Master's thesis

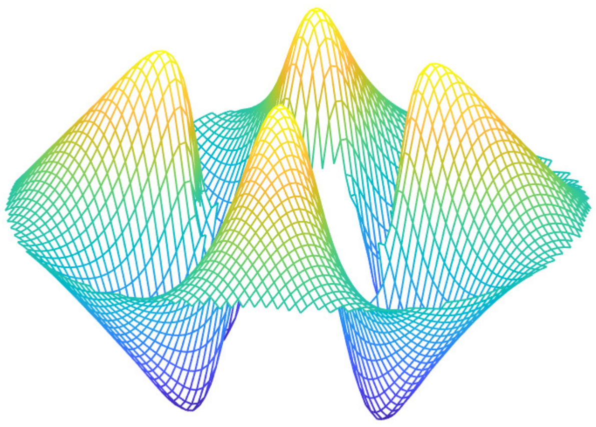
NTNU
Norwegian University of Science and Technology
Faculty of Engineering
Department of Energy and Process Engineering

Terje Tybring Lindtveit

Improvement of Immersed Boundary Method for Biofluid Dynamics

Forbedring av immersed boundary metode for biofluiddynamikk

Master's thesis in Mechanical Engineering
Supervisor: Bernhard Müller
June 2023

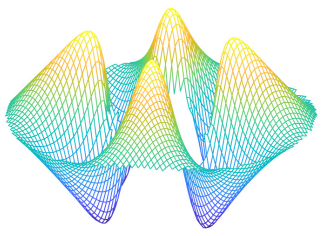


NTNU
Norwegian University of Science and Technology

Terje Tybring Lindtveit

Improvement of Immersed Boundary Method for Biofluid Dynamics

Forbedring av immersed boundary metode for biofluiddynamikk



Master's thesis in Mechanical Engineering
Supervisor: Bernhard Müller
June 2023

Norwegian University of Science and Technology
Faculty of Engineering
Department of Energy and Process Engineering

NTNU

Norwegian University
of Science and Technology

Department of Energy
and Process Engineering

EPT-M-2023

MASTER THESIS

for

Student Terje Tybring Lindtveit

Spring 2023

Improvement of Immersed Boundary Method for Biofluid Dynamics
Forbedring av immersed boundary metode for biofluiddynamikk

Background and objective

The immersed boundary method (IBM) has become an attractive alternative to body-fitted grid methods in Computational Fluid Dynamics (CFD), because the IBM alleviates grid generation. For in the IBM, the boundaries of rigid or flexible bodies or phases are immersed into Cartesian grids. However, the resolution of thin layers is a challenge for the IBM. Thus, the accuracy and efficiency of the IBM needs to be improved for better resolution of boundary layers and wakes.

The improvement of the IBM is motivated by the goal to accurately simulate flow and fluid-structure interaction (FSI) not only for flutter of wings and for flow-induced vibrations in power plants, but also for the flapping motion of the soft palate in the human pharynx. During sleep, the soft palate can make contact with the pharynx wall and lead to obstructive sleep apnea (OSA). Because of its great importance for public health, OSA is investigated in a larger interdisciplinary research project entitled “Virtual Surgery in the Upper Airways - New Solutions to Obstructive Sleep Apnea Treatment (VirtuOSA)”, which is funded by the Research Council of Norway.

The objective of the master’s project is to develop, implement and test an improvement of the immersed boundary method (IBM) for biofluid dynamics. The IBM will be used in VirtuOSA to simulate FSI in the upper airways of OSA patients. For testing the improved IBM, scalar benchmark test cases will be considered in 2D, e.g. for the 2D heat conduction equation. Accuracy and efficiency of the method will be assessed and compared with the standard IBM. The algorithm of the improved IBM is intended to be implemented and tested for the compressible Navier-Stokes equations. The master’s project will be a part of VirtuOSA.

The following tasks are to be considered:

1. to check the literature for improvements of the immersed boundary method (IBM),
2. to develop, implement and test improved IBMs for a 2D steady state scalar test problem,
3. to develop, implement and test improved IBMs for a 2D transient scalar test problem,
4. to develop, implement and test an improved IBM for a more complex 2D test problem.

-- ” --

Within 14 days of receiving the written text on the master thesis, the candidate shall submit a research plan for his project to his supervisor.

When the thesis is evaluated, emphasis is put on processing of the results, and that they are presented in tabular and/or graphic form in a clear manner, and that they are analyzed carefully.

The thesis should be formulated as a research report with summary in English, conclusion, literature references, table of contents etc. During the preparation of the text, the candidate should make an effort to produce a well-structured and easily readable report. In order to ease the evaluation of the thesis, it is important that the cross-references are correct. In the making of the report, strong emphasis should be placed on both a thorough discussion of the results and an orderly presentation.

The candidate is requested to initiate and keep close contact with his academic supervisor throughout the working period. The candidate must follow the rules and regulations of NTNU as well as possible directions given by the Department of Energy and Process Engineering.

Risk assessment of the candidate's work shall be carried out, in cooperation with the supervisor, according to the department's procedures. The risk assessment must be documented and included as part of the final report. Events related to the candidate's work adversely affecting the health, safety or security, must be documented and included as part of the final report. If the documentation on risk assessment represents a large number of pages, the full version is to be submitted electronically to the supervisor and an excerpt is included in the report. Those who have a theoretical exercise only need to check this and fill out page 1 of the form provided by the Department of Energy and Process Engineering.

Pursuant to “Regulations concerning the supplementary provisions to the technology study program/Master of Science” at NTNU §20, the Department reserves the permission to utilize all the results and data for teaching and research purposes as well as in future publications.

The master's thesis is to be submitted in NTNU’s examination system Inspera Assessment by 14:00 h on June 11, 2023.

☐ Work to be done in lab
☐ Field work

Department of Energy and Process Engineering, January 09, 2023

Bernhard Müller
Academic Supervisor

Abstract

In this thesis, a high-order sharp interface immersed boundary method for solving the heat conduction equation is presented. The approach is based on a second-order finite difference scheme with the immersed boundaries resolved through a sharp interface ghost point method. In this approach, the ghost point values are determined by a Weighted Least Squares technique in which the boundary condition is imposed at the immersed boundary with a high-order approximating polynomial. The main contribution of this thesis is the development of a method for achieving higher-order solutions through the use of Richardson extrapolation with a grid-doubling approach, thus avoiding the use of large ghost point stencils. The methods are verified for steady-state heat conduction problems with Dirichlet and Neumann boundary conditions. Furthermore, the verification is extended to time-dependent heat conduction cases by simulation of heat conduction in a cross-section of an infinite cylinder with constant surface temperature. The implementation of the method is shown to accurately resolve the boundary conditions and shows potential for application in areas of practical interest within the domain of engineering and biofluid dynamics. The future potential of the method and possible improvements are discussed.

Sammendrag

I denne masteroppgaven presenteres en høyordens skarp grenses "immersed boundary" metode utviklet for varmeligningen. Metoden er basert på en andre ordens "finite difference" løser hvor geometriene er representert ved hjelp av en spøkelsespunkt metode. I denne prosedyren bestemmes spøkelsespunktene ved hjelp av en minste kvadraters metode der randbetingelsen på geometrien blir tilnærmet med et høyordens polynom. Den viktigste bidraget i denne oppgaven er utviklingen av en metode for å oppnå høyordens løsninger ved bruk av Richardson-ekstrapolasjon, som dermed unngår bruk av store spøkelsespunkt stensiler. Metodene blir verifisert for stasjonære varmeledningsproblemer med Dirichlet- og Neumann-randbetingelser. Videre utvides verifiseringen til tidsavhengige varmeledningstilfeller ved å simulere varmeledning i et tverrsnitt av en uendelig sylinder med konstant overflatetemperatur. Implementeringen av metoden viser stor grad av nøyaktighet for randbetingelsene og viser potensiale for anvendelse innenfor relevante områder for ingeniørfag og biofluid-dynamikk. Fremtidig potensiale for metoden blir diskutert, og mulige forbedringer er drøftet.

Acknowledgements

I would like to express my sincere appreciation to my supervisor, Professor Bernhard Müller, for his exceptional guidance and support throughout my specialization project, master's thesis, and the various courses and lectures I have had the privilege of attending. You have provided me with valuable insights into the fascinating methods of the CFD field, and I am truly grateful for the time and effort you have invested in helping me acquire the necessary knowledge and skills for success. Working with someone as knowledgeable and experienced as you has been an honor, and your mentorship has been invaluable to me. I will forever be grateful for this opportunity.

Contents

Acronyms

BC Boundary Condition.

BI Boundary Intercept.

CFD Computational Fluid Dynamics.

FDM Finite Difference Method.

FEM Finite Element Method.

FSI Fluid-Structure Interaction.

FTCS Forward Time Centered Space.

FVM Finite Volume Method.

GP Ghost Point.

GUI Graphical User Interface.

HDF5 Hierarchical Data Format version 5.

IB Immersed Boundary.

IBM Immersed Boundary Method.

IP Image Point.

OSAS Obstructive Sleep Apnea Syndrome.

PDE Partial Differential Equation.

WLSQ Weighted Least Squares.

Nomenclature

α	Thermal diffusivity
β	Bilinear interpolation coefficient vector
Δl	Length of image point normal probe
ΔT	Temperature error
Δt	Time step size
ΔT_{err}	Temperature delta from exact solution
Δx	Grid spacing in x-direction
Δy	Grid spacing in y-direction
ϕ	Generic flow parameter
ρ	Density
θ	Angle
ζ	Neumann boundary condition
a	Measure of area
c_p	Specific heat at constant pressure
d	Measure of distance
f	Generic grid solution
h	Grid spacing (uniform)
k	Thermal conductivity
k_d	Weight function scaling
L_2	L_2-norm
M	WLSQ weight matrix
N	Cell number in given dimension, grid size (uniform)

n	Normal
n_t	Time level
N_{GP}	Number of ghost points
P	Point location
p	Coefficient quantity, spatial order of accuracy
p	Coefficient quantity
q	WLSQ stencil size
r	Radius, refinement factor, von Neumann number, polynomial order, Richardson extrapolation
T	Temperature
t	Time
t_f	Simulation end time
t_i	Initial time
T_{ex}	Temperature of exact solution
V	Vandermonde matrix
W	Diagonal weight matrix
w	Weight coefficient
$x^{'}$	Local x coordinate relative to boundary intercept
$y^{'}$	Local y coordinate relative to boundary intercept

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation and Background

Computational Fluid Dynamics (CFD) has emerged as a powerful and transformative tool for analyzing fluid flow phenomena and in many fields it has revolutionized the way engineers and scientists study fluid flows. Today, CFD has found extensive applications in various domains, ranging from biomedical research, mechanical engineering, and aerospace to weather and climate modeling. Within the realm of CFD, special attention has been given to flows exhibiting Fluid-Structure Interaction (FSI) phenomena due to their significance in numerous biological and engineering systems. These systems often involve intricate dynamic interactions between fluid flow and moving or deforming boundaries, resulting in complex physical phenomena. Accurate understanding and modeling of FSI is crucial for a wide range of applications, including flexible pipes, turbines, aeroelastic phenomena, and biofluid dynamics.

In biofluid dynamics, one such area of research is the study of respiratory disorders in the human upper airways such as Obstructive Sleep Apnea Syndrome (OSAS). OSAS is a common sleep disorder characterized by repeating partial or complete obstruction of the upper airway during sleep. These episodes known as apneas occur when muscles supporting the soft tissue relax to a degree in which the tongue and soft palate collapse the airway, and are closely related to the flow conditions in the airways [1].

For this reason, understanding the airflow of the respiratory system will be critical in assisting medical professionals to improve and perform effective surgery for patients. For its importance to public health, OSA is being investigated in an interdisciplinary research project entitled "Virtual Surgery in the Upper Airways - New Solutions to Obstructive Sleep Apnea Treatment" (VirtuOSA) funded by the Research Council of Norway. As part of the project, research has been conducted on modeling physiological effects in the upper airways during respiration using, in part, using Immersed Boundary Method (IBM) for the moving boundaries resulting from the FSI phenomena occurring in the airway. [2]

Conventional approaches used in CFD for FSI problems typically rely on body-conforming meshes in order to accurately resolve the fluid-structure boundary. However, as the solid boundaries undergo motion or deformation, the need to regenerate or update the body-conforming mesh arises. This process is quite computationally expensive, and when dealing with highly dynamic boundaries or when employing higher-order methods it becomes increasingly challenging to ensure the well-posedness of these algorithms [3]. These limitations have inspired a growing interest in non-body conforming approaches like the immersed boundary method, which eliminates the need for mesh regeneration and may significantly reduce the associated costs related to updating the boundary geometry. Nevertheless, this alternative method introduces its own set of challenges that require careful consideration and resolution.

This thesis builds on concepts and methods explored in the author's project work [2], and the following contents of this chapter are revised versions of said project work with a number of alterations adjusted for the new direction of this thesis.

1.2 Literature Review

The term Immersed Boundary Method (IBM) is originally a reference to the method used by Peskin which he presented in the paper "Flow patterns around heart valves: A numerical method", where it was used to model elastic tissue fluid-structure interactions of mitral valves in the human heart. Since then a range of derivative modifications and approaches have been developed. Building upon this concept, Goldstein et al. [4] developed a method for rigid bodies using feedback forcing. A general overview of immersed boundary methods is presented in [3], where the immersed boundary term is extended to also include methods developed under the term "Cartesian grid methods", and methods with similar capabilities for viscous flows with immersed boundaries on grids that do not conform to the boundary shapes. In this thesis, the same extended definition will be used but in relation to problems not confined to fluid flows.

According to Mittal and Iaccarino [3] and Khalili et al. [5], the immersed boundary method can be broadly classified into two main categories based on the procedure by which the boundary condition is imposed. Namely continuous forcing and sharp interface (discrete forcing) approaches. Methods such as that of Peskin [6] and Goldstein et al. [4], whereby the boundary condition is imposed by modification of the modeling equations with a forcing term and applied to the whole domain, fall into the first category. The main advantage of the continuous forcing methods is that they are independent of spatial discretization. However, some drawbacks are their instabilities for high Reynolds number flows with rigid boundaries due to the inherent stiffness imposed on the forcing terms, particularly for unsteady flow [7][4]. Additionally, the force distribution function can introduce significant smearing over the nodes surrounding the boundary leading to reduced accuracy [3].

To avoid the issue of modeling the forcing term, Mohd-Yusof [7] developed a method that extracts the forcing from the numerical solution. In this approach, the forcing term is computed at each iteration by enforcing the tangential velocity at the boundaries using the velocity difference of the external points to their mirrored velocities internally. The advantages of this method are that of removing the instabilities and strict time-step limitations of the method developed by Goldstein et al. [4], and the absence of user-specified parameters in the forcing terms. However, the forcing distribution still extends into the fluid domain which can significantly degrade the accuracy near the boundaries [3].

A sharp-interface method, based on the ghost point Finite Difference Method (FDM) approach, which imposes the Boundary Condition (BC) by reconstruction at the ghost points, was developed by Tseng and Ferziger [8]. Where ghost points are defined as points on the Cartesian grid located within a boundary with their neighbors in the fluid domain. In this method, the issue of the force distribution function smearing is mitigated. Furthermore, the method has been shown to have large potential for highly complex bodies with moving boundaries and high Reynolds number flows, in which the need to accurately resolve boundary layers puts greater emphasis on local accuracy.

Many variations exist, and a range of reconstruction possibilities of varying accuracy is presented in [9]. These methods have been employed by Tseng and Ferziger [8] and were also used by Ghias et al. [10] for a range of subsonic compressible flows. In [11], these principles are built upon for a fast and efficient solver for flows with complex three-dimensional moving boundaries. Using similar concepts, a high-order method for low Mach number compressible flows with acoustic wave propagation is presented in [5]. Capturing acoustic wave effects may be of particular interest for the simulation of the upper airways and OSAS, where acoustics could have an important role in the dynamics.

In [12][13][1] higher-order reconstruction schemes are employed on the boundaries using a Weighted Least Squares (WLSQ) method. Higher-order reconstruction approaches for the ghost point method often require large stencils into the fluid domain, which might lead to exacerbating the issues of the interpolation stencil intersecting nearby boundaries [12]. With the WLSQ method as implemented in [12][13] this problem is somewhat mitigated, as it can inherently adjust to capture fluid points elsewhere for the ghost point reconstruction. Though this may still have a negative effect on the accuracy of the solution. Additionally, for thin geometries, the higher-order ghost point stencils into the solid domain may not be possible with conventional methods.

The use of Richardson extrapolation to achieve an improved, higher-order solution for the immersed boundary method has not been extensively emphasized in previous research on immersed boundary methods. The Richardson extrapolation is a method in which originally second-order centered difference solutions of Partial Differential Equations

(PDE) are combined to obtain a higher-order estimation of the exact solution. In the paper "The approximate arithmetical solution by finite differences of physical problems involving differential equations, with an application to the stresses in a masonry dam" [14], Richardson applied the method to achieve fourth-order accuracy for the Laplace equation.

The method has been demonstrated also for higher-order approximation such as sixth-order using three grid solutions, though the focus here will be on the fourth-order approximation with grid doubling. In the original method by Richardson [14] the second-order central differences are used exclusively, and as further emphasized in [15] this is in fact a requirement of the method for achieving fourth-order accuracy. In [14], Richardson noted that it was conceivable that the extrapolation would be extended to a continuous solution for the whole domain. Later, in [16] a method is presented for which the correction terms are interpolated and fourth-order accuracy is also achieved on the fine grid solution. The main motivation for the use of Richardson extrapolation on the sharp interface ghost point IBM is the potential to limit the geometric restrictions of higher-order ghost point interpolation stencils into the solid domain.

1.3 Objective

The objective of this thesis is to implement and verify a sharp interface ghost-point immersed boundary method for the two-dimensional heat conduction equation for both steady and unsteady problems of heat flow. Following this an "Improved IBM" approach for increasing the accuracy of the WLSQ IBM based on Richardson extrapolation will be investigated. While the verification is limited only to heat conduction, a suggested implementation for the Navier-Stokes equations, relevant to biofluid dynamics and the work of VirtuOSA, will be presented.

A real-time application for running simulations featuring a Graphical User Interface (GUI) is developed, intended to provide an example featuring tools to debug and investigate the behavior of the implementation. The application is written in C++ using common open-source libraries for data logging, math, and graphics which are further specified in appendix C. The application also includes post-processing scripts and material used for various parts of this thesis.

The contents herein are written in the context of its application to fluid mechanics. As such, the terminology that will be used throughout is centered around fluid flows, even though many concepts detailed in this report have broader applications in science and engineering.

1.4 Outline

The report is organized as follows: In section 2 the governing equation of heat conduction is formulated. Section 3 presents the numerical method for the discretization of the heat equation and a detailed formulation of the immersed boundary methods and Improved IBM. This section also includes a short description of how the IBM could be applied to the Navier-Stokes Equation and definitions for measuring the discretization errors. The benchmark problems are used for verification and development and are presented in section 4. In section 5 the performance and complexity of the methods and implementations are discussed. Results of the WLSQ IBM and Improved IBM are presented and discussed in section 6. Conclusions are given in chapter 7. Finally, section 8 discusses the future outlook for the methods detailed herein.

Chapter 2

Governing Equations

In this study, the two-dimensional heat conduction equation will be considered. The following chapter is a revised version of that given in [2]. The two-dimensional heat conduction equation is a fundamental Partial Differential Equation (PDE) for heat and fluid flow which models many physical processes [17] and is a convenient starting point for the study on improvements to finite difference IBM methods.

2.1 Heat Conduction Equation

The heat conduction equation can be formulated as

$$\frac{\partial T}{\partial t} = \alpha \nabla^2 T \,, \tag{2.1}$$

where T is temperature, α thermal diffusivity of the medium defined as

$$\alpha = \frac{k}{\rho c_p} \,, \tag{2.2}$$

where k is the thermal conductivity, ρ is density and c_p is specific heat at constant pressure. For the two-dimensional isotropic and homogenous case in Cartesian coordinates, eq. (2.1) can be written as

$$\frac{\partial T}{\partial t} = \alpha \left(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} \right) \,. \tag{2.3}$$

The heat conduction equation requires both initial conditions for the domain and boundary conditions to be provided. The boundaries are typically either of the Dirichlet or Neumann boundary condition types. Other options such as Robin boundary conditions are not considered in this study. For the Dirichlet condition, the temperature T is directly described at the boundary, while for the Neumann condition, the normal temperature gradient $\frac{\partial T}{\partial n}$ is given at the boundary.

Chapter 3

Discretization

This chapter presents the formulations by which the governing equations and immersed boundaries are discretized. For spatial discretization, the ghost-point Finite Difference Method (FDM) will be used. Other common approaches in CFD include the Finite Volume Method (FVM) and Finite Element Method (FEM). The contents of this chapter, and in particular sections 3.1, 3.2.2, 3.3 are revised versions of the author's previous work [2], with modifications and new material added.

3.1 Forward Time Centered Space

The solution domain is discretized by finite differences using the forward Euler method in time and central differences in space, also known as Forward Time Centered Space (FTCS) discretization as introduced by Roache [18].

From the heat conduction equation (2.3) the discretized form by FTCS is

$$T_{i,j}^{n+1} = T_{i,j} + r_x \left(T_{i+1,j}^n - 2T_{i,j}^n + T_{i-1,j}^n \right) + r_y \left(T_{i,j+1}^n - 2T_{i,j}^n + T_{i,j-1}^n \right) , \tag{3.1}$$

where n is the time level and $T_{i,j}^n$ is the FDM approximation of the exact solution $T(x_i, y_j, t_n)$, where $x_i = i\Delta x$, $y_j = j\Delta y$, $t_n = n\Delta t$. Δx and Δy are the constant grid spacings, and Δt the time step size. The von Neumann numbers are defined by

$$r_x = \frac{\alpha \Delta t}{\Delta x^2}\,, \qquad r_y = \frac{\alpha \Delta t}{\Delta y^2}\,. \tag{3.2}$$

The heat conduction equation discretized by the FTCS scheme has the following stability criterion:

$$r_x + r_y \leq \frac{1}{2}\,. \tag{3.3}$$

3.2 Immersed Boundary Method

3.2.1 Weighted Least Squares Method

In this section, a sharp-interface ghost-point IBM method utilizing a WLSQ approach for ghost-point approximation will be presented. The fundamental idea of this approach, as presented in [3][13], is to impose the boundary conditions of the IB on the fluid by a layer of ghost points similar. As such, no modifications to the discretization scheme is required. For a second-order scheme, one layer of ghost points is needed when separating the fluid and solid domains across the IB. The ghost points for this approach are defined as grid nodes within the solid domain with at least one neighbor in the fluid [3]. Following [12][13], the boundary condition is then imposed on the ghost point values by high-order polynomial interpolation with weighted least square error minimization. The procedure of immersing boundaries and separation of fluid and solid domains is shown in figure 3.1, along with normal vector, Boundary Intercept (BI), and WLSQ stencil. Note that the WLSQ stencil is centered at the nearest fluid node to the BI.

