irreg}	N_{cntr}	CPU(s)	N_{iter}	$\|E_n\|_\infty$	Ratio(order)
26	920	506	2.1577	5	0.2445E-01	
52	3528	1820	8.5897	5	0.6882E-02	3.9577(1.83)
104	14048	7172	36.9984	5	0.1744E-02	4.0715(1.98)

Table 5.2 above shows the results of a grid refinement study with errors in the infinity norm and other information. Again, we can see that the method still has second order accuracy when we use the embedding technique. Furthermore, in this Table we can see that the number of iterations (only 5) is independent of the mesh size as in the case of two space dimension. The CPU time does not increase much.

So far, in the discussion for a Poisson equation on an irregular domain, we form an interface problem that requires a known fixed jump in the solution and set an unknown jump in the normal derivative of the solution. Then we iteratively update the jump in the normal derivative using the GMRES iteration to 10^{-5} tolerance until the original boundary condition is satisfied. Alternatively, we can set a known fixed jump in the normal derivative and an unknown jump in the solution. Then we use a similar GMRES iteration to update the jump in the solution until the original boundary condition is satisfied. For example, the following interior Holmholtz/Poisson equation with a Neumann boundary condition

$$\begin{aligned} &\Delta u + ku = f, && \text{in} \quad \Omega, \\ &u_n = q, && \text{on} \quad \partial\Omega. \end{aligned} \tag{5.10}$$

It may be treated as the following interface problem

$$\begin{aligned}
&\Delta u = \begin{cases} f, & if\ \ (x,y,z) \in \Omega, \\ 0, & if\ (x,y,z) \notin \Omega, \end{cases} \\
&[u] = g, \qquad \text{on} \quad \partial\Omega, \\
&[u_n] = q, \qquad \text{on} \quad \partial\Omega, \\
&u = 0. \qquad \text{on} \quad \partial\text{R}.
\end{aligned} \tag{5.11}$$

Again, the solution u is a linear functional of g. We determine $g(s_1, s_2)$ such that the solution $u(g)$ satisfies the original boundary condition in (5.10) above. i.e., $u_n(g) = q$. This can be solved using the GMRES iteration exactly as we discussed earlier in Chapter 3.

Chapter 6

Conclusions and future work

6.1 Research Conclusions

In this dissertation, we described a numerical method for 3D elliptic interface problems in which the β coefficient, the source term, the solution and its derivatives, have a discontinuity across the interface Γ. The fast solver can only be applied to the Poisson problems with piecewise constant coefficients. The number of iterations is nearly independent of the mesh size and the β coefficients jump. More importantly, the computed normal derivative from each side of the interface Γ appear to be second order accurate. The fast solver can be applied to Holmholtz/Poisson problems on irregular domains which may have many applications. In detail, we have presented the augmented approaches for solving 3D elliptic interface problems and problems defined on 3D irregular domains. Using augmented approaches, one or several augmented variables are introduced along a co-dimensional interface or boundary. When the augmented variable(s) is known, we can solve the governing PDE efficiently. In the discrete case, this gives a system of equations for the solution with given augmented variable(s). However, the solution that depends the augmented variable(s) usually do not satisfy all the interface relations or the boundary condition. The discrete interface relation or the boundary condition forms the second linear system of equations for the augmented variable whose dimension is much smaller than that of the solution to the PDE. Therefore, we can use GMRES iterative method to solve the Schur complement system for the augmented variable(s). In many instances, the GMRES method converge quickly, although it is still an open question how to precondition the system of linear equations without explicitly form a coefficient matrix.

By using the level set method as we discussed in chapter 2 how to determine geometric information of the interface, how to locate the projection of an irregular grid point on the interface, how to reconstruct the interface by interpolation, and how to perform local coordinate transformation.

Then in chapter 3, based on the IIM proposed by LeVeque and Li, 1994,[27] we have developed our 3D augmented approach which is second order fast algorithm for elliptic interface problems with piecewise constant but discontinuous coefficients. Before applying the IIM, we precondition the PDE first. In order to take advantage of existing fast Poisson solver on cubic domains, an intermediate unknown function, the jump in the normal derivative across the interface, is introduced. Then the GMRES iteration is employed to solve the Schur complement system derived from the discretization. Numerical experiments showed that the fast algorithm was very successful and efficient when the coefficients are piecewise constant.