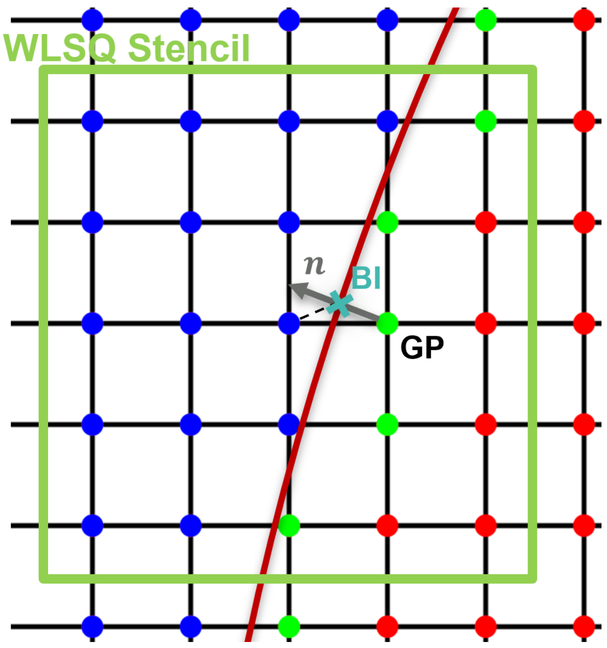


Figure 3.1: Schematic of the immersed boundary flagging process including an illustration of ghost point normal vector, BI and WLSQ stencil for a ghost point adjacent to the boundary. Solid (inactive) points are shown in red, with ghost points in green and fluid points shown in blue.

For a generic variable, ϕ, approximated in the region around the body intercept with relative coordinates $x^{'} = x - x_{BI}$, $y^{'} = y - y_{BI}$ the method follows:

$$\phi(x^{'}, y^{'}) \approx \sum_{i=0}^{r}\sum_{j=0}^{r} C_{i,j} x^{'i} y^{'j} \,, \qquad\qquad i + j \leq r\,. \tag{3.4}$$

where $C_{i,j}$ are the coefficients and r is the polynomial order. As shown in [12], equation (3.4) is the Taylor series expansion of ϕ across the boundary intercept location.

From the WLSQ formulation, [12][13] show the coefficients can be determined by minimizing the weighted error for q data points in the following manner

$$\min_{c} \sum_{n=1}^{q} \left[w_n \left(V_n C - \phi_n\right)\right]^2 \,, \tag{3.5}$$

where n is the nth data point and w_n is the corresponding weight. V_n is the nth row of the Vandermonde matrix V on the form $V = \{V_1^T, V_2^T, ..., V_q^T\}^T$ with

$$V_n = \left\{1, x^1y^0, x^0y^1, ..., x^{r-1}y^0, x^0y^{r-1}, ..., x^{r-2}y^{r-1}, x^{r-1}y^{r-2}, x^ry^0, x^0y^r\right\}. \tag{3.6}$$

And C is the coefficient vector defined by eq. (3.4) on the following form

$$C = \left\{C_{0,0}, C_{1,0}, C_{0,1}, ..., C_{r-1,0}, C_{0,r-1}, ..., C_{r-2,r-1}, C_{r-1,r-2}, C_{r,0}, C_{0,r}\right\}, \tag{3.7}$$

such that the vector product V_nC in eq. (3.5) is the approximate solution of ϕ from eq. (3.4) for data point n [13].

In previous papers, the choice of weight function varies. In [12] the weights are given by a cosine weight function, while in [19] inverse distance weighting is used. In this thesis, a variation of the exponential weighting function given in [13] is employed, given by the following definitions

$$w_n = e^{-\frac{d_n^2}{a_d}}, \qquad a_d = k_d \sum_{n=1}^{q} \left({x'_n}^2 + {y'_n}^2\right). \tag{3.8}$$

Where $d_n = \sqrt{{x'_n}^2 + {y'_n}^2}$ is the distance from the nth data point to the boundary intercept and a_d defines a measure of the area covered by the data points q of the numerical stencil. Adjustment of the weight function distribution is achieved by changing the value of k_d.

The number of polynomial coefficients, p, to determine is a function of the polynomial order r, and is listed in table 3.1 for the two-dimensional case with $r = 1, \dots, 4$. In order to determine the coefficients, a suitable number of data points around the body intercept are needed. This region of points should be chosen such that ideally all values are strongly correlated with the ghost point. Additionally, in order to avoid the WLSQ problem becoming ill-posed the number of points within the region must be greater than the number of coefficients, such that $q > p(r)$.

Equation (3.4) is solved not only for the fluid data points but also for the boundary intercept and corresponding ghost point. The data is structured such that the first entry is given by the ghost point and with subsequently $q - 1$ data points in the surrounding fluid domain following.

Table 3.1: Number of coefficients $p(r)$ for polynomial order r in 2D.

r	$p(r)$
1	3
2	6
3	10
4	15

Some examples of circular (or spherical in 3D) search regions and their construction algorithms can be found in [13][12][19]. However, in this thesis, a square region is used for simplicity. The reasoning for this is tied to the assumption that for a large stencil, the weight function will essentially redefine the stencil back into a circular shape. This requires oversizing the stencil, which is not good for performance, but assuming that this holds true it saves some implementation complexity from the implementation. With the shape defined, the size of the region must be determined in such a way that the M matrix, is well-conditioned. In previous studies, several algorithms for this procedure have been presented. In [12], the size of the search region is adaptively chosen such as to get a well-conditioned matrix. In [13][19] similar methods are used with rank-based stencil selection to determine which points are added. In this work, the number of points to be used for the minimization problem is given as a predetermined value. From there, a subgrid centered around the fluid node closest to the boundary intercept is iteratively expanded until the required fluid points are captured.

The exact solution to the WLSQ problem of eq. (3.5) for the vector of coefficients C is given by

$$C = (WV)^{+} W\phi \tag{3.9}$$

where ϕ is the vector of data point values $\phi = \left\{\phi_0, \phi_1, ..., \phi_q, \right\}$ and superscript '+' denotes the pseudo-inverse of a matrix [12]. W is the diagonal weight matrix and V is the Vandermonde matrix. For $r = 1$, V reads

$$V = \begin{bmatrix} 1 & x|_1 & y|_1 & xy|_1 \\ 1 & x|_2 & y|_2 & xy|_2 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x|_q & y|_q & xy|_q \end{bmatrix} . \tag{3.10}$$

With W being $q \times q$ and V being $q \times p$ matrices. Singular Value Decomposition (SVD) is used to calculate the $(WV)^+$ pseudo-inverse in eq. (3.9). Further, by defining and storing a weight matrix

$$M = (WV)^+ W \tag{3.11}$$

with dimensions $p \times q$ such that $C = M\phi$, it is now readily seen that the coefficients $C_{i,j}$ are a linear combination of ϕ_n which can be written in the form

$$\begin{aligned} C_{0,0} &= \sum_{n=1}^{q} M(1,n) \cdot \phi(x_n^{'}, y_n^{'}) \\ C_{1,0} &= \sum_{n=1}^{q} M(2,n) \cdot \phi(x_n^{'}, y_n^{'}) \\ C_{0,1} &= \sum_{n=1}^{q} M(3,n) \cdot \phi(x_n^{'}, y_n^{'}) \\ &\vdots \end{aligned} \tag{3.12}$$

From eq. (3.4), knowing that the coefficients are those of the Taylor series expansion at the boundary intercept, the values and derivatives at the boundary intercept are given by

$$C_{0,0} = \phi(x_{BI}, y_{BI}) \qquad C_{1,0} = \frac{\partial \phi}{\partial x}(x_{BI}, y_{BI}) \qquad C_{0,1} = \frac{\partial \phi}{\partial y}(x_{BI}, y_{BI}) \tag{3.13}$$

Therefore for a given Dirichlet or Neumann boundary condition, the ghost point value can be expressed by the result of equations (3.12) and (3.13) [12]. For a Dirichlet condition at the boundary, $\phi(x_{BI}, y_{BI}) = \phi_{BI}$, the ghost-point value is computed from

$$\phi_{GP} = \frac{\phi_{BI} - \sum_{n=2}^{q} M(1,n) \cdot \phi(x_n^{'}, y_n^{'})}{M(1,1)} . \tag{3.14}$$

With a Neumann boundary condition $\frac{\partial\phi}{\partial n}(x_{BI}, y_{BI}) = \zeta$ the ghost point value is given by

$$\phi_{GP} = \frac{\zeta - \sum_{n=2}^{q}(n_x M(2,n) + n_y M(3,n)) \cdot \phi(x_n^{'}, y_n^{'})}{n_x M(2,1) + n_y M(3,1)}\,, \tag{3.15}$$

with n_x, n_y the components of the normal vector at the boundary intercept.

The WLSQ method of boundary reconstruction provides a high degree of flexibility with respect to geometry and choice of the numerical stencil for the accuracy achieved. While it requires a significantly larger numerical stencil than the image point method, it retains the flexibility of conforming to any number of boundaries which it may encounter within its search region. The method also extends easily to higher-order reconstruction. The key element to successfully implementing the method is ensuring the well-posedness of the least squares problem. A convenient measure to ensure this is to monitor the condition number of matrix M. If the condition number becomes large (e.g. $O(10^6)$ or larger), then eq. (3.9) becomes very sensitive to input values such that small numerical errors of the ghost point values can amplify and lead to numerical instability [12]. Increasing the numerical stencil q to include more fluid points in the domain is then required in order for the method to remain stable. Typically, the MV matrix is singular meaning that usually one cannot get a well-posed WLSQ problem with the minimum required stencil size [12].

Because of the user-specified parameters, the WLSQ IBM requires some additional set-up work. The procedure which is followed in this thesis can be summarized generally in the following steps (usually in order):

1. Determine the polynomial order r of eq. (3.4) for the reconstruction.
2. Determine the number of points q to include in the WLSQ problem.
3. Adjust the weight function scaling k_d, optimizing for the best fit of eq. 3.4 with regards to boundary reconstruction accuracy for a given stencil size q.

As such this implementation of the WLSQ IBM introduces significant complexity in terms of configuration and optimization compared to the IP IBM which does not require any set-up. For each choice of polynomial order r, steps 2 and 3 must be performed to optimize the results, or indeed even find a stable solution. The approach which was employed herein for a given polynomial order r was to initially select a large stencil size q in step

2 of the procedure. This way the weight scaling k_d of step 3. can be optimized alone with respect to accuracy and stability. Once k_d is determined, the stencil size q may be reduced approaching limits given in table 3.1 (with k_d increased proportionally) such that a number of "excess" points which would be given effectively zero weight are excluded.

The computational cost of the method is, like the image point method, dependent on the particular geometries modeled along with method order r, and stencil size q. With stationary boundaries, the method is performant even for large numerical stencils, as the main computational expense is performed ahead of time and then stored in the matrix M. The operations that are performed for each ghost point are then limited to highly performant vector operations [12]. However, if the boundaries are changing (e.g. FSI), the cost of the method will be much greater as the weight matrix $M = (MV)^{+}W$ must again be computed. A more detailed analysis of performance is presented in chapter 5.

3.2.2 Image Point Method

This section describes the sharp-interface image point IBM method utilized in the author's project work [2] which will be referred to for comparison with the WLSQ IBM method. The fundamental approach is the same as for the WLSQ approach, and they share identical formulations for determining ghost points where all grid nodes of the domain are flagged according to whether they are inside any solid domain, as illustrated in figure 3.2 by Khalili et al. [13].

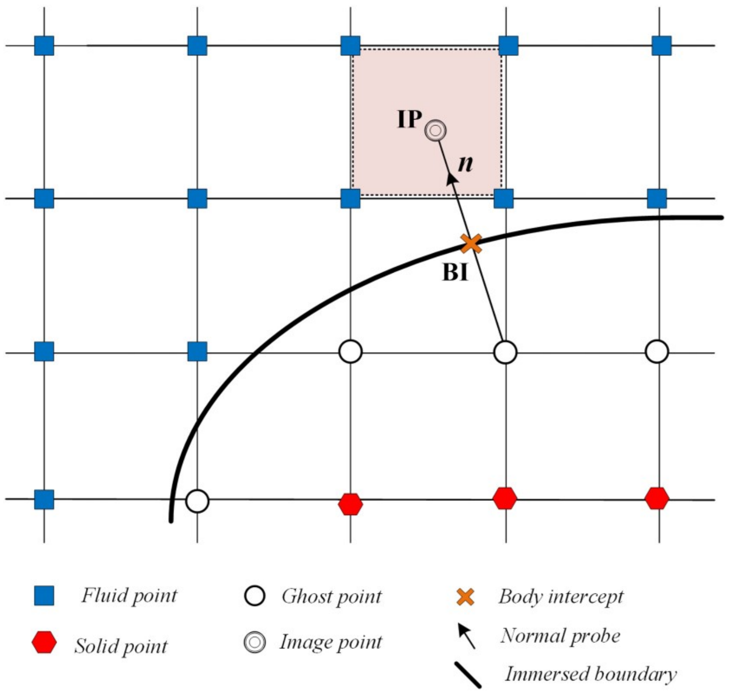


Figure 3.2: Schematic of points used to determine the flow variables at a ghost point adjacent to an immersed boundary. [13]

With the image-point approach, the values at the ghost point are determined by the boundary conditions and interpolated values in the fluid domain. An Image Point (IP) is introduced, defined as the mirror point of the GP, normal to the IB. Its stencil intersects the immersed boundary defining the Boundary Intercept (BI) location at its half-length [5]. When the BI and corresponding IP have been determined, expressing the value at the IP for a generic flow variable ϕ is achieved using bilinear interpolation [11]. The bilinear interpolating polynomial needs four nodal values surrounding the image point and takes the following form [11][13]:

$$\phi\left(x,y\right) = C_1 xy + C_2 x + C_3 y + C_4 \,, \tag{3.16}$$

where the four unknown coefficients are determined by the Vandermonde matrix and surrounding nodal values:

$$\mathbf{C} = \mathbf{V}^{-1}\phi \,, \tag{3.17}$$

with the values of the four surrounding nodes $\phi = \left[\phi_1, \phi_2, \phi_3, \phi_4\right]^T$, and Vandermonde matrix of the form

$$\mathbf{V} = \begin{bmatrix} xy|_1 & x|_1 & y|_1 & 1 \\ xy|_2 & x|_2 & y|_2 & 1 \\ xy|_3 & x|_3 & y|_3 & 1 \\ xy|_4 & x|_4 & y|_4 & 1 \end{bmatrix} . \tag{3.18}$$

In the case of a sample point being located on a boundary with Neumann BC given by ζ, the normal gradient $\frac{\partial\phi(x_{BI},y_{BI})}{\partial n} = \zeta$ is instead used [13]. Taking the normal derivative of eq. (3.16) gives then

$$\zeta = C_1(yn_x + xn_y) + C_2 n_x + C_3 n_y \, . \tag{3.19}$$

As such the row corresponding to the sample point in the Vandermonde matrix is replaced by

$$[(yn_x + xn_y), n_x, n_y, 0] \ . \tag{3.20}$$

Following formulations in [11][13], eq. (3.16) and eq. (3.17), the image point value ϕ_{IP} can be expressed as

$$\phi_{IP} = \sum_{i=1}^{4} \beta_i \phi_i + T.E. \tag{3.21}$$

where β is the coefficient vector given by

$$\beta = \left(\mathbf{V}^{-1}\right)^T [xy|_{IP}, x|_{IP}, y|_{IP}, 1]^T \tag{3.22}$$

such that the coefficients are only dependent on the geometric coordinates. Consequently, they can be determined once the grid, boundaries, and image points are determined, and will only need updating if these parameters are changed. With stationary boundaries and a static grid, the main computational cost can therefore be performed in the initialization step.

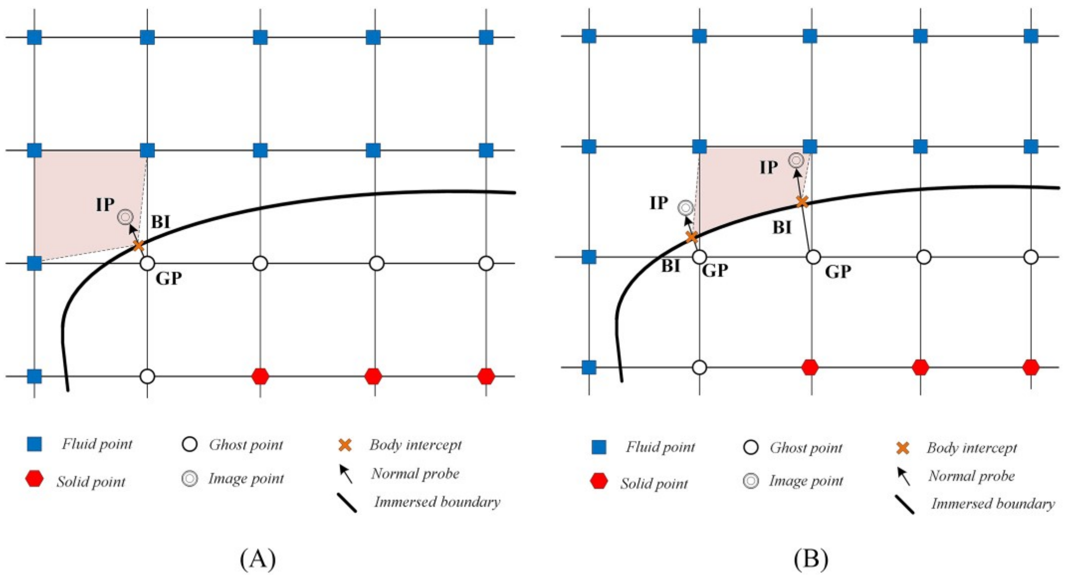


Figure 3.3: A, Schematic of the situation when one surrounding interpolation point is a body intercept; B, Schematic of the situation when two of the surrounding interpolation points are body intercepts. [13]

In some situations, the nodes surrounding the image point may be ghost points, as seen in figure 3.3 by Khalili et al. [13]. In these cases, the nodes chosen for the interpolation polynomial are then replaced by the boundary condition at the corresponding BI location following [11].

Finally, obtaining the value at the ghost point is achieved with a linear extrapolation of the IP and GP values using the boundary conditions. For a Dirichlet BC, this can be expressed as

$$\phi_{GP} = 2\phi_{BI} - \phi_{IP} + \mathcal{O}\left(\Delta l^2\right) , \tag{3.23}$$

where Δl is the length of the normal stencil from GP to IP [13]. For a Neumann BC the ghost point is computed by

$$\phi_{GP} = \phi_{IP} - \Delta l \zeta + \mathcal{O}\left(\Delta l\right) , \tag{3.24}$$

where the normal gradient given by the BC is represented by ζ. This computation of the ghost point values is according to [11] 2nd-order for the Dirichlet BC and 1st-order for the Neumann BC in respect to Δl and therefore also in relation to the grid spacing given that $\Delta l = \mathcal{O}(h)$.

3.3 Improved IBM

The Improved IBM approach herein is a Richardson extrapolation method for which the main goal is to achieve higher-order solutions without the need for large ghost-point stencils. Richardson extrapolation is a technique used to improve the accuracy of numerical approximations of solutions to differential equations. It is commonly used in CFD to improve the accuracy of numerical solutions to the equations governing fluid flow. The technique involves using solutions at different levels of refinement and combining them using a weighted average to produce a more accurate approximation. This can be particularly useful when solving complex problems involving fluid flow, where a high degree of accuracy is required.

The method as originally presented takes separate 2nd-order solutions using a grid doubling method, using a fine grid and a coarse subgrid defined of alternate points, combining them to obtain a 4th-order solution defined on the subgrid [16].

For two uniformly spaced grids with a grid spacing h and $2h$, let the numerical solutions given by eq. (3.1) be T_h and T_{2h} respectively. For a constant von Neumann number, with time step Δt on the fine grid, it follows from eq. (3.3) that the coarse grid has a time step of $\frac{1}{4}\Delta t$. Knowing the truncation error of eq. (3.1) can be written (in simplified form) for the two grids as

$$T_h = T_{ex} + \frac{1}{2}\Delta t - \frac{1}{12}h^2 + \mathcal{O}(\Delta t^2, h^4) \tag{3.25}$$

$$T_{2h} = T_{ex} + 2\Delta t - \frac{1}{3}h^2 + \mathcal{O}(\Delta t^2, h^4)\,, \tag{3.26}$$

where T_{ex} is represents the exact solution. And with every time level of the coarse grid representing four time levels on the fine grid, it follows that

$$T_r = \frac{4T_h - T_{2h}}{3} = T_{ex} + \mathcal{O}(\Delta t^2, h^4)\,. \tag{3.27}$$

Where T_r is the Richardson extrapolation of the fine and coarse grids at synchronized time levels at the coinciding node locations. As shown the Richardson extrapolation increases the order of the method to 2nd-order in time and 4th-order in space.

According to [15], the general form of the extrapolation for centered difference schemes and omitting the temporal component, using fine grid and coarse grid solutions f_1 and f_2 is:

$$f_{exact} = f_1 + \frac{f_1 - f_2}{r^p - 1} + \mathcal{O}(\Delta^{p+2})\,. \tag{3.28}$$

Where f_{exact} is the exact solution, p is the order of the method and r is the refinement factor defined by $r = h_2/h_1$. With grid doubling such that $r = 2$, and FTCS having spatial order $p = 2$, eq. (3.28) can be simplified to

$$f_{exact} = \frac{4}{3}f_1 - \frac{1}{3}f_2 + \mathcal{O}(\Delta^{p+2}) \tag{3.29}$$

which is 4th-order accurate for solutions f_1 and f_2 [15] and is equivalent to eq. (3.27). In either case, as described before, f_1 must be evaluated for alternate nodes, which coincide with nodes on f_2, as such the 4th-order accurate solution is only defined on the coarse grid solution. A method described as "Completed Richardson extrapolation" [16] shows how 4th-order can be achieved on the full fine grid solution, however, this is beyond the scope of this project.

In the program which has been developed, the Richardson extrapolation is applied purely as a post-processing step to the solutions either at steady state or at synchronized time levels of the simulation. Though there is also the potential for the method to be incorporated into the time step iteration, as has been presented by Richards [20]. In this case, the Richardson extrapolation for the fine grid is computed, and both fine and coarse grid solutions are updated by this extrapolation at each synchronized time level.

3.4 Considerations for the Compressible Navier-Stokes Equations

The implications of the WLSQ IBM for compressible Navier-Stokes equations are mainly boundary condition treatment for the velocity, pressure, and temperature. The immersed boundary method used herein has also been applied to the compressible Navier-Stokes equations in previous papers [13]. For walls the no-slip condition is typically applied, meaning a Dirichlet BC is imposed on the velocity. For the pressure, a zero gradient, $\frac{\partial p}{\partial n} = 0$, on the boundaries may be used to approximate the boundary layer. The temperature field requires that the wall has either a temperature or a heat flux specified, as such both Dirichlet and Neumann BC's can be used depending on the situation. But assuming adiabatic boundaries, the temperature BC can be enforced with a zero density gradient, $\frac{\partial \rho}{\partial n} = 0$. Thus the Dirichlet and Neumann boundary conditions would be applied the same way as detailed in section 3.2.1.

The Improved IBM can likewise be applied to the Navier-Stokes Equations in the same manner as described in section 3.3. While discretization methods and other factors can have an impact on the validity of eq. (3.29), its application is general to any differential equation and can be applied to the compressible Navier-Stokes equations without modification. A key limitation is that centered differences must be employed for 4th-order accuracy, if uncentered differences are used the extrapolation is only 3rd-order accurate [15].

3.5 Discretization Error

The errors in the discretization methods are monitored by observing errors on the form $\Delta T_{err} = T - T_{ex}$ and evaluating the L_2-norm of this field. The L_2-norm for a generic field ϕ is defined by

$$\begin{aligned} \|\phi\|_2 &= \sqrt{\Delta x \Delta y \sum_{i,j}^{N} \phi_{i,j}^2} \\ &= h\sqrt{\sum_{i,j}^{N} \phi_{i,j}^2} \end{aligned} \tag{3.30}$$

for $h = \Delta x = \Delta y$.

For steady-state problems, the convergence is monitored by observation of the iterative changes

$$E_i = \frac{\|\phi_n - \phi_{n-1}\|_2}{\|\phi_1 - \phi_0\|_2} \tag{3.31}$$

where ϕ_0 is the initial field. As the simulation is approaching steady-state, E_i should approach zero.

Chapter 4

Benchmark Problems

In this chapter, the benchmark cases and methods used will be presented. In order to test and verify the implementation and methods for the WLSQ IBM and Improved IBM a set of suitable benchmark problems is needed. The first benchmark problem is the heat equation (2.3) evaluated on a domain between two concentric circles for steady-state heat conduction.

In the second benchmark problem, the transient behavior of the methods is evaluated. In this case, the heat conduction in the cross-section of an infinitely long cylinder is simulated from a zero initial condition with constant surface temperature given by a Dirichlet BC.

The thermal diffusivity is $\alpha = 1.0$ and the thermal conductivity is $k = 1.0$ throughout.

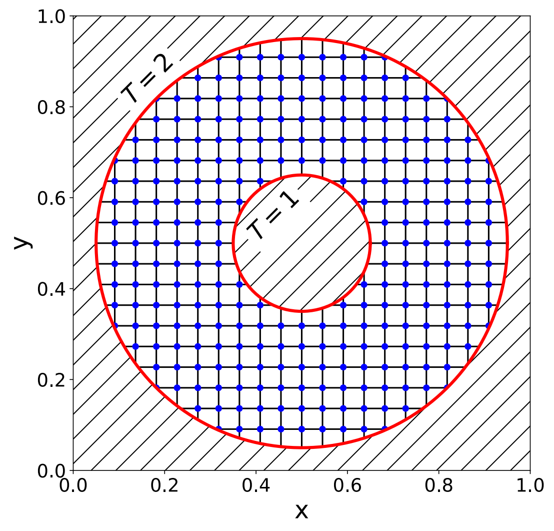
4.1 Steady State Heat Conduction between Concentric Circles

In this benchmark problem, the heat equation (2.3) is evaluated for steady state heat flow between concentric circles in a unit square Cartesian domain. The performance of the WLSQ IBM and Improved IBM is evaluated for combinations of Dirichlet and Neumann boundary conditions, and the full case setup is illustrated in figure 4.1. In this figure, any grid points that lie outside the fluid domain are hidden away. This benchmark problem is carried over from the author's project work [2], and the following section is adapted with modifications from said work. The test case has also been evaluated with other IBM codes such as that of [13], making it a suitable point of comparison.

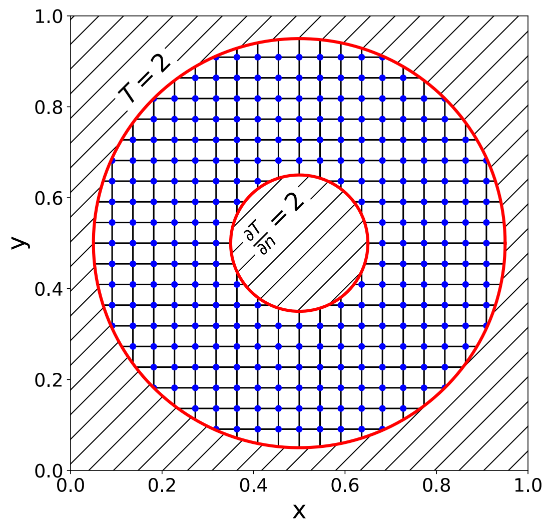
The exact solution for this problem in polar coordinates is in the form

$$T(r) = A \ln r + B\,, \tag{4.1}$$

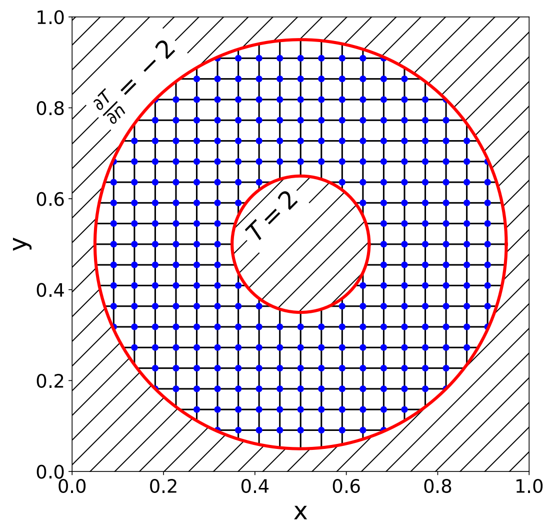
where the constants A and B are determined based on boundary type and values [13].



(a) *Dirichlet – Dirichlet*



(b) *Dirichlet – Neumann*



(c) *Neumann – Dirichlet*

Figure 4.1: Schematic of benchmark geometry for steady state heat conduction between concentric circles. The outline of the immersed bodies is highlighted in red, for which boundary conditions are labeled. The underlying Cartesian grid with nodes in the solution domain is indicated with blue-colored dots.

First, some configuration work for the WLSQ method as outlined in section 3.2.1 is performed along with an accuracy investigation into the effects of changing the order of the polynomial approximation r. This investigation was performed on the steady-state benchmark with the Dirichlet configuration. An overview of the benchmark geometry and boundary configurations is given in figure 4.1, with the Dirichlet configuration specified in 4.1a.

In order to evaluate the differences in a consistent manner the stencil size was set large enough such that at the extremities the weights given are very near zero. Following table 3.1 it is expected that for $r = 1, ..., 4$ the largest stencil size would be for that of $r = 4$. Assuming that lower orders require smaller stencils, a reasonable size was found to be $q = 35$. This way the assumption is that the result is approximately independent of stencil size and the weight function correlates more directly to the accuracy and can therefore be optimized in isolation. Additionally, by keeping the stencil size constant throughout, the optimal weight scaling can be compared directly for each choice of r. The weight function is then optimized by monitoring the condition number of matrix M and making adjustments to the weight scaling k_d to balance accuracy and stability. The results of this investigation are then used as a guideline for the setup of the following configurations. And for verification purposes, the main analysis will be focused on the fourth-order configuration, $r = 3$ as this enables comparison with findings of [13].

Table 4.1: Overview of configurations for the steady state heat conduction between concentric circles benchmark with Dirichlet and exact solution given by $A \approx 0.9066$, $B \approx 2.7259$.

Case No.	N	$r_x + r_y$	t_f	n_t	α	r_{inner}	r_{outer}	T_{inner}	T_{outer}
1	11	│	│	3×10^2	│	│	│	│	│
2	21	│	│	12×10^2	│	│	│	│	│
3	41	0.1	0.15	48×10^2	1.0	0.149	0.449	1	2
4	81	│	│	192×10^2	│	│	│	│	│
5	161	│	│	768×10^2	│	│	│	│	│
6	321	│	│	3072×10^2	│	│	│	│	│
7	641	│	│	12288×10^2	│	│	│	│	│

The configurations with Dirichlet BC for both boundaries are presented in table 4.1 and assigned case numbers $1 - 7$. Each successive configuration performs a grid refinement on the Cartesian grids $N \times N$ starting from case No. 1 with $N = 11$ up to $N = 641$. For the

Dirichlet case, the field is initialized in a linear distribution between the boundaries in order to reduce computational time as compared with a uniformly initialized grid. Based on the results of the above study, a weight scaling of $k_d = 0.01$ is used. The inner and outer circles are given a radius of approximately $r = 0.15$ and $r = 0.45$, and the reason the actual values are offset by 0.001 is to avoid situations where the grid points overlap the boundary perfectly at low resolution as this could lead to large asymmetry in the solution.

Table 4.2: Overview of configurations for the steady-state heat conduction between concentric circles benchmark with Dirichlet BC on the outer circle and Neumann BC on the inner circle, with the analytical solution given by $A = 0.298$, $B \approx 2.2386$.

Case No.	N	$r_x + r_y$	t_f	n_t	α	r_{inner}	r_{outer}	T_{inner}	$\left(\frac{\partial T}{\partial n}\right)_{outer}$
8	11			7×10^2					
9	21			28×10^2					
10	41	0.1	0.35	112×10^2	1.0	0.149	0.449	2	2
11	81			448×10^2					
12	161			1792×10^2					
13	321			7168×10^2					
14	641			28672×10^2					

Table 4.3: Overview of configurations for the steady-state heat conduction between concentric circles benchmark with Neumann BC on the outer circle and Dirichlet BC on the inner circle, with the analytical solution given by $A = -0.898$, $B \approx 0.2904$.

Case No.	N	$r_x + r_y$	t_f	n_t	α	r_{inner}	r_{outer}	$\left(\frac{\partial T}{\partial n}\right)_{inner}$	T_{outer}
15	11			8×10^2					
16	21			32×10^2					
17	41	0.2	0.8	128×10^2	1.0	0.149	0.449	-2	2
18	81			512×10^2					
19	161			2048×10^2					
20	321			8192×10^2					
21	641			32768×10^2					

Following the Dirichlet configurations, the mixed Dirichlet-Neumann and Neumann-Dirichlet (read outer-inner boundary) configurations are given in table 4.2 and table 4.3 respectively. The same grid refinement process outlined above is performed for each boundary configuration. When mixed boundary conditions are considered, the field is uniformly initialized with a constant value given by the Dirichlet boundary.

For each refined grid, the Richardson extrapolation method is performed generating five

Improved IBM solutions in total for each of the three boundary configurations of tables 4.1, 4.2, and 4.3.

The stopping criterion for the simulations is based on eq. (3.31), where the convergence is measured by the change in L_2-norm of the temperature field between time levels. The stopping criterion is based on this iterative change approaching zero.

4.2 Heat Conduction in a Cross-Section of an Infinite Cylinder

In order to further test the methods in view of problems that require transient solutions, a suitable test case for the transient heat equation is needed. In this benchmark problem, the heat equation is evaluated in the cross-section of a cylinder of infinite length with constant surface temperature T_s and zero initial temperature.

According to Jaeger and Carslaw [21], the temperature for this problem in cylindrical coordinates can then be described by radial position $0 \leq r \leq r_s$ and time t alone. In Fourier-Bessel form this is given by

$$T(r,t) = T_s \left[1 - \frac{2}{r_s} \sum_{n=1}^{\infty} e^{-k\alpha_n^2 t} \frac{J_0(r\alpha_n)}{\alpha_n J_1(r_s\alpha_n)} \right] , \tag{4.2}$$

where $\pm r_s\alpha_n,\ n = 1, 2, \ldots,$ are the roots of

$$J_0(r_s\alpha) = 0\,. \tag{4.3}$$

k is the thermal conductivity, and $J_n(x)$ are Bessel functions of the first kind.

The simulations are compared against eq. (4.2), utilizing a finite series of $n = 1, 2, \ldots, 200$. With this limitation, the form is no longer exact, and in particular it is inaccurate in the immediate time after the initial condition when t is small. As such, use has been limited to well-developed states.

Following the same process as before, the boundaries are immersed into a unit square Cartesian domain. The full case setup is illustrated in figure 4.1. In this figure, any grid points that lie outside the fluid domain are hidden away.

Table 4.4: Overview of configurations for the transient benchmark of heat conduction in a cross-section of an infinite cylinder

Case No.	N	$r_x + r_y$	t_i	t_f	n_t	α	r_s	T_s
1	11	|	|	|	350	|	|	|
2	21	|	|	|	1400	|	|	|
3	41	0.02	0.0	0.035	5600	1.0	0.449	2
4	81	|	|	|	22400	|	|	|
5	161	|	|	|	89600	|	|	|
6	321	|	|	|	358400	|	|	|
7	641	|	|	|	1433600	|	|	|

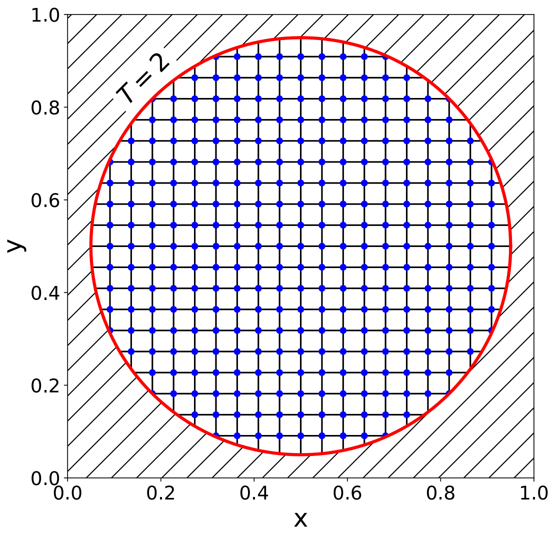


Figure 4.2: Schematic of benchmark geometry for the transient heat conduction in a cross-section of an infinite cylinder. The outline of the immersed body is highlighted in red, with the Dirichlet boundary condition labeled. The underlying Cartesian grid with nodes in the solution domain is indicated with blue-colored dots.

The configurations considered are presented in table 4.4. Similarly to the steady state problem, the simulations are performed on $(N \times N)$ grids as defined in figure 4.2 from $N = 11$ with grid doubling up to $N = 641$. The domain is initialized by constant initial temperature $T_i = 0$, starting at initial time $t_i = 0$.

The simulations are run up to an end time t_f, where the Improved IBM method is performed and results are analyzed.

Chapter 5

Performance of the Numerical Methods

This section will present comments on the performance and complexities of significant components related to the discretization methods. Some parts of the developed code and technical aspects of the hardware and architecture will be featured, for which performance numbers are given in section 6.3.

5.1 Forward Time Centered Space

The FTCS discretization for the 2D heat equation is presented in section 3.1, and a possible implementation is shown in appendix D. Following this function implementation there are two main steps

1. Create a temporary copy of the current field values $\mathcal{O}(N^2)$
2. Calculating and assigning new values to the current field $\mathcal{O}(N^2)$

meaning the function call has a complexity $\mathcal{O}(N^2)$. With one function call occurring per time level, the complexity associated with a full simulation is then $\mathcal{O}(n_t\, N^2)$, where n_t is the number of time levels from the initial condition to the end time.

5.2 Weighted Least Squares Method

5.2.1 Construction

The largest time complexity associated with the WLSQ method is related to constructing and solving the least squares problem of eq. (3.11). However, for problems with stationary boundaries, the result can be stored and needs not be recomputed unless the parameters of the WLSQ method are changed. For stationary boundaries, and since usually $n_t \gg N$, this set-up cost is typically very small with regards to the total time complexity and can therefore be neglected. In such cases, the performance is mostly dictated by the update function described in section 5.2.2.

The construction process is defined in this case as the parts of the algorithm which can be pre-computed and stored. The implementation details are included as source code in appendix D. The implementation can be summarized in the following steps:

1. Construct the numerical stencil
2. Construct the Vandermonde matrix V
3. Compute the weights and assemble the weight matrix W
4. Compute and store the weight matrix M, for which the pseudoinverse of $(WV)^+$ is solved by the SVD algorithm $\mathcal{O}(p^2q)$

The performance of the method is dominated by the SVD algorithm with complexity $\mathcal{O}(p^2q)$ [22], of which $p(r)$ is a function of the order and q is the stencil size. However, it should be noted that there is significant complexity involved in the other steps, so the total cost of the method is also quite dependent on i.e. of the stencil construction.

In this particular implementation, the JacobiSVD class from the Eigen library is employed. Depending on stencil size, stability, and accuracy requirements there may exist other more suitable methods to compute the pseudoinverse.

5.2.2 Update Function

The update function is the most relevant in terms of the performance of the WLSQ method in this thesis, as the boundaries are always stationary. The main update function of the

WLSQ IBM can be implemented as presented in appendix D. Following this method, the major time complexities can be summarized as

1. Updating data point values $\phi = \left\{\phi_1, \phi_2, ..., \phi_q, \right\}$ which has complexity $\mathcal{O}(q)$
2. Calculating ghost point value by the Dirichlet (3.14) BC or the Neumann (3.15) BC of which the vector dot product has the highest complexity of $\mathcal{O}(q)$

where q is the number of points used in the WLSQ problem (3.9). As such, for a total of N_{GP} ghost points, the full complexity of applying boundary conditions is then $\mathcal{O}(N_{GP}\, q)$. Identically as before, when summed over the full simulation of n_t time levels, the complexity totals $\mathcal{O}(n_t\, N_{GP}\, q)$.

Then for the case of stationary boundaries, the overall complexity is therefore determined by $\mathcal{O}(n_t \max(N_{GP}q,\ N^2))$. This result shows that the complexity is dependent on the immersed boundaries. In general, while typically $q \ll N^2$, the number of ghost points varies greatly based on geometry in the range $N_{GP} \lessapprox N^2$. If the boundary has a large surface area relative to domain size and resolution, the condition $N_{GP}q \geq N^2$ may be true, and as a result, the complexity would then be dominated by the WLSQ IBM. However, in practice, the FTCS scheme is normally the dominant complexity because of the resolution needed for such geometries. For the FTCS scheme, it should also be noted that while the complexity is $\mathcal{O}(N^2)$, a significant constant time proportion from floating-point operations is only applicable to the fluid points of the domain, as these computations are skipped for any solid or ghost points. Meaning overall performance is greatly dependent on the number of fluid points. As such, it can be assumed that in most cases the overall complexity remains $\mathcal{O}(n_t N^2)$.

5.3 Improved IBM

The Richardson extrapolation method used for the Improved IBM following eq. (3.27) has the complexity of $\mathcal{O}(N^2)$ where N refers to the number of grid points in each spatial direction of the coarse grid. However, the method requires very few floating point operations and as such the cost of this method is small in comparison to both the FTCS and WLSQ methods. Additionally, it need only be computed for time levels of interest and can be run as a post-processing step.

5.4 Hardware and Architecture

The developed C++ code is built with the MSVC compiler targeting the x64 architecture for Microsoft Windows 11. The hardware which is utilized is presented in table 5.1.

Table 5.1: Overview of system specifications.

System Specifications	
Processor	Intel(R) Core(TM) i7-8700K CPU @ 3.70GHz, 3696 Mhz, 6 Core(s), 12 Logical Processor(s)
GPU	NVIDIA GeForce GTX 1080
OS	Microsoft Windows 11 Home, Version 10.0.22000 Build 22000

Chapter 6

Results and Discussion

In this chapter, the results of the WLSQ IBM and Improved IBM will be presented and discussed for a range of boundary conditions and test cases with comparisons to results for similar methods found in the available literature.

6.1 Steady State Heat Conduction between Concentric Circles

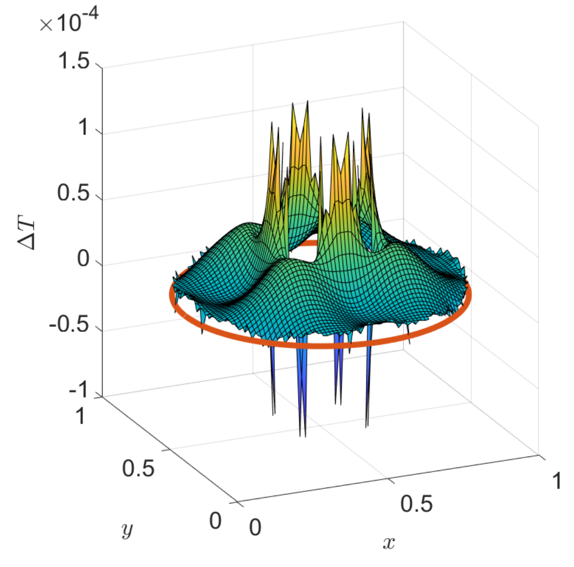
The steady-state benchmark case presented in section 4.1 is utilized for verification of the implementation and particularly for comparison of results to the findings of [13] as well as the author's own project work [2]. The simulations were performed for Dirichlet and mixed Dirichlet-Neumann boundary conditions as given by the configurations presented in tables 4.1, 4.2, and 4.3.

Before setting up the simulation cases, an investigation of the WLSQ method accuracy with varying polynomial approximation r is performed as described in section 4.1. For a stencil size of $q = 35$ it was found that k_d would need to be small to distribute the points evenly on the weighting function. Interestingly, tests indicated that the most accurate results with this implementation are found when the condition number of M is large. It was generally observed that the accuracy would improve as the weight scaling is reduced and that the condition number would increase in response, leading to instability. It was also noticed that the condition number criteria of $< \mathcal{O}(10^6)$ [12] was not sufficient for this implementation.

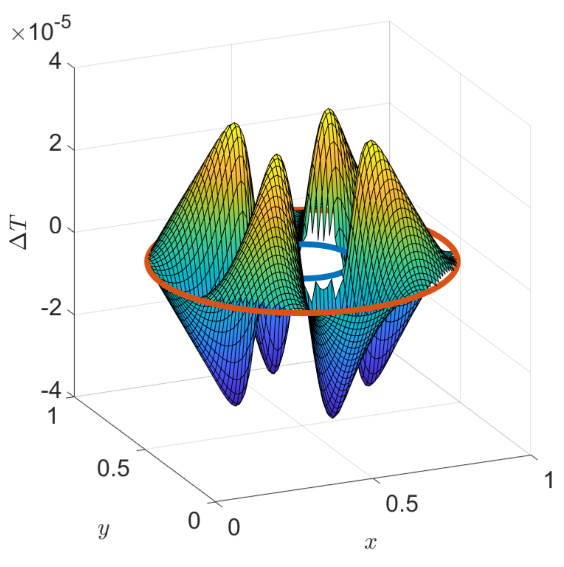
The observations during tests were that for small von Neumann numbers, the accuracy could be improved by allowing condition numbers several orders of magnitude higher. Conversely, when the von Neumann number approaches the stability limit of the FTCS the condition number may need to be much lowered. For this reason, there were no absolute criteria set for the condition number. Nevertheless, it still proved to be a decent indicator of stability when the von Neuman number is kept constant. For this reason, the von Neumann number was kept small in order to achieve high accuracy and keep a decent stability margin.