In chapter 5, as apart of the dissertation, we have investigated some applications of the fast 3D solver. We developed embedding techniques to solve interior or exterior Poisson equations on complicated regions with Dirichlet or Neumann boundary conditions. The idea is to embed the irregular region into a cube to extend the Poisson equation to the entire cubic domain to introduce suitable jump conditions and to get an interface problem. In the future, we hope to develop further analysis of the augmented fast IIM approaches.

6.2 Future work

In the future we plan to conduct further numerical investigations about other boundary conditions, i.e., Neumann and Robin boundary conditions. Although the main theoretical analysis is presented in chapter 3, the change of boundary condition may requires many subroutines and functions due to the nature of our interface, which demands more time and energy. Additional test examples in 3D space are also needed. We have so far explored large enough ranges of coefficient jumps from 1 up to 2000 and large enough mesh size from 26 up to 104. In many instances, the GMRES method converge quickly, although it is still an open question how to precondition the system of linear equations without explicitly form a coefficient matrix. Also, I plan in future to test this augmented approach dealing with complicated interfaces.

There are other plan of work can be considered, maybe in a long term effect. There are high convergence order IIM existing for 3D Poisson problems with discontinuous coefficients and solution. It would be an interesting challenge to extend our work to 3D biharmonic equations by adopting our 3D-IIM solver into our algorithm.

REFERENCES

[1] L.M. Adams. A multigrid algorithm for immersed interface problems. In Proceedings of Copper Mountain Multigrid Conference, NASA Conference Publication 3339, pages 1-14,1995.

[2] J. Adams, P. Swarztrauber, and R. Sweet. Fishpack. Efficient Fortran sub-programs for the solution of separable elliptic partial differential equations. http://www.cisl.ucer.edu/css/software/fishpack; http://netlib.org/fishpack/.

[3] I. Babuska. *Modeling, Mesh Generation, and Adaptive Numerical Methods for Partial Differential Equations*. Springer-Verlag, New York, 1995.

[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] J.B. Bell and D.L. Marcus. A second order projection method for variable-density flows. *J. Comput. Phy., 101:334-338, 1992.*

[6] R.P. Beyer. A computational model of the cochlea using the immersed boundary method. *J. Comput. Phys., 98:145-162, 1992.*

[7] 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. *Numer. Anal., 8:722-736, 1971.*

[8] J. W. Cahn and J.E. Taylor. Surface motion by surface diffusion. *Acta Metall. Mater, 42: 1045, 1994.*

[9] Y. C. Chang, T. Y. Hou, B. Merriman, and S. Osher. A level set formulation of Eulerian interface capturing method for incompressible fluid flows. J. *Comput. Phys.,124:449-464, 1996.*

[10] D. L. Chopp. Computing minimal surfaces via level set curvature flow. J. *Comput. Phy., 106:77-91, 1993.*

[11] A. J. Chorin. Numerical solution of the Navier-Stokes equations. *Math. Comp., 22:745-762, 1968.*

[12] J.W. Demmel*, Applied Numerical Linear Algebra*, SIAM, 1997.

[13] S. Deng. Immersed Interface Method for Three Dimensional Interface Problems and applications. J. *Comput. Phys., 184:215-243, 2003.*

[14] S. Deng, K. Ito, and Z. Li. Three dimensional elliptic solvers for interface problems and applications. J. *Comput. Phys., 184:215-243, 2003.*

[15] V. Faber, A. White, and R. Sweet. In extension of the implicit capacitance matrix algorithm or 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. IMACS, 1981.

[16] L. Fauci and C.S. Peskin. A computational model of aquatic animal locomotion. J. Comput. Phys., 77:85-108, 1988.

[17] R. Fedkiw, T. Aslam, B. Merriman, and S. Osher. A non-oscillatory Eulerian approach to interfaces in multimaterial flows (the Ghost Fluid Method). J. Comput. Phys., 152:1999, 457-492.

[18] R. Fedkiw, T. Aslam, and S. Xu. The ghost fluid method for deflagration and detonation discontinuities. *January 11, 2000*.

[19] R. Fedkiw and X. Liu. The ghost fluid method for viscous flows. In M. Hafez, editor, progress in Numerical Solutions of Partial Differential Equations. Archachon, France, 1998.

[20] L. Greengard and M. Moura. On the numerical evaluation of electrostatic fields in composite materials. Acta Numerica, pages 379-410, Cambridge University Press, Cambridge, UK, 1994.

[21] T. Hou, Z. Li, S. Osher, and H. Zhao. A hybrid method for moving interface problems with application to the Hele-Shaw flow. J. Comput. Phys., 134: 236-252, 1997.