Table 6.1: k_d and L_2-norm corresponding to the simulations presented in figure 6.1 for varying WLSQ polynomial order r with grid size $N = 161$.

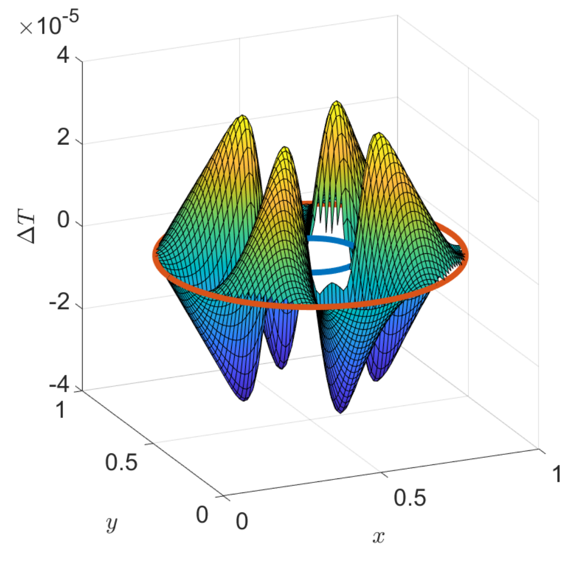
r	k_d	L_2
1	0.00025	1.6949×10^{-5}
2	0.005	1.0287×10^{-5}
3	0.005	1.0370×10^{-5}
4	0.005	1.0381×10^{-5}



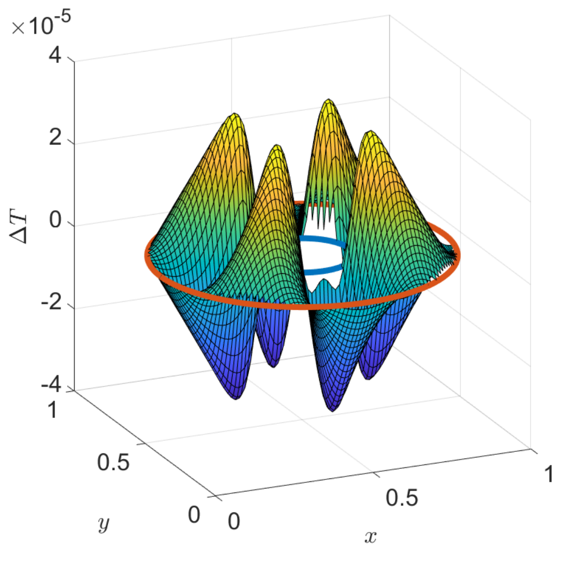
(a) $r = 1$



(b) $r = 2$



(c) $r = 3$



(d) $r = 4$

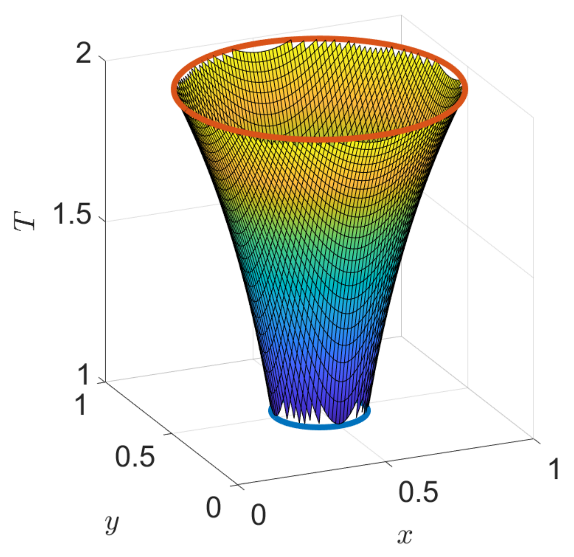
Figure 6.1: Comparison showing error distributions (a), (b), (c), and (d) of the simulation results for the WLSQ IBM with varying polynomial order r at resolution $N = 161$. Simulations are performed on the steady-state benchmark problem with Dirichlet boundary conditions.

Results from the WLSQ accuracy study in the form of error distribution plots with accuracy ranging from 2nd-order to 4th-order at grid resolution $N = 161$ are presented in figure 6.1. Corresponding weight scaling and errors are shown in table 6.1. These results indicate that there are minimal changes in accuracy for the steady state benchmark with Dirichlet boundaries for simulations with $r \geq 2$. This is based on the error distributions taking a very similar form, with small variations at the boundary as well as very consistent L_2-norm.

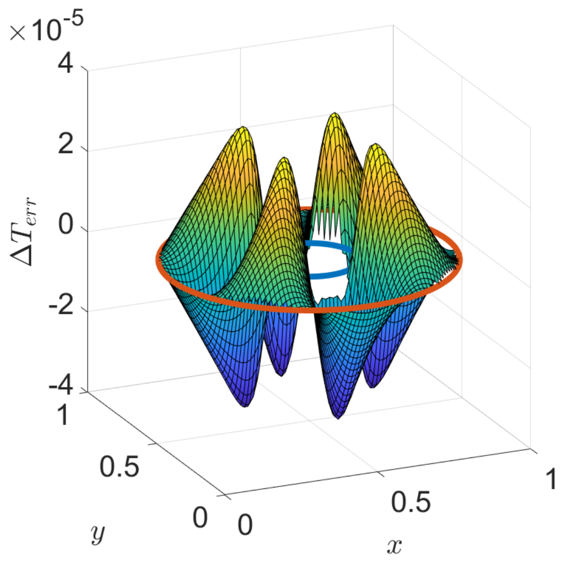
Focusing on the second-order WLSQ method ($r = 1$) some differences are observed. Most notably the error distribution is significantly changed throughout the domain, with large errors observed at the boundaries. This increase in error is also observed in the L_2-norm, which is approximately $\times 1.6$ larger. Based on this result choosing a WLSQ method of third order or greater appears necessary in order to achieve good accuracy.

As previously discussed in chapter 4, the fourth-order boundary reconstruction ($r = 3$) is chosen for the main analysis as it provides a point of comparison with [13]. However, with all results for $r \geq 2$ showing similar performance, it may be possible to employ a lower order and achieve similar results.

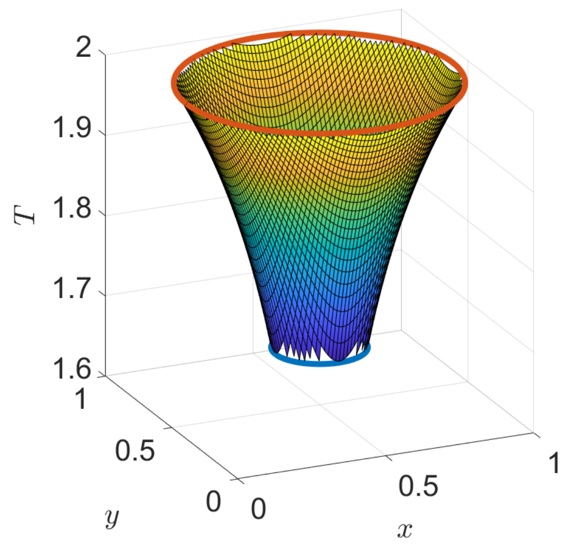
Continuing with the fourth-order WLSQ configuration for the remaining steady-state simulations, the stencil size was kept constant at $q = 35$. The weight scaling and von Neumann numbers were adjusted for each boundary configuration, with a basis in the results above. It is assumed that the observations of the WLSQ accuracy study are also valid for mixed Dirichlet-Neumann boundary conditions, but this was not studied. The simulation results for a single mesh size $N = 161$ for each boundary configuration are presented in figure 6.2, the left-hand side shows temperature T, whereas the right-hand side displays the error in the form of temperature delta compared with the analytical solution. They correspond to cases 5, 12, and 19 from tables 4.1, 4.2 and 4.3 respectively.



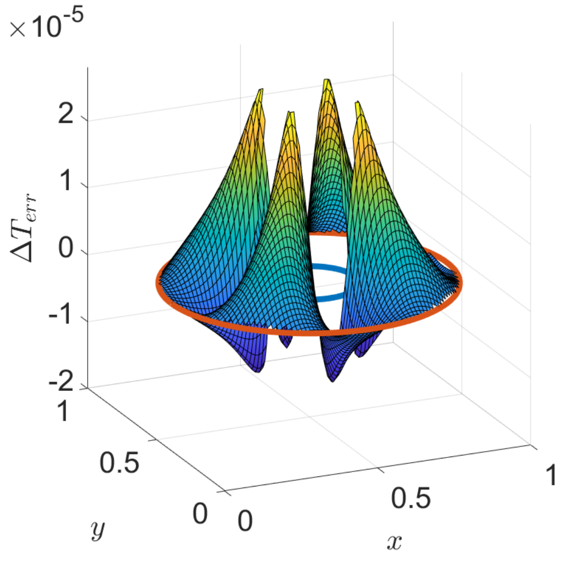
(a) Dirichlet-Dirichlet



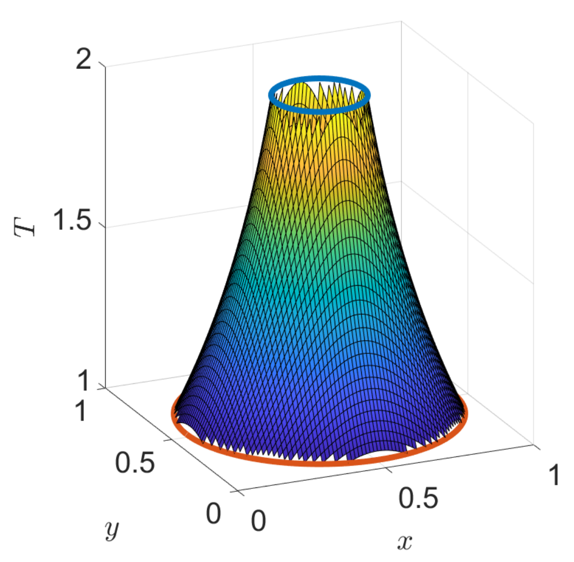
(b) Error



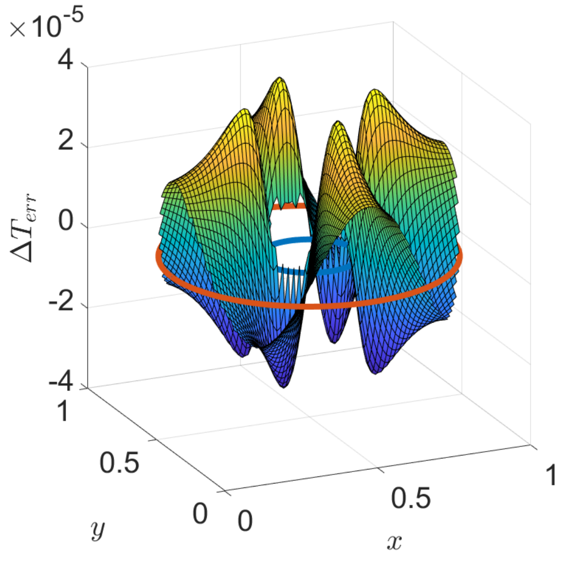
(c) Dirichlet-Neumann



(d) Error



(e) Neumann-Dirichlet



(f) Error

Figure 6.2: Comparison showing simulation results with grid size N=161 for (a), (c), and (e) boundary condition configurations on the steady-state benchmark problem. Including error distribution plots (b), (d), and (f) for each respective configuration.

6.1.1 Dirichlet Boundaries

The simulations with Dirichlet boundaries ran for $n_t = 300$ time levels on the coarsest grid corresponding to case No. 1 of table 4.1. Convergence for each configuration was monitored throughout and is presented in figure 6.3. In this representation, all configurations are drawn at synchronized time levels. All configurations are observed to approach steady-state with iteration errors $|E_i| \leq \mathcal{O}(10^{-14})$ at time levels ranging from $n_t = 250$ to $n_t = 300$.

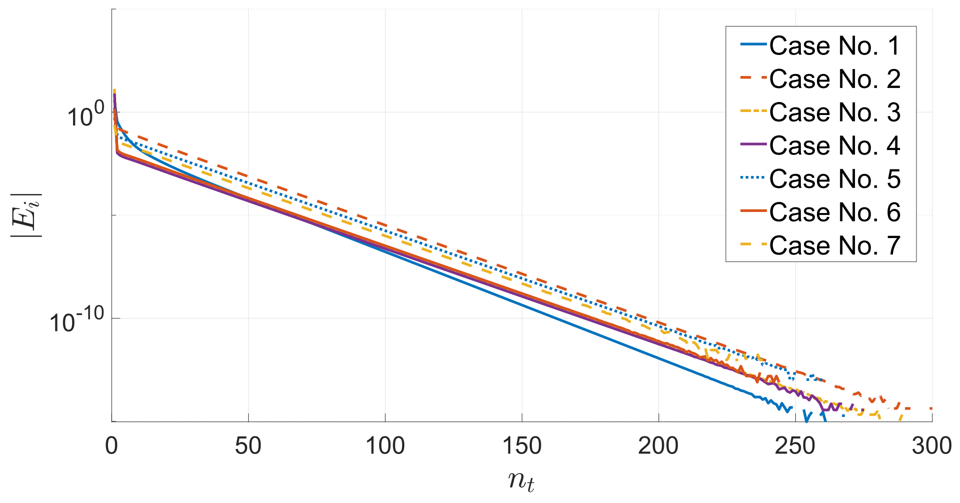


Figure 6.3: Convergence of steady-state simulations with Dirichlet boundaries.

The error distribution for the WLSQ method for the Dirichlet boundary is observed to be effectively fully smooth by inspection of figure 6.2b, with the largest errors located not along the boundary, but rather in the internal region of the domain with magnitudes $\mathcal{O}(10^{-5})$ for grid size $N = 161$. As such the 4th-order WLSQ method indicates significant improvement over the image point method for which it was shown that the same case led to large, sharp errors located directly at the boundaries [2]. From figure A.1 it is also observed that the error distribution for this configuration is very consistent throughout the grid refinement process. This indicates that the boundary is well-resolved and that small changes to the mesh geometry resulting from the immersed boundary flagging process do not further propagate to large changes in the solution. While the current test cases do not provide a view into accuracy on highly complex geometries, this result is promising in this aspect. Overall this is significant in reference to the Improved IBM as the Richardson extrapolation method requires the field values to be smooth everywhere and have monotone behavior with respect to grid refinement.

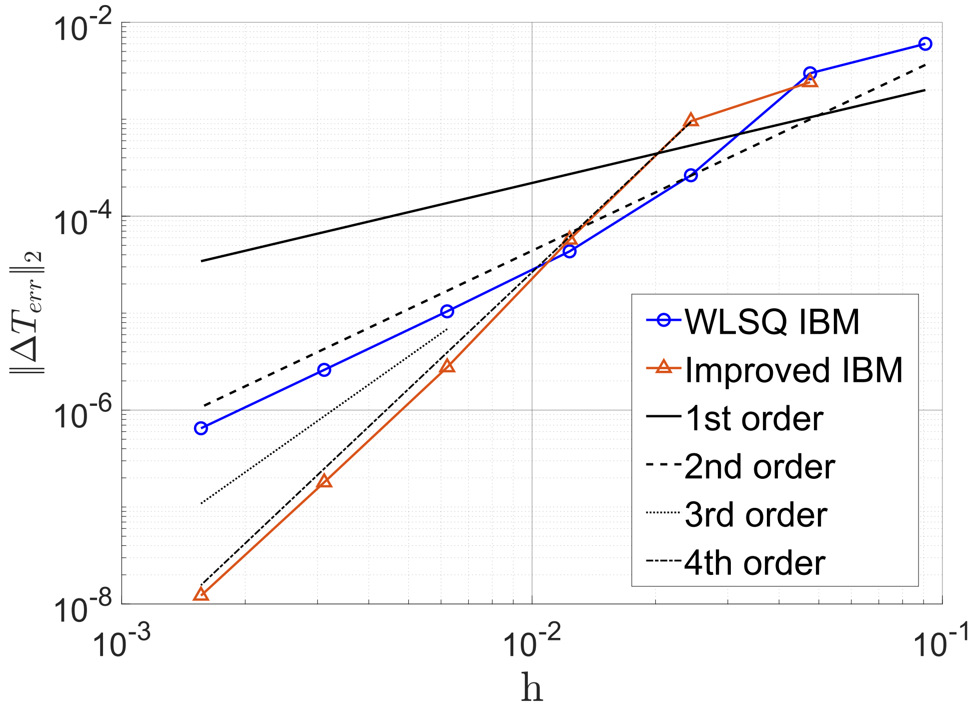


Figure 6.4: Grid convergence rates for the steady-state benchmark problem with Dirichlet boundary conditions including reference lines indicating the order of accuracy.

This is further established in figure 6.4 where clear 2nd-order, monotone convergence of the L_2-norm is observed for all cases starting from case No. 4 of table 4.1 with $N = 81$

and errors reduced to $\leq \mathcal{O}(10^{-6})$ at $N = 641$. As such, figures 6.2a, 6.2b and 6.4 verify that the implementation of the WLSQ IBM for Dirichlet boundaries is achieving the expected order of accuracy, and resolves the benchmark problem with errors which are in line with other reports such as Khalili et al. [13].

On the other end, for coarser meshes, the grid refinement shows both increased error as well as non-monotone convergence. And is a clear indication that with insufficient resolution the accuracy of the WLSQ IBM degrades rapidly. This is to be expected when using the WLSQ IBM method and is reported by several authors as previously described in section 3.2.1. This limitation is particularly evident for high polynomial order as this requires the inclusion of many points in the fluid domain as indicated in table 3.1. Measures such as reducing the order of the method can be useful for coarse grids because this reduces the required size of the numerical stencil, and may in some cases have little effect on accuracy as shown in figure 6.1, and table 6.1.

While several authors also report second-order L_2 convergence for Dirichlet boundaries with IP IBM for similar benchmark problems [2][13], an important distinction is that for the IP method, a clear non-monotone convergence trend is seen throughout the grid refinement process.

6.1.2 Neumann Boundaries

For the Neumann boundary condition, there are two distinct configurations as each of the circular boundaries is tested individually. These are presented in tables 4.2 and 4.3. As with the Dirichlet boundaries, simulation results of the WLSQ method for Neumann boundaries with grid size $N = 161$ are presented in figures 6.2e and 6.2c. Results for all configurations can be found in the form of contour plots in appendix A.

For configurations $8 - 14$ shown in table 4.2 with Dirichlet BC on the outer circle and Neumann BC on the inner (Dirichlet-Neumann) the simulations ran for $n_t = 700$ time levels at $\Delta t = 0.0005$ on the coarsest grid. The convergence results for this configuration are presented in figure 6.5.

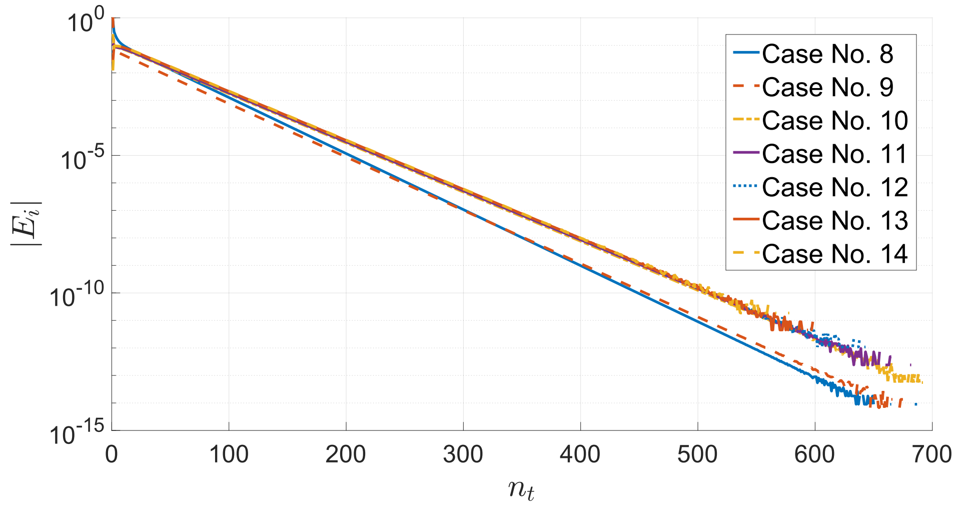


Figure 6.5: Convergence of steady-state simulations with with Dirichlet BC on outer circle and Neumann BC on inner circle.

For configurations $15 - 21$ from table 4.3 with Neumann BC on the outer circle and Dirichlet BC on the inner (Neumann-Dirichlet) the simulations ran for $n_t = 800$ time levels at $\Delta t = 0.001$ on the coarsest grid. The convergence results for this configuration are presented in figure 6.6.

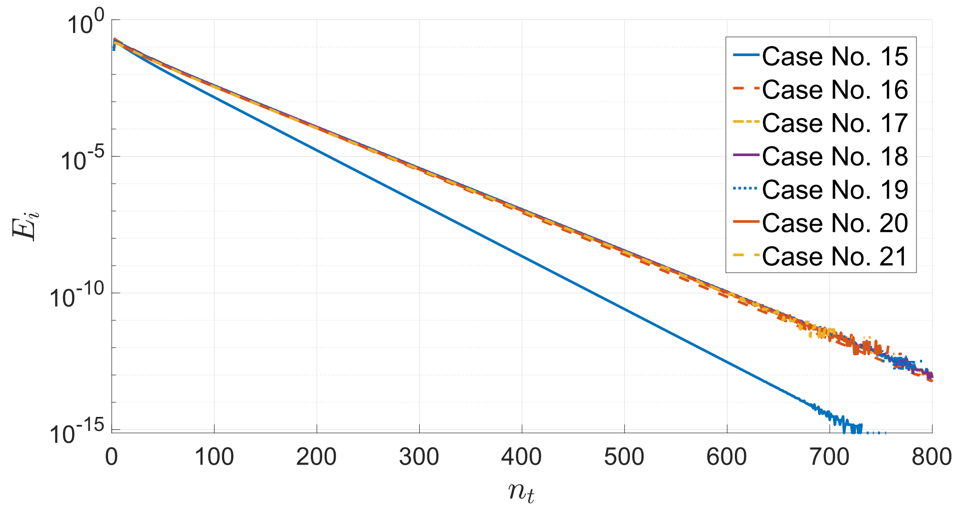


Figure 6.6: Convergence of steady-state simulations with Neumann BC on outer circle and Dirichlet BC on inner circle.

As before, all configurations are drawn at synchronized time levels corresponding to the coarsest grid. All configurations are observed to approach steady-state with $|E_i| \leq \mathcal{O}(10^{-13})$ after $600 - 700$ iterations for Dirichlet-Neumann and $700 - 800$ for Neumann-Dirichlet.

Based on the results presented in these figures, it is observed that the field values along the boundary remain smooth, but a clear distinction for the Neumann condition is that the errors are larger along the boundary than for the Dirichlet case. Particularly large errors are found at the Neumann boundary in the Dirichlet-Neumann configuration. Still, all configurations achieve similar accuracy, within one order of magnitude and with L_2-norms reduced to $\leq \mathcal{O}(10^{-6})$.

Grid refinement results are shown in appendix A, figures A.3 and A.5. For the Neumann-Dirichlet configuration, the refinement results are similar to that of the Dirichlet case, with quite consistent behavior for high-resolution grids. However, the Dirichlet-Neumann configuration shows some inconsistencies. These are mainly in regard to the bias of the error distribution which is shown to wander quite significantly between grids even at high resolution. This may be a problem for the Richardson extrapolation employed by the Improved IBM, as it requires that the behavior is consistent.

The convergence rates of the L_2-norm for the grid refinement study are presented in figure 6.8 for Dirichlet-Neumann configurations, and figure 6.8 for Neumann-Dirichlet, with a clear indication of 2nd-order convergence in both cases for grid sizes $N \geq 81$. This is a key result, as it shows that the WLSQ IBM implementation achieves 2nd-order accuracy in space for all configurations, verifying the FTCS and WLSQ IBM implementation behave similarly to that presented in [13].

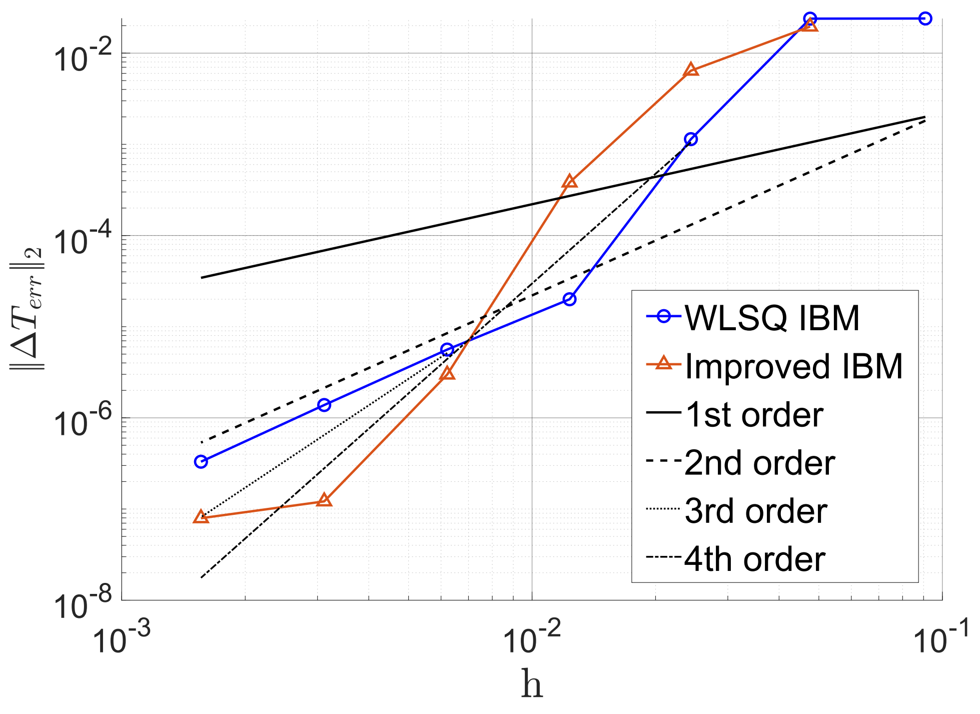


Figure 6.7: Grid convergence for the steady-state benchmark problem with Dirichlet BC on outer circle and Neumann BC on inner circle. The plot includes reference lines indicating the order of accuracy.

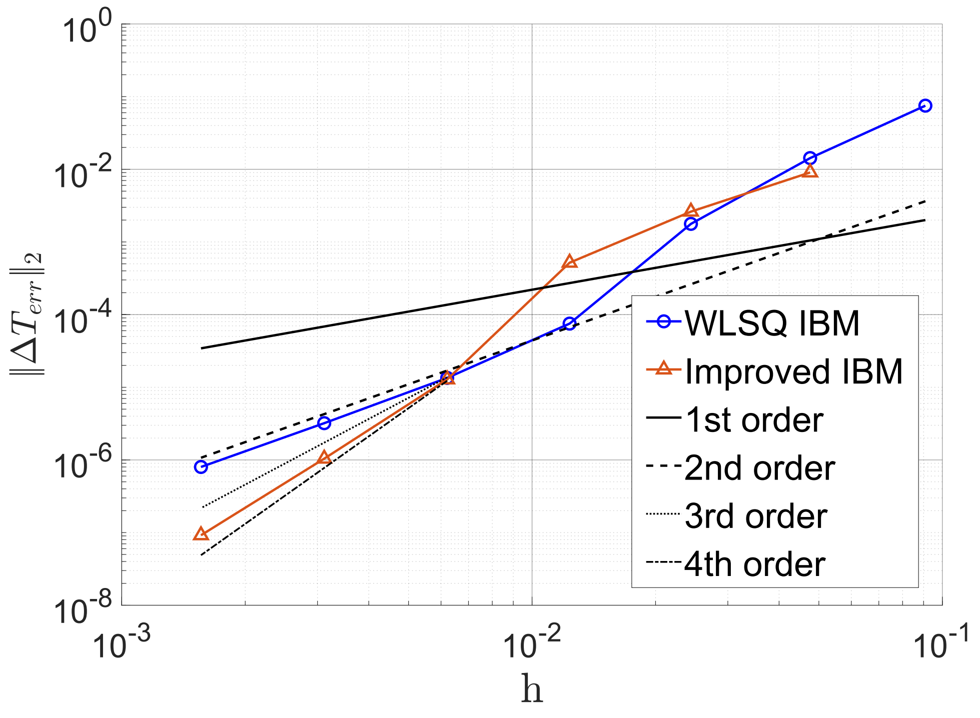


Figure 6.8: Grid convergence for the steady-state benchmark problem with Neumann BC on outer circle and Dirichlet BC on inner circle. The plot includes reference lines indicating the order of accuracy.

It is observed that obtaining 2nd-order convergence with Neumann boundaries seems to require a finer grid spacing than for the Dirichlet boundaries. It should be noted that the weight scaling in this study is adjusted for the accuracy of the fine grids, and so improvement for the coarser grids could be achieved by manual tuning of parameters.

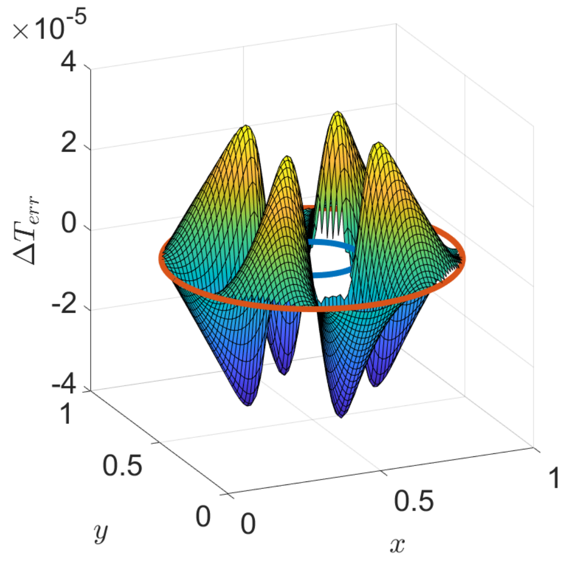
In previous papers, it has been explained and verified that the IP IBM is reduced to 1st-order accuracy with Neumann boundaries [11][13]. This fact was explored further by Khalili et al. [13] where it was compared against a WLSQ IBM with similar observations of improved accuracy as shown here.

6.1.3 Improved IBM

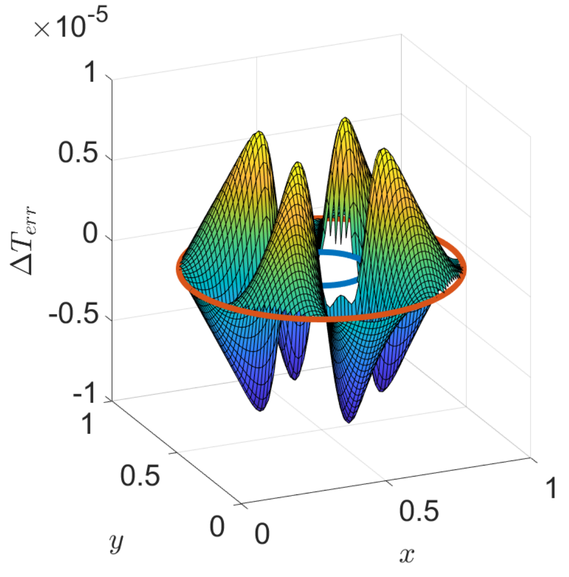
The Improved IBM by the Richardson Extrapolation method is applied as a post-processing step to all configurations from tables 4.1, 4.2, and 4.3, with contour plots of errors given in appendix A. The resulting error distribution for the Dirichlet boundary employing configurations 5 and 6 with coarse grid $N = 161$ and fine grid $N = 321$ is presented in greater detail in figure 6.9. As the figure shows, the error distribution remains quite smooth in the internal field of the fluid domain. Though by comparison to the WLSQ IBM solutions on the coarse and fine grid as shown in subfigures 6.9a and 6.9b, it is clear that the errors at the boundary do not see the same level of improvement. This indicates that the boundaries still exhibit some inconsistency through grid refinement. Still, results show that the errors have been greatly reduced globally when compared to the fine grid solution for grids with high resolution.

For the Dirichlet-only configuration, a clear 4th-order accuracy is shown in figure 6.4 and therefore verifies that the extrapolation is functioning as the theory suggests. This is an important finding, as this was not observed with basis in the image point method, and has not been previously shown for the WLSQ IBM method to the knowledge of the author.

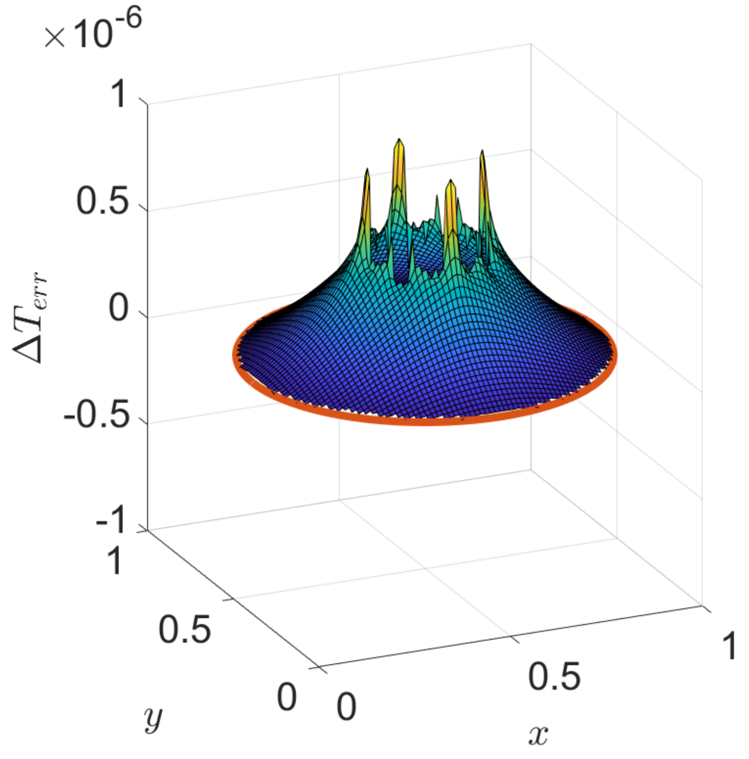
The results from the Richardson extrapolations are however not unilateral. Because for mixed boundaries, while the Neumann-Dirichlet configuration shows a consistent increase to near 4th-order accuracy, the Dirichlet-Neumann configuration shows increased accuracy but with severely inconsistent convergence. It is suspected that this is a consequence of the wandering seen in the error distributions as discussed in section 6.1.2 and shown in figure A.4.



(a) WLSQ IBM case No. 5 ($N = 161$) error.



(b) WLSQ IBM case No. 6 ($N = 321$) error.



(c) Improved IBM [5-6] error

Figure 6.9: Surface plot of Improved IBM error for grid size $N = 161$ on the steady-state benchmark problem with Dirichlet boundaries as compared to the baseline WLSQ IBM errors for grid sizes (a) $N = 161$ and (b) $N = 321$ employed in the Richardson Extrapolation.

An important observation is that for all configurations of the steady-state benchmark, an increase in accuracy is only observed at high resolution. As such the overall increase of accuracy through grid refinement is somewhat limited, and care is needed in order to avoid situations where the Improved IBM results in a less accurate solution. It is plausible that with the WLSQ IBM configured with emphasis on coarser grids, similar results could be achieved at a lower resolution.

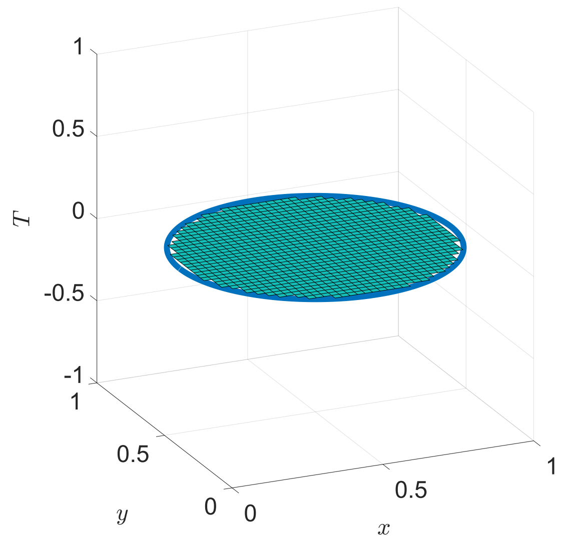
6.2 Heat Conduction in a Cross-Section of an Infinite Cylinder

In order to investigate how the WLSQ and Improved IBM extend to time-dependent use cases, a transient benchmark problem based on heat conduction in a cross-section of an infinite cylinder presented in section 4.2 is employed. The simulations are configured as listed in table 4.4. Due to the similarity of geometry and boundary conditions, it is assumed that the parameters used for the WLSQ IBM as it was set up in section 6.1 are valid. The simulations are therefore performed with $r = 3$, $q = 35$, and weight scaling $k_d = 0.005$.

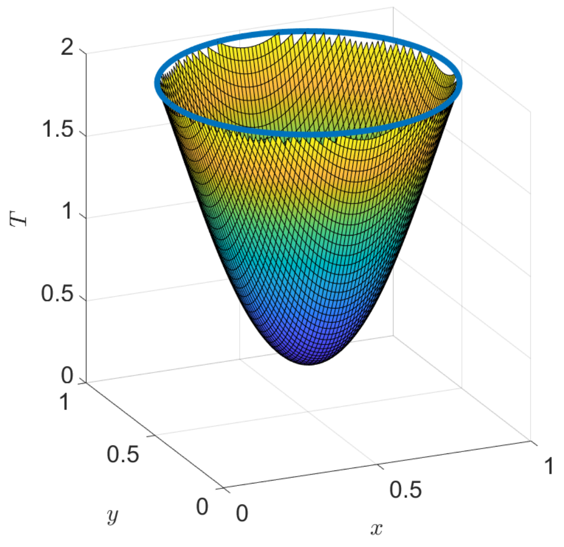
In order to reduce time-dependant errors of $\mathcal{O}(\Delta t)$ from the FTCS scheme, a small von Neumann number is employed. As described in section 4.2 the simulations are initialized with zero temperature at initial time $t_i = 0$ and are run up to time $t_f = 0.035$ where they are subsequently analyzed.

The Improved IBM is then performed as a post-processing step following the method outlined in section 3.3. The transient benchmark problem employed does not cover as extensive of analysis on the methods as the steady-state method, mainly because the geometry is simpler and performed only for the Dirichlet boundary condition. Additionally, based on the steady-state results, it can be expected that a Neumann condition may lead to different behavior.

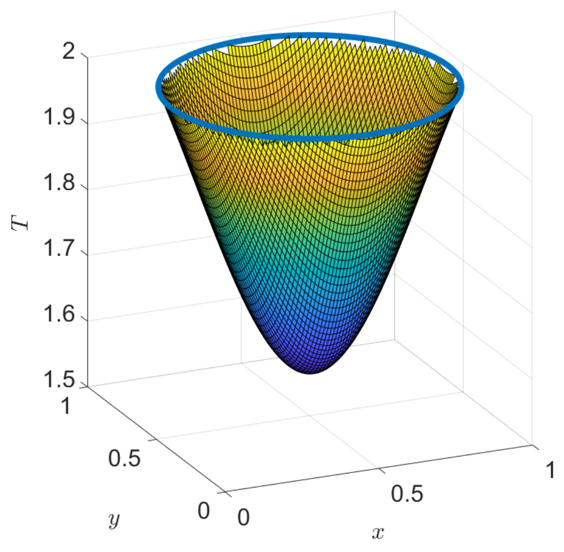
Results showing the time evolution of a simulation with grid size $N = 161$ are presented in figure 6.10. As time evolution shows, the case is well developed at its stopping criterion, and subsequently no issues regarding the accuracy of the analytical solution of eq. (4.2) is expected.



(a) $t = 0.0$



(b) $t = 0.01$



(c) $t = 0.035$

Figure 6.10: Surface plot of WLSQ IBM case No. 5 with grid size $N = 161$ on the unsteady benchmark problem for three points in time, each individually shown in (a), (b), and (c).

A grid convergence study is performed on the cases presented in table 4.4, with convergence rates presented in figure 6.11. It is found that the WLSQ IBM performs to expectations, achieving 2nd-order accuracy with errors in line with previous results. In addition, full improvement to the expected 4th-order accuracy is observed for the Improved IBM. The WLSQ IBM achieves 2nd-order accuracy at a grid size of $N = 41$ and above, which also coincides with the Improved IBM achieving 4th-order accuracy. In this configuration, the Improved IBM shows an improvement in accuracy for any grid spacing, however, based on results observed in section 6.1 this should not always be expected.

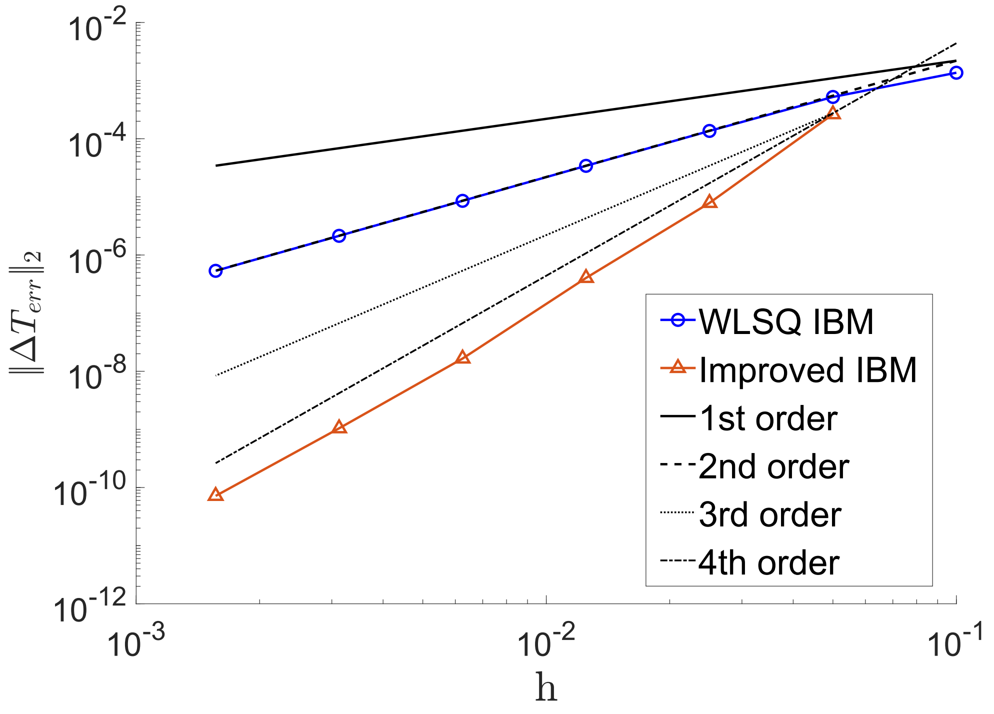


Figure 6.11: Grid convergence rates for the unsteady benchmark problem with Dirichlet boundary condition including reference lines indicating the order of accuracy.

6.3 Performance

Table 6.2: Simulation performance benchmarked on the target hardware.

Configuration	Boundary Conditions	Time
Steady state	Dirichlet	1h 20m 15s
	Dirichlet-Neumann	3h 11m 17s
	Neumann-Dirichlet	3h 32m 41s
Unsteady	Dirichlet	1h 38m 59s

The simulations were performed on a standard personal home computer with specifications as listed in table 5.1. The steady-state benchmark with Neumann boundaries took the longest at roughly three hours, while for Dirichlet boundaries and the transient benchmark, the simulations took roughly one and a half hours. The simulation timings are provided in table 6.2.

Chapter 7

Conclusions

In this thesis, an improved sharp interface immersed boundary method utilizing Richardson extrapolation to achieve higher order accuracy has been developed and verified for the two-dimensional heat conduction equation discretized by the FTCS scheme. The immersed boundaries are resolved by employing a high-order sharp interface ghost point approach utilizing a Weighted Least Squares method. The methods have been assessed on a steady state benchmark problem of heat conduction between concentric circles with Dirichlet and Neumann boundary conditions. Additionally, the methods have been investigated on a time-dependent problem modeled for heat conduction in a cross section of an infinite cylinder with constant surface temperature.