[22] T. Y. Hou, J. S. Lowengrub, and M. J. Shelley. Removing the stiffness from interfacial flows with surface tension. J. Comput. Phys., 14:312-338, 1994.

[23] J. Hunter, Z. Li, and H. Zhao. Autophobic spreading of drops., J. Comput. Phys.,183:335-366, 2002.

[24] C. Johnson. Numerical Solutions of Partial Differential Equations by the Finite Element Method. Cambridge University Press, Cambridge, UK, 1987.

[25] R. L. LeVeque, Finite Difference Methods for Ordinary and Partial Differential Equations, SIAM, 2007.

[26] R. J. LeVeque and Z. Li. The immersed interface method for elliptic equations with discontinuous coefficients and singular sources. SIAM J. Numer. Anal., 31:1019-1044, 1994.

[27] Z. Li. The immersed interface Method- A Numerical Approach for Partial Differential Equations with Interfaces. Ph.D. thesis, University of Washington, 1994.

[28] Z. Li. A note on immersed interface methods for three-dimensional elliptic equations. Comput. Math. Appl., 31:9-17, 1996.

[29] Z. Li. A fast iterative algorithm for elliptic interface problems. SIAM J. Numer. Anal.,35:230-254, 1998.

[30] Z. Li. An overview of the immersed interface method and its applications. *Taiwanese J. Mathematics, 7:1-49, 2003.*

[31] Z. Li. IIMPACK, a collection of fortran codes for interface problems. Anonymous ftp at ftp.ncsu.edu /pub/math/zhilin/Package and http://www4.ncsu.edu/zhilin/IIM, last updated 2005.

[32] Z. Li and K. Ito. Maximum principle preserving schemes for interface problems with discontinuos coefficients. SIAM J. Sci. Comput., 23:339-361, 2001.

[33] Z. Li and K. Ito, The immersed interface method, Numerical solutions of PDEs involving interfaces and irregular domains, SIAM, 2006.

[34] Z. Li, H. Zhao, and H. Gao. A numerical study of electro-migration voiding by evolving level set functions on a fixed Cartesian grid. J. Comput. Phys., 152:281-304, 1999.

[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. The rapid evaluation of volume integrals of potential theory on general regions. *J. Comput. Phys., 100:236-245. 1992.*

[37] A. Mayo and Greenbaum. Fast parallel iterative solution of Poisson's and the biharmonic equations on irregular regions. SIAM J. Sci. Statist. Comput., 13:101-118, 1992.

[38] A. Mckenney, L. Greengard, and A. Mayo. A fast Poisson solver for complex geometries. *J. Comput. Phys., 118:348-355, 1995.*

[39] K. W. Morton and D. F. Mayers. Numerical solution of Partial Differential Equations. Cambridge University Press, Cambridge, UK, 1995.

[40] W. W. Mullins. Mass transport at interface in single component systems. Metall. Trans. A, 26:1917, 1995.

[41] S. Osher and R. Fedkiw. Level set methods: An overview and some recent results. *J. Comput. Phys., 169:272-288, 2001.*

[42] 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.

[43] C. S. Peskin. Numerical analysis of blood flow in the heart. J. Comput. Phys., 25:220-252, 1977.

[44] C. S. Peskin. Lectures on mathematical aspects of physiology. Lectures in Appl. Math.,19:69-107, 1981.

[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] Y. Saad and M. H. Schultz. GMRES: A generalized minimal residual algorithm for solving non-symmetric linear systems. SIAM J. Sci. Statist. Comput., 7:856-869, 1986.

[47] P. G. Saffman and G. I. Taylor. The penetration of a fluid into a porous medium or Hele-Shaw cell containing a more viscous liquid. Proc. Roy. Soc. Lond. Ser. A, 245:312-329,1958.

[48] J. A. Sethian. Level Set Methods and Fast Marching methods, 2nd edition, Cambridge University Press, Cambridge, UK, 1999.

[49] G. R. Shubin and J. B. Bell. An analysis of the grid orientation effect in numerical simulation of miscible displacement. Comput. Methods Appl. Mech. Engrg., 47:47-71, 1984.

[50] M. Sussman, P. Smereka, and S. Osher. A level set approach for computing solutions to incompressible two-phase flow. J. Comput. Phys., 114:146-159, 1994.

[51] A. N. Tikhonov and A. A. Samarskii. Homogeneous difference schemes. USSR Comput. Math. Math. Phys., 1:5-67, 1962.

[52] J. Zhu and J. A. Sethian. Projection methods couples to level set interface technique. J. Comput. Phys.,102:128-138, 1992.