The WLSQ IBM implementation is found to achieve the theorized 2nd-order accuracy of the FTCS formulation for every boundary configuration and benchmark investigated. For Dirichlet BC, the higher order WLSQ method is shown to simulate the benchmark problems correctly with boundaries resolved to a high degree of accuracy including smooth distributions and monotone convergence. However, for the steady state benchmark, the error characteristics are found to be two-fold as in this case when employing Neumann BC on the inner circle a reduction of accuracy was observed at the boundary along with non-monotonic convergence. This inconsistent convergence has been investigated and it is shown that the implications for the improved IBM method can be profound. Additionally, because of the large stencil size required for the fourth order WLSQ method, good accuracy could not be guaranteed for coarse grids. Furthermore, it was found when optimizing parameters for best accuracy, that the stability of the method would be severely impacted and that a low von Neumann number was required. This has a serious impact on the simulation time.

Further, the Improved IBM implementation achieved the theorized 4th-order accuracy in space for both steady and time dependant problems with Dirichlet BC's. This is a key result relating to the improvement seen from the WLSQ method, as the IP method employed in the author's project work [2] showed no such improvement. With Neumann BC applied on the outer circle on the steady-state problem an improvement was found to near 4th-order, however when the inner circle was defined by a Neumann condition the convergence was irregular. Results indicate that the non-monotonic convergence observed for the Improved IBM may be from changes in the boundary errors resulting from grid refinement.

Correctly configuring the WLSQ IBM (and consequently the Improved IBM) to achieve these results has shown to be a labor-intensive process of iterative parameter optimization. For this reason, some skepticism of the consistency of the results is maintained. With parameters slightly out of tune, the results could change significantly. And so it is recommended to view these results not as showing performance which can be expected for any problem without modification, but as a snapshot of the capabilities of the currently quite primitive implementation.

Chapter 8

Future Work

The fundamental motivation for the project is the application of the method to problems involving the Navier-Stokes on flows with transient phenomena, including complex and moving boundaries. From the current state of development and verification, there is a significant road of challenges ahead. The WLSQ IBM has previously been utilized for some problems in this area [13][12], but the effectiveness of the Improved IBM is uncertain for moving boundaries and for more complex equations.

Furthermore, while the WLSQ is shown to accurately resolve Dirichlet boundaries, results indicate that further investigation into improving the accuracy and consistency for Neumann BC's is necessary in order to achieve the expected fourth order accuracy for the Improved IBM in this case.

A rather straightforward addition to the method herein would be to extend the Improved IBM to the fourth order on the fine grid solution following the work of Roache and Knupp [16] and Richards [20]. This approach was considered during this thesis and preliminary results increased order of accuracy also on the fine grid points bounded by the coarse grid of the fluid domain. However, extending the extrapolated fine grid solution to the full limits of the boundary would require the inclusion of ghost point values. This could lead to loss of accuracy at the boundaries, though with the assumption of a smooth extension of the field values into the solid, it seems reasonable that this would be negligible. Results regarding this method were excluded due to time limitations.

References

[1] E. Khalili, “Fluid-structure interaction and immersed boundary method for the compressible Navier–Stokes equations using high order methods,” PhD thesis, NTNU, 2017.

[2] T. T. Lindtveit, “Improved immersed boundary method for biofluid dynamics,” Project Work, NTNU, 2022.

[3] R. Mittal and G. Iaccarino, “Immersed boundary methods,” *Annual Review of Fluid Mechanics*, vol. 37, no. 1, pp. 239–261, 2005.

[4] D. Goldstein, R. Handler, and L. Sirovich, “Modeling a no-slip flow boundary with an external force field,” *Journal of Computational Physics*, vol. 105, no. 2, pp. 354–366, 1993.

[5] M. E. Khalili, M. Larsson, and B. Müller, “Immersed boundary method for viscous compressible flows around moving bodies,” *Computers & Fluids*, vol. 170, pp. 77–92, 2018.

[6] C. S. Peskin, “Flow patterns around heart valves: a numerical method,” *Journal of Computational Physics*, vol. 10, no. 2, pp. 252–271, 1972.

[7] J. Mohd-Yusof, “Combined immersed-boundary/b-spline methods for simulations of flow in complex geometries,” *Center for Turbulence Research Annual Research Briefs*, vol. 161, no. 1, pp. 317–327, 1997.

[8] Y.-H. Tseng and J. H. Ferziger, “A ghost-cell immersed boundary method for flow in complex geometry,” *Journal of Computational Physics*, vol. 192, no. 2, pp. 593–623, 2003.

[9] S. Majumdar, G. Iaccarino, P. Durbin *et al.*, “RANS solvers with adaptive structured boundary non-conforming grids,” *Annual Research Briefs*, vol. 1, 2001.

[10] R. Ghias, R. Mittal, and T. Lund, “A non-body conformal grid method for simulation of compressible flows with complex immersed boundaries,” in *42nd AIAA Aerospace Sciences Meeting and Exhibit*, 2004, p. 80.

[11] R. Mittal, H. Dong, M. Bozkurttas, F. Najjar, A. Vargas, and A. von Loebbecke, “A versatile sharp interface immersed boundary method for incompressible flows with complex boundaries,” *Journal of Computational Physics*, vol. 227, no. 10, pp. 4825–4852, 2008.

[12] J. H. Seo and R. Mittal, “A high-order immersed boundary method for acoustic wave scattering and low-mach number flow-induced sound in complex geometries,” *Journal of Computational Physics*, vol. 230, no. 4, pp. 1000–1019, 2011.

[13] M. E. Khalili, M. Larsson, and B. Müller, “High-order ghost-point immersed boundary method for viscous compressible flows based on summation-by-parts operators,” *International Journal for Numerical Methods in Fluids*, vol. 89, no. 7, pp. 256–282, 2019.

[14] L. F. Richardson, “The approximate arithmetical solution by finite differences of physical problems involving differential equations, with an application to the stresses

in a masonry dam," *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, vol. 210, no. 459-470, pp. 307–357, 1911.

[15] P. Roache, *Verification and Validation in Computational Science and Engineering.* Hermosa Publishers, 1998.

[16] P. J. Roache and P. M. Knupp, "Completed richardson extrapolation," *Communications in Numerical Methods in Engineering*, vol. 9, no. 5, pp. 365–374, 1993.

[17] S. Patankar, *Numerical Heat Transfer and Fluid Flow*, ser. Series in Computational Methods in Mechanics and Thermal Sciences. Taylor & Francis, 1980.

[18] P. Roache, *Computational Fluid Dynamics*, ser. Lecture Series. Hermosa Publishers, 1976.

[19] C. Brehm, C. Hader, and H. Fasel, "A locally stabilized immersed boundary method for the compressible navier–stokes equations," *Journal of Computational Physics*, vol. 295, pp. 475–504, 2015.

[20] S. A. Richards, "Completed richardson extrapolation in space and time," *Communications in Numerical Methods in Engineering*, vol. 13, pp. 573–582, 1997.

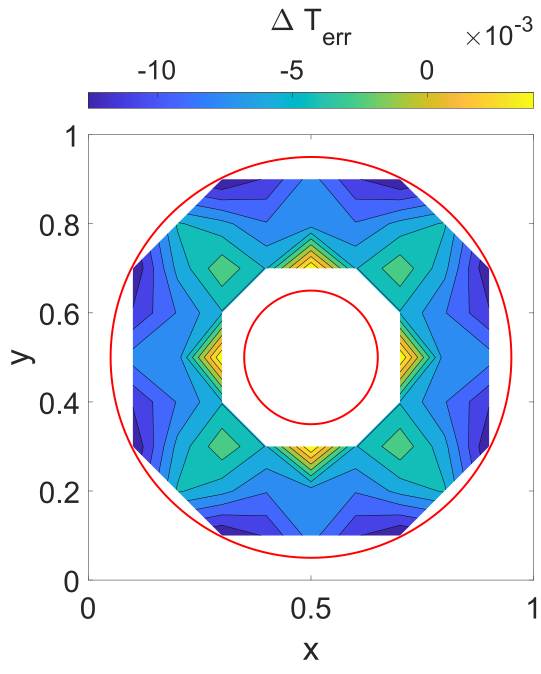
[21] J. C. Jaeger and H. S. Carslaw, *Conduction of heat in solids.* Clarendon Press Oxford, United Kingdom, 1959.

[22] "Eigen::JacobiSVD description," https://eigen.tuxfamily.org/dox/classEigen_1_1JacobiSVD.html, accessed: 09-06-2023.

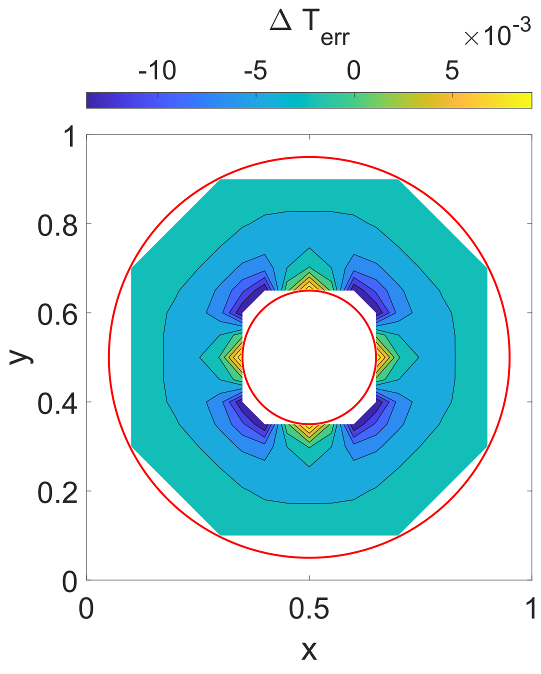
Appendix

A Contour Plots - Steady State Heat Conduction between Concentric Circles

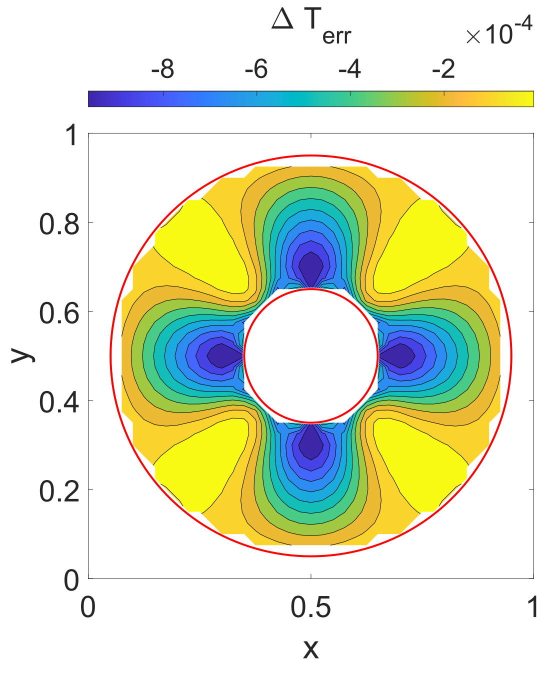
A.1 Dirichlet Boundaries



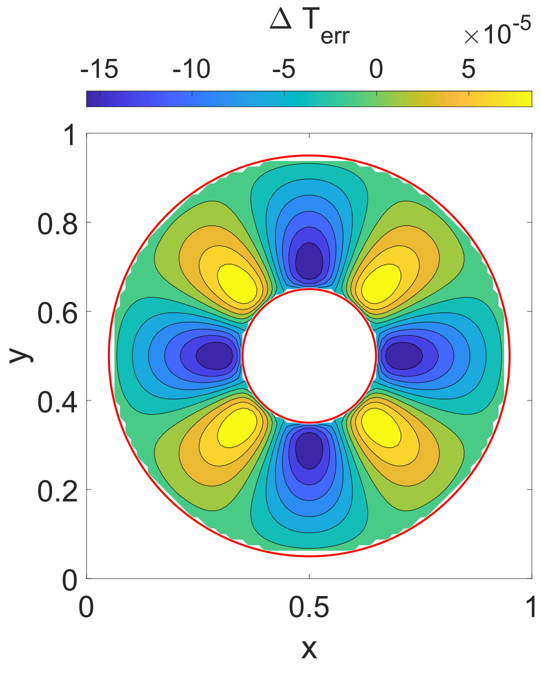
(a) Case No. 1



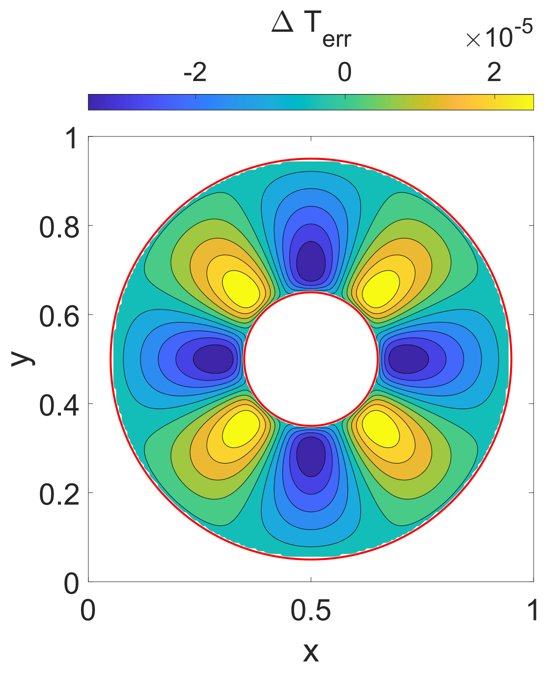
(b) Case No. 2



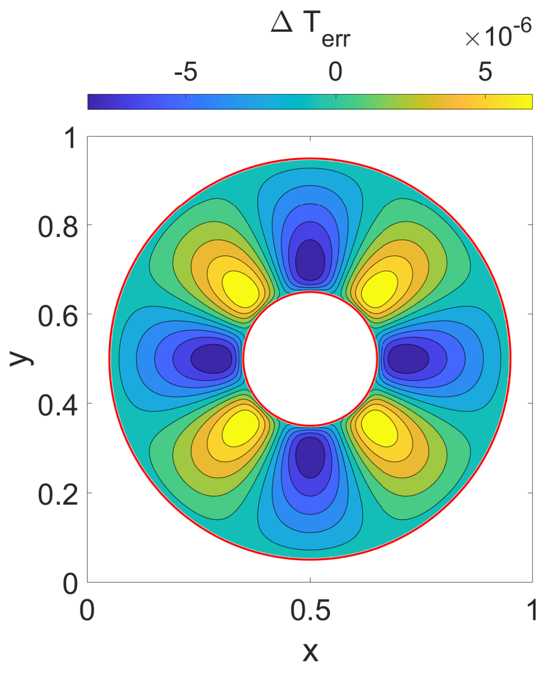
(c) Case No. 3



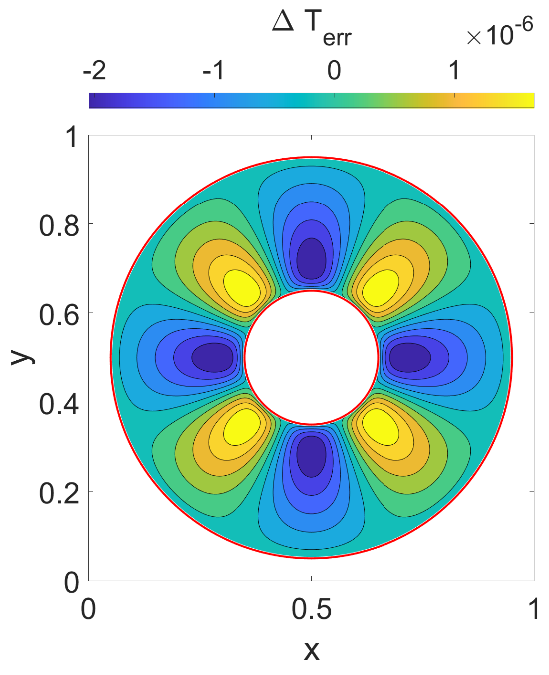
(d) Case No. 4



(e) Case No. 5

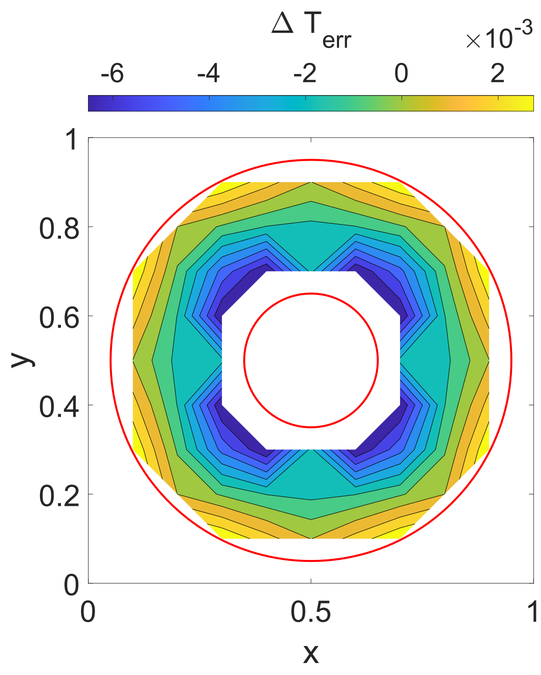


(f) Case No. 6

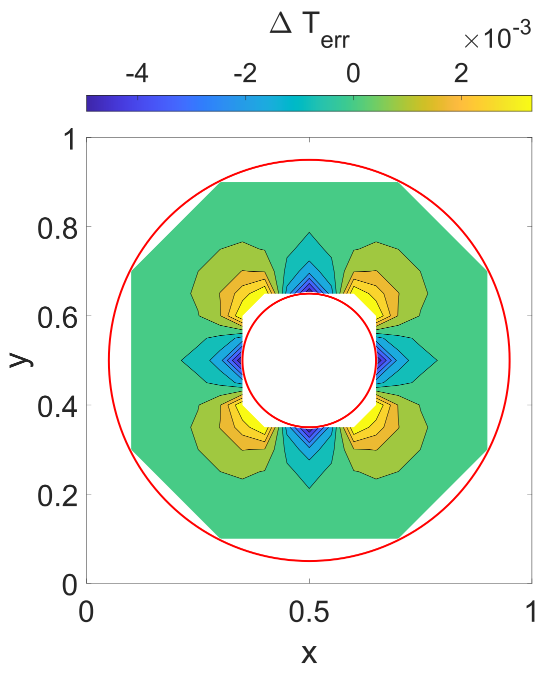


(g) Case No. 7

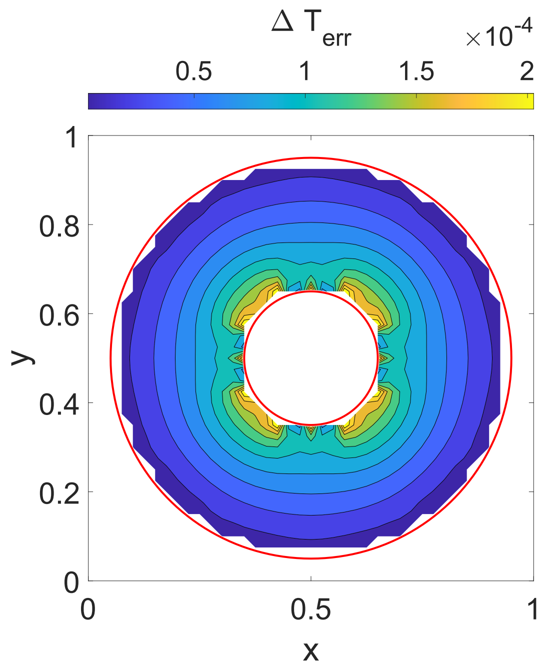
Figure A.1: Error contours for WLSQ IBM cases with steady state heat conduction between concentric circles for Dirichlet BC on both boundaries. Figures (a)-(g) show results corresponding to each case in table 4.1.



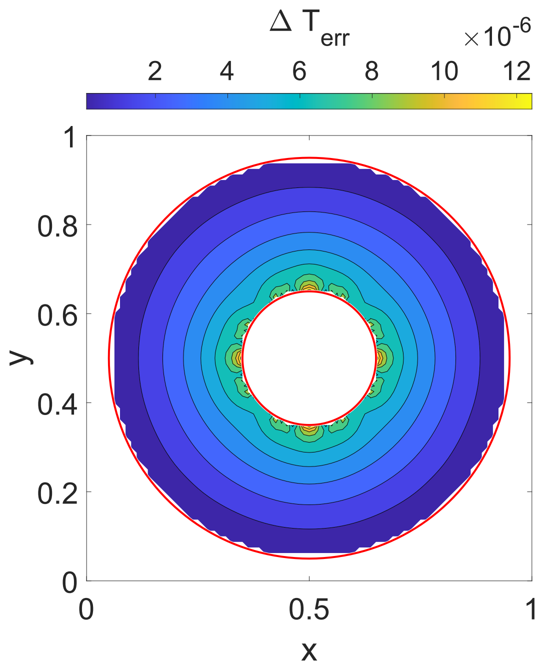
(a) Improved IBM on Case No. [1-2]



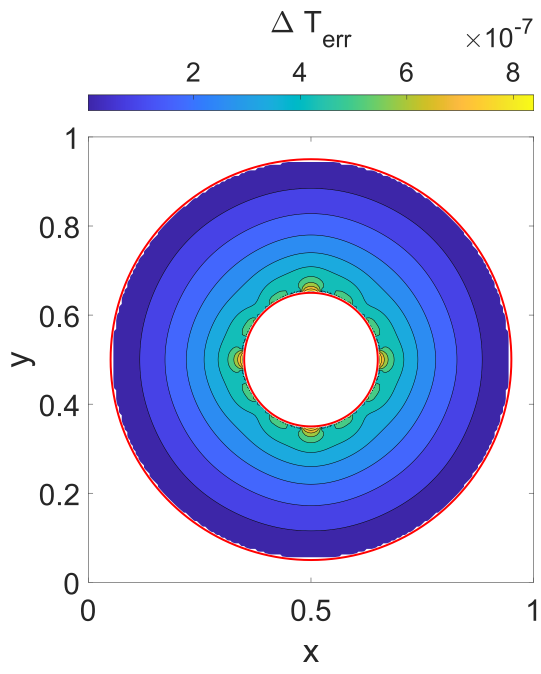
(b) Improved IBM on Case No. [2-3]



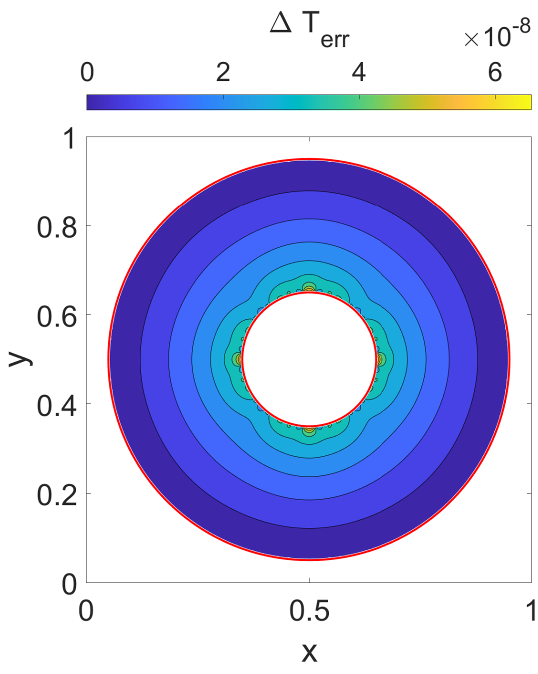
(c) Improved IBM on Case No. [3-4]



(d) Improved IBM on Case No. [4-5]



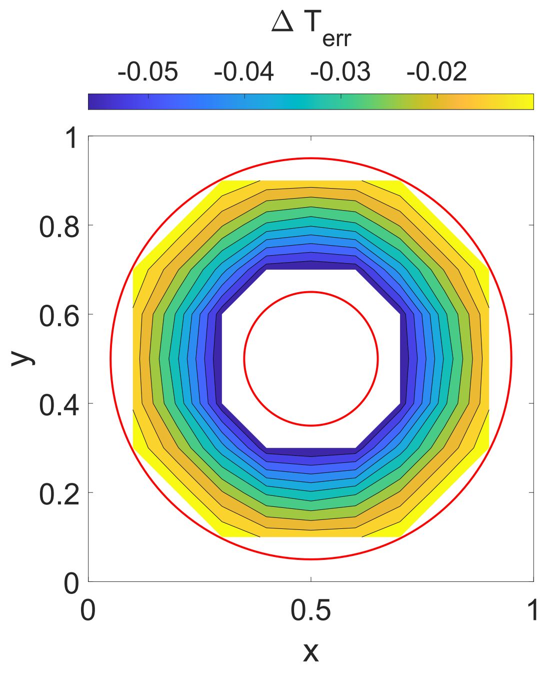
(e) Improved IBM on Case No. [5-6]



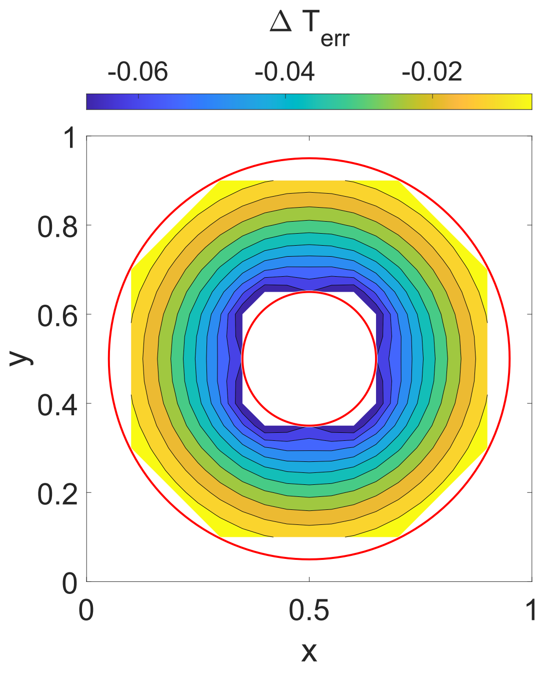
(f) Improved IBM on Case No. [6-7]

Figure A.2: Error contours for Improved IBM with steady state heat conduction between concentric circles for Dirichlet BC on both boundaries. Figures (a)-(f) show results corresponding to each pair of cases in table 4.1.

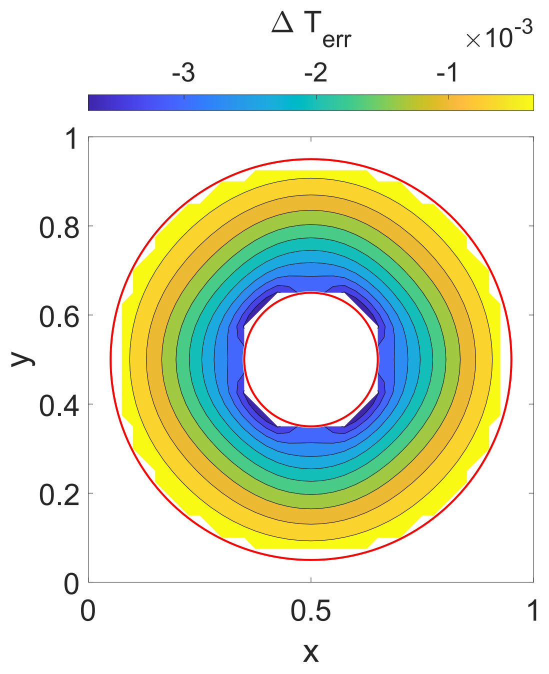
A.2 Dirichlet - Neumann Boundaries



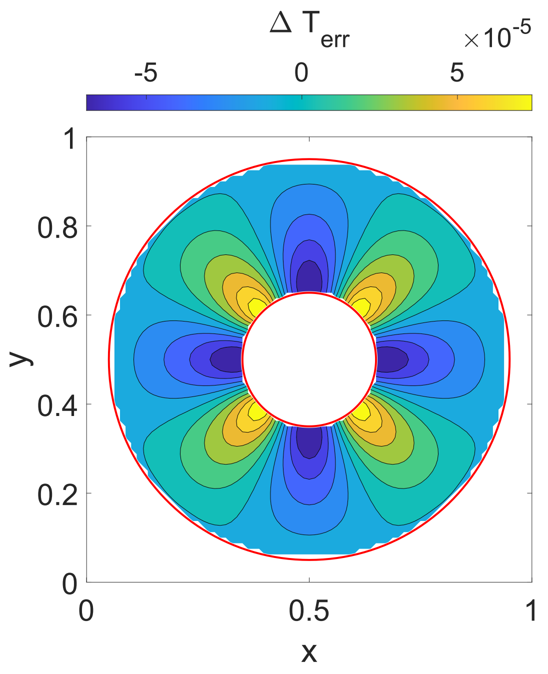
(a) Case No. 8



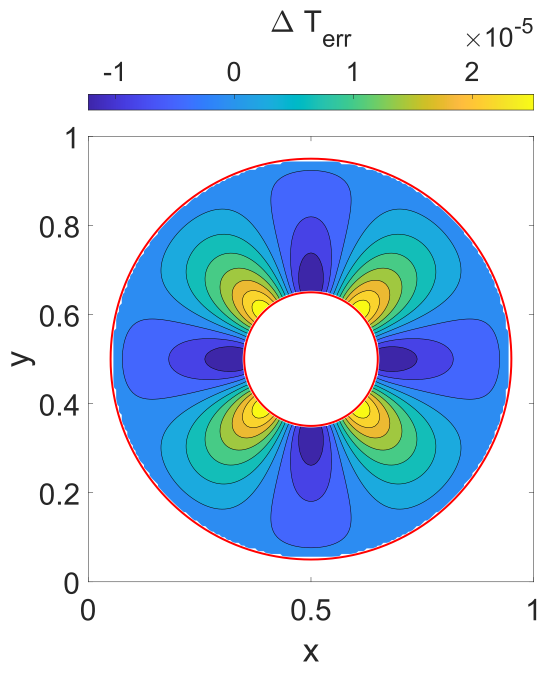
(b) Case No. 9



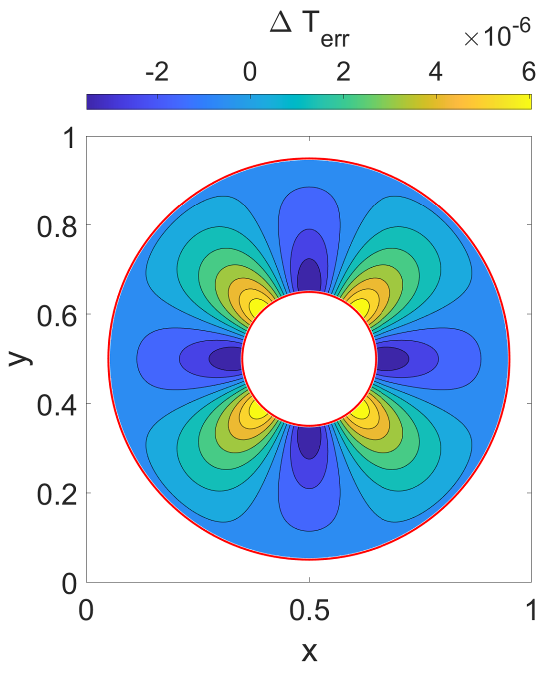
(c) Case No. 10



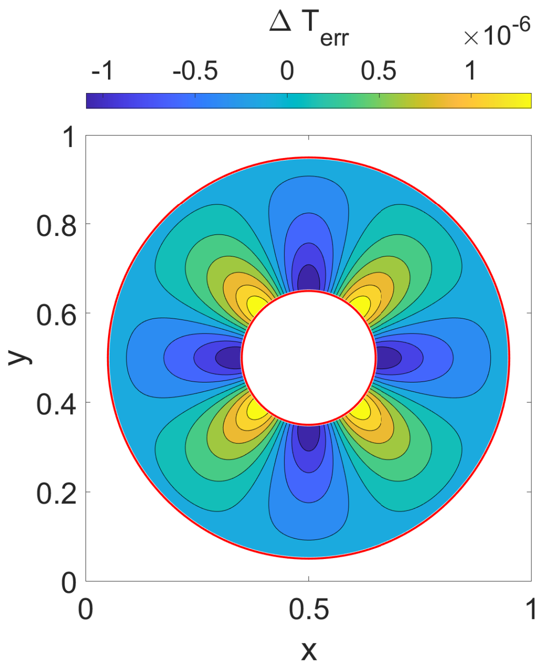
(d) Case No. 11



(e) Case No. 12

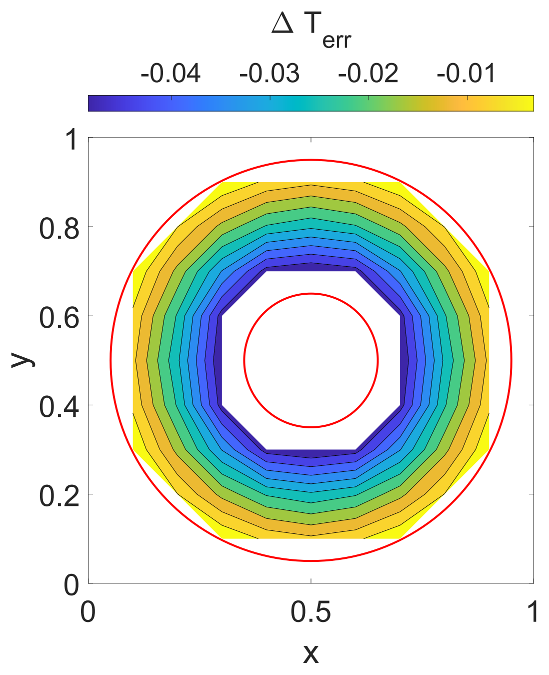


(f) Case No. 13

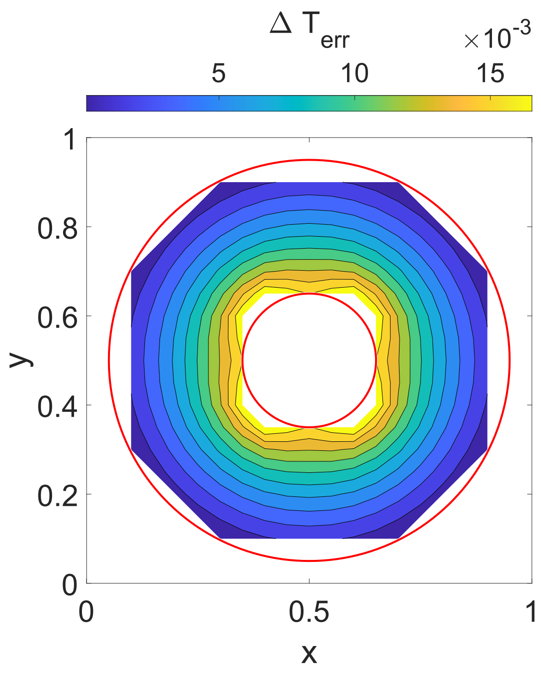


(g) Case No. 14

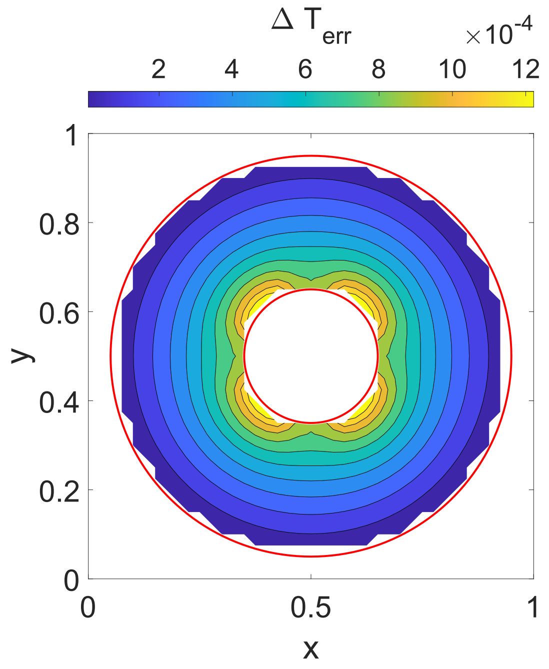
Figure A.3: Error contours for WLSQ IBM cases with steady state heat conduction between concentric circles for Dirichlet BC on the outer circle and Neumann BC on the inner circle. Figures (a)-(g) show results corresponding to each case in table 4.2.



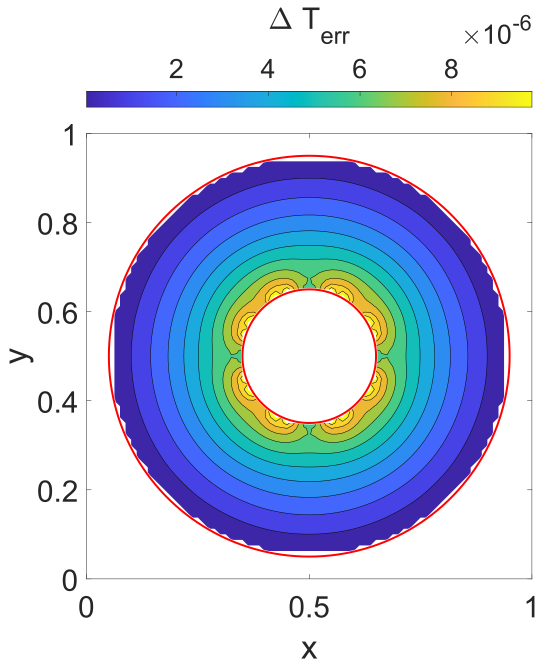
(a) Improved IBM on Case No. [8-9]



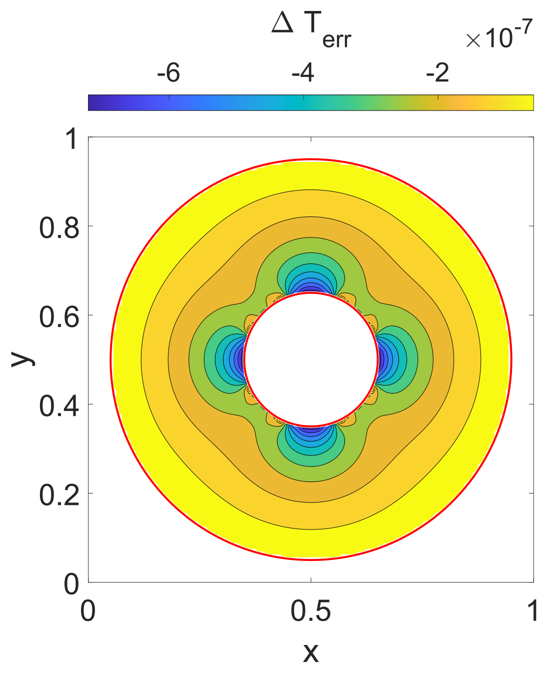
(b) Improved IBM on Case No. [9-10]



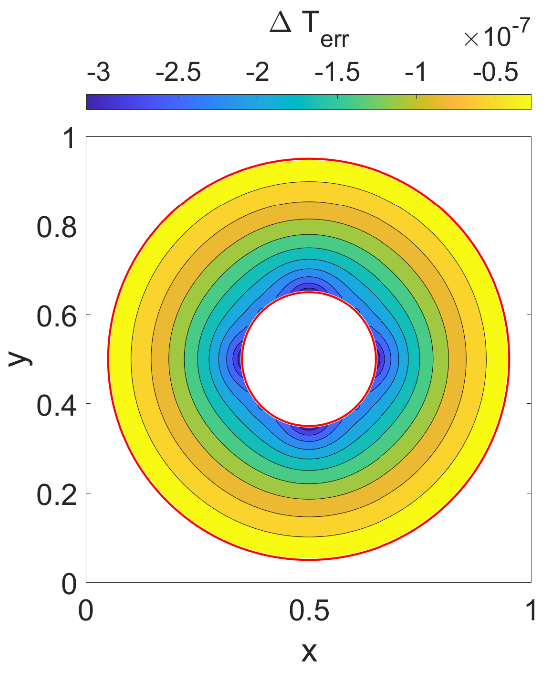
(c) Improved IBM on Case No. [10-11]



(d) Improved IBM on case No. [11-12]



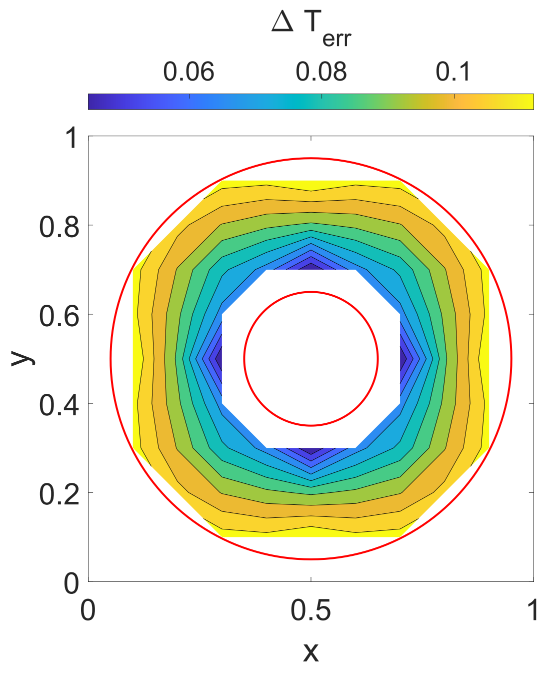
(e) Improved IBM on Case No. [12-13]



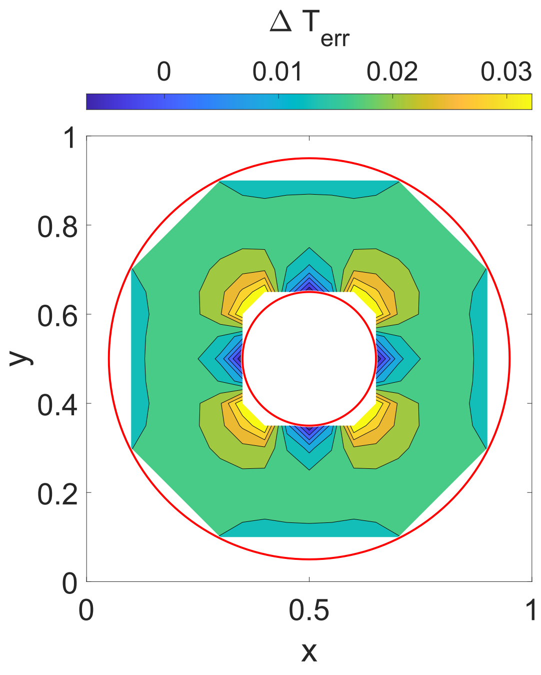
(f) Improved IBM on Case No. [13-14]

Figure A.4: Error contours for Improved IBM with steady state heat conduction between concentric circles for Dirichlet BC on the outer circle and Neumann BC on the inner circle. Figures (a)-(f) show results corresponding to each pair of cases in table 4.2.

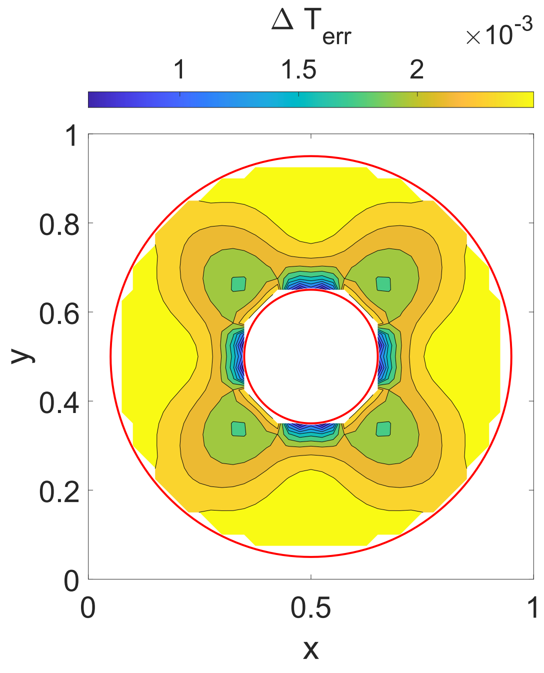
A.3 Neumann - Dirichlet Boundaries



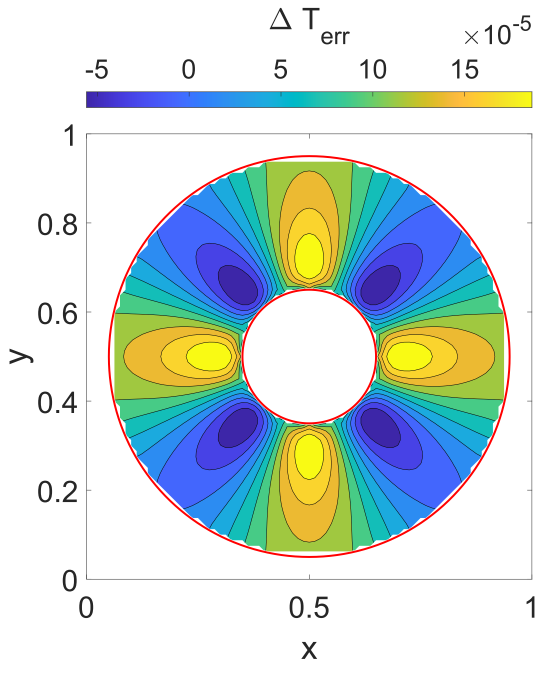
(a) Case No. 15



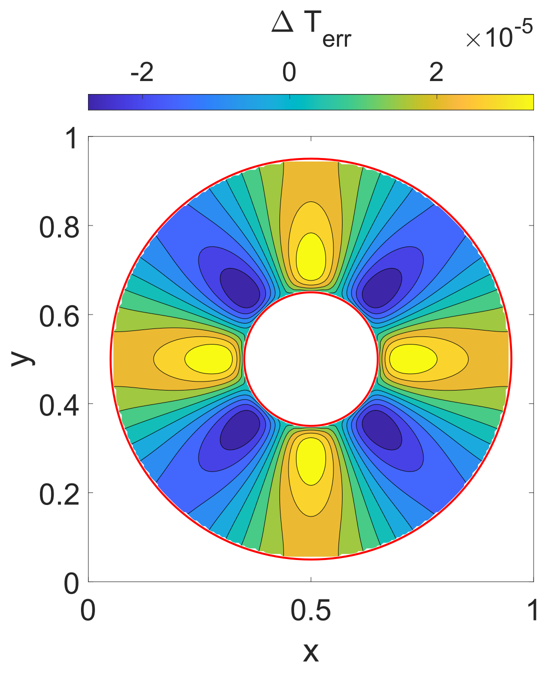
(b) Case No. 16



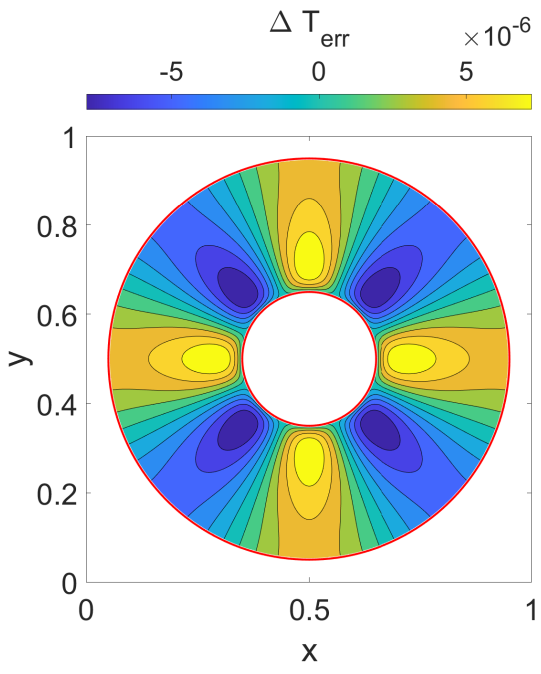
(c) Case No. 17



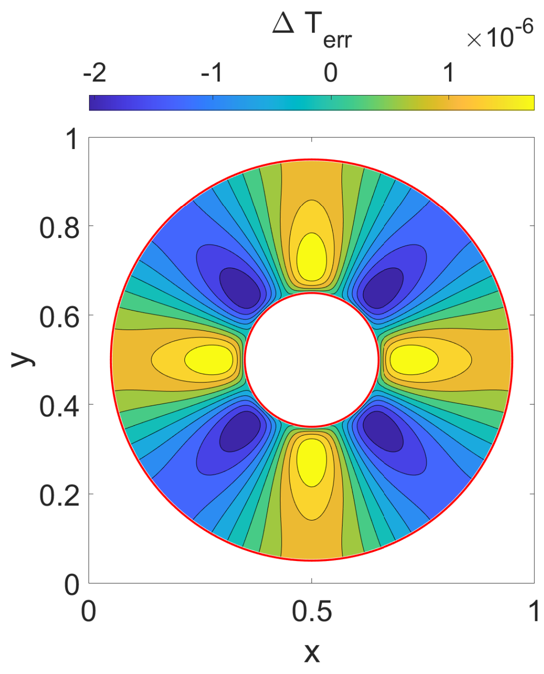
(d) Case No. 18



(e) Case No. 19

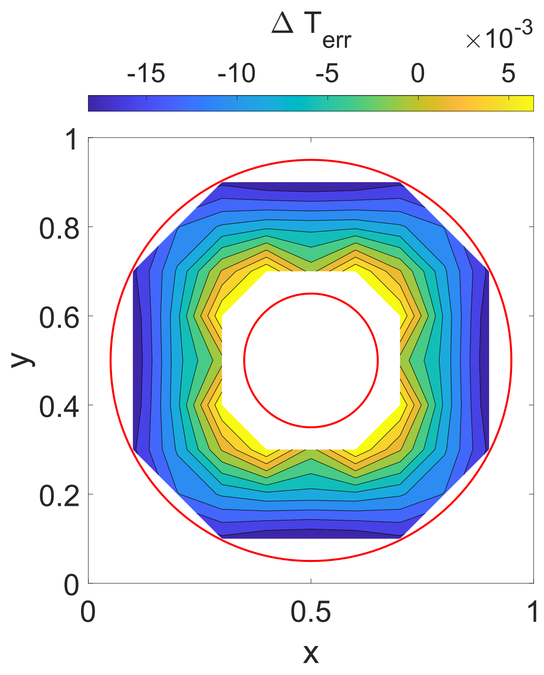


(f) Case No. 20

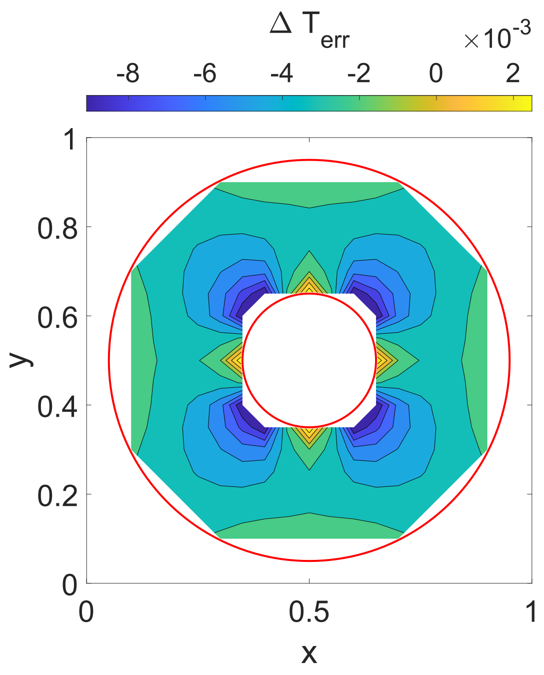


(g) Case No. 21

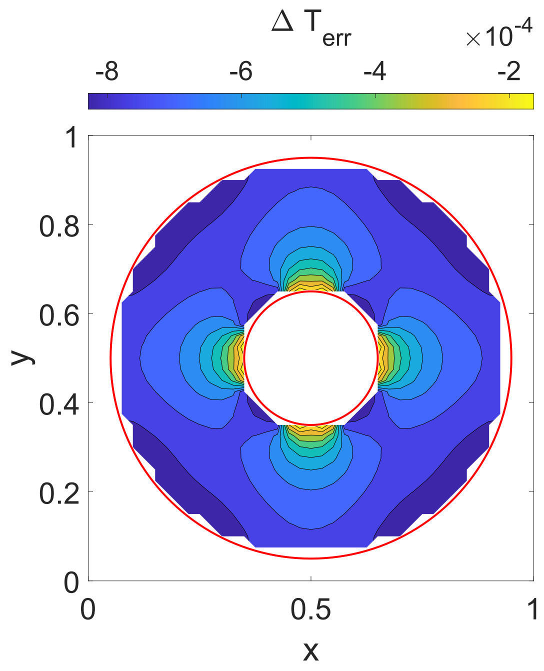
Figure A.5: Error contours for WLSQ IBM cases with steady state heat conduction between concentric circles for Neumann BC on the outer circle and Dirichlet BC on the inner. Figures (a)-(g) show results corresponding to each case in table 4.3.



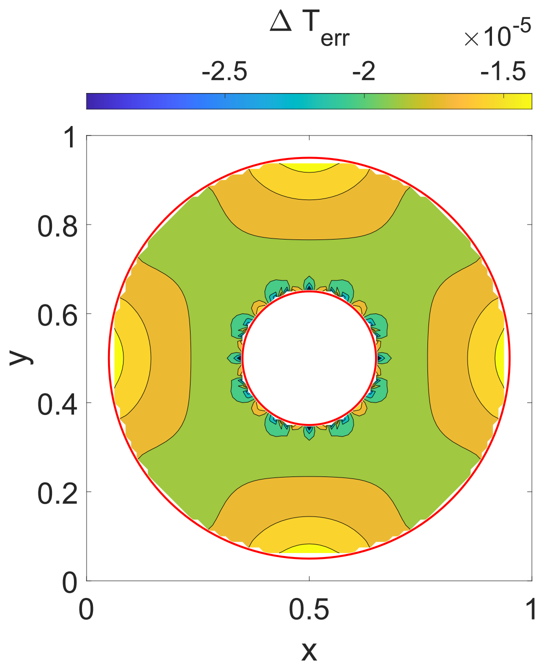
(a) Improved IBM on Case No. [15-16]



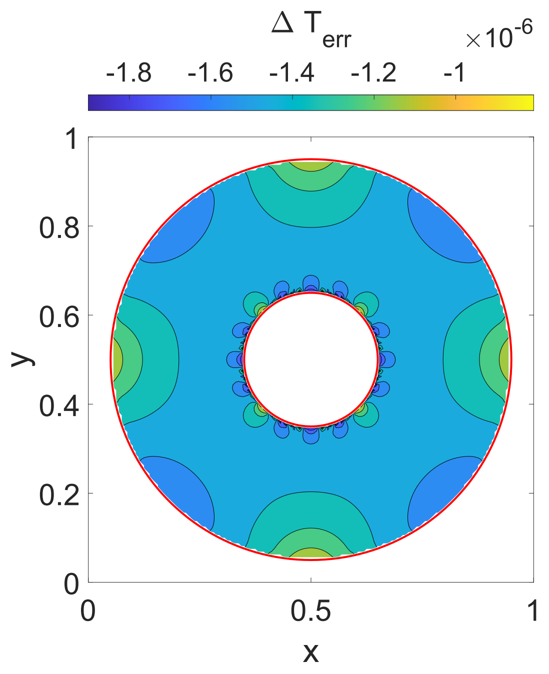
(b) Improved IBM on Case No. [16-17]



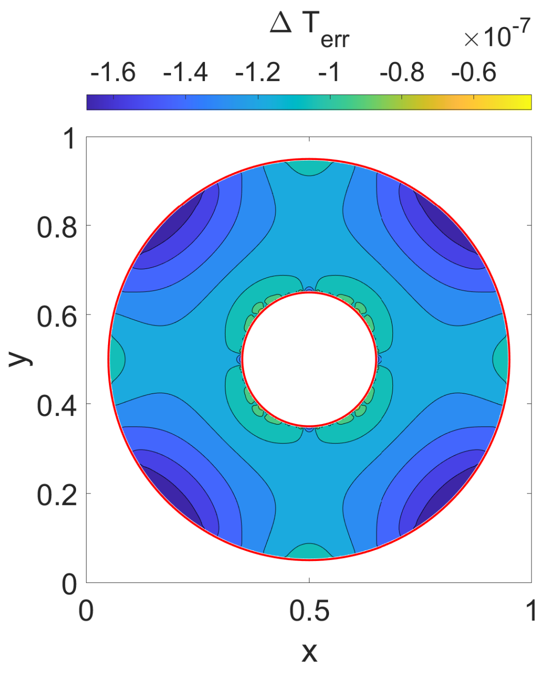
(c) Improved IBM on Case No. [17-18]



(d) Improved IBM on Case No. [18-19]



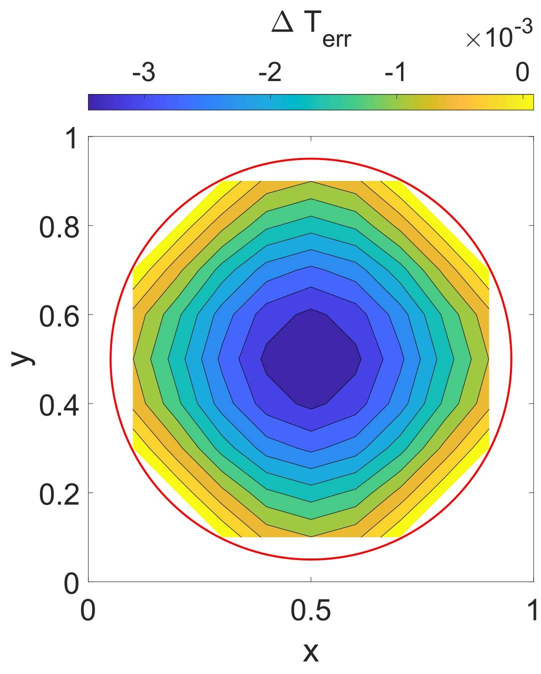
(e) Improved IBM on Case No. [19-20]



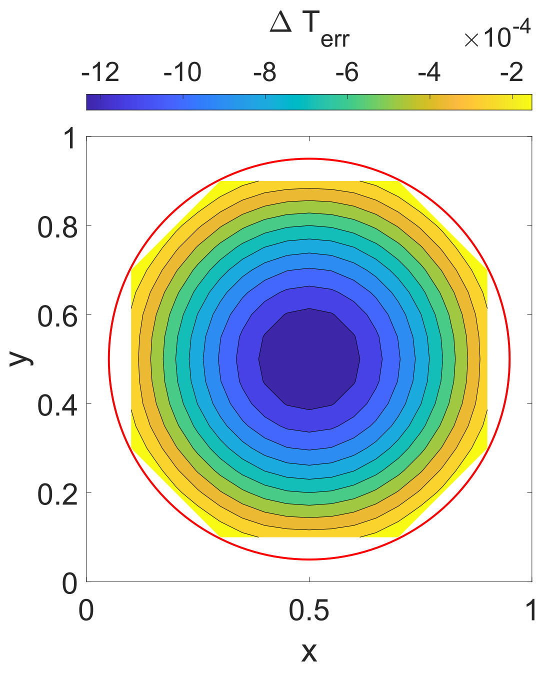
(f) Improved IBM on Case No. [20-21]

Figure A.6: Error contours for Improved IBM with steady state heat conduction between concentric circles for Neumann BC on the outer circle and Dirichlet BC on the inner circle. Figures (a)-(f) show results corresponding to each pair of cases in table 4.3.

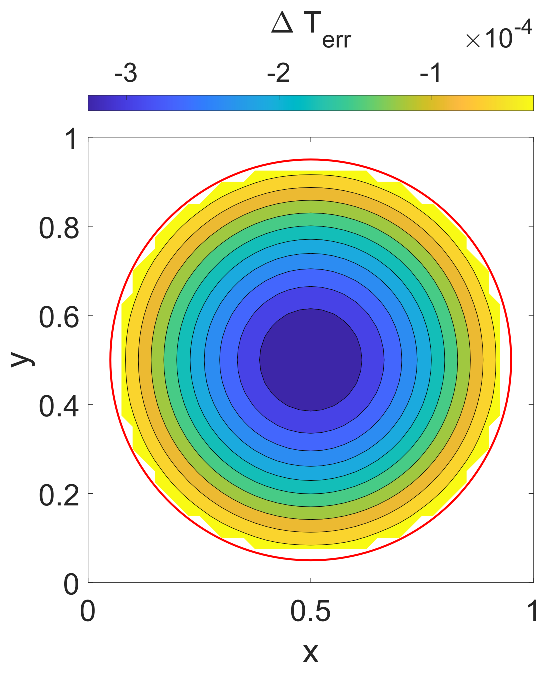
B Contour Plots - Heat Conduction in a Cross-Section of an Infinite Cylinder



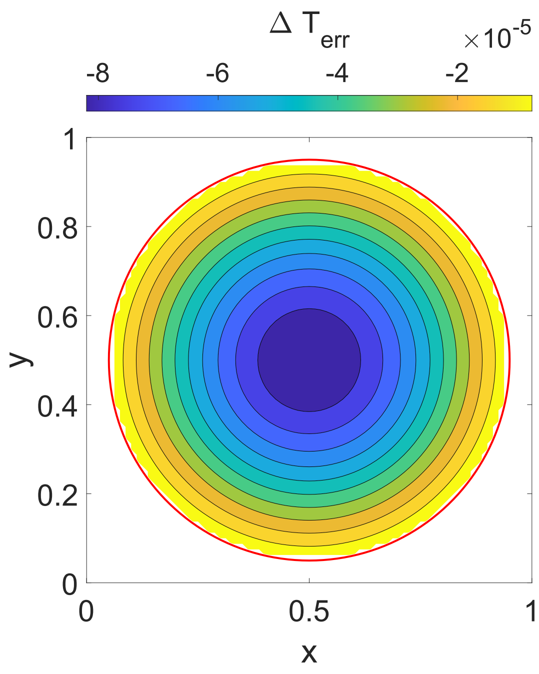
(a) Case No. 1



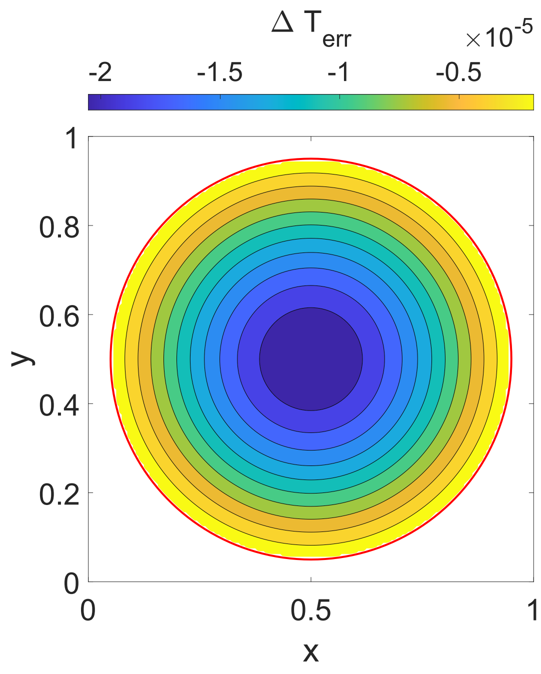
(b) Case No. 2



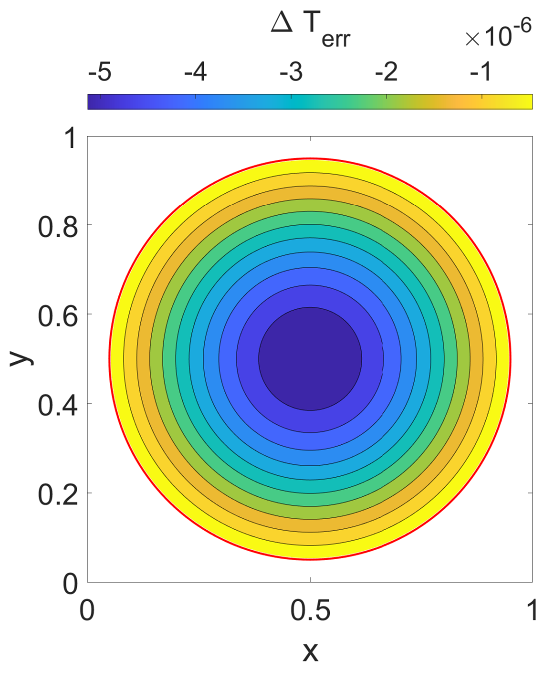
(c) Case No. 3



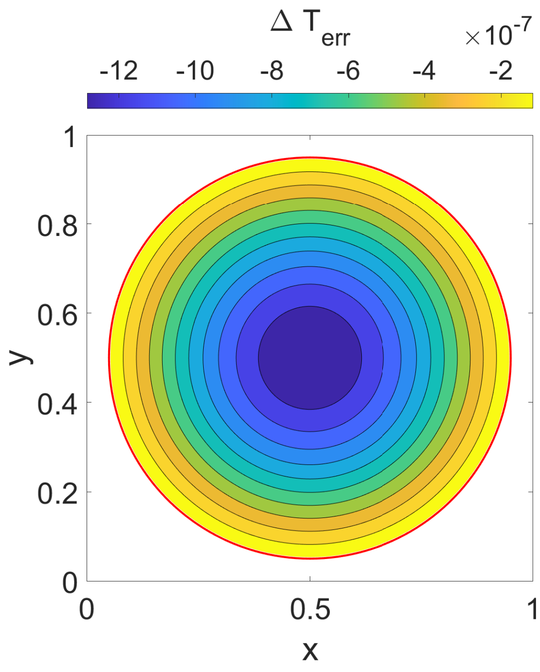
(d) Case No. 4



(e) Case No. 5

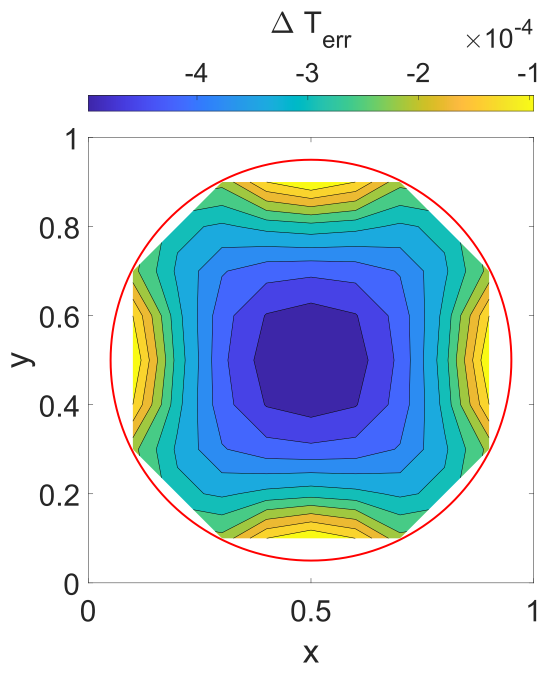


(f) Case No. 6

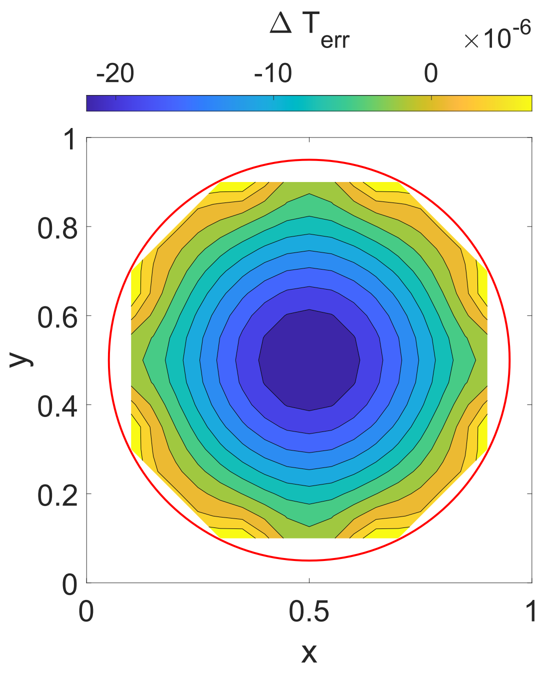


(g) Case No. 7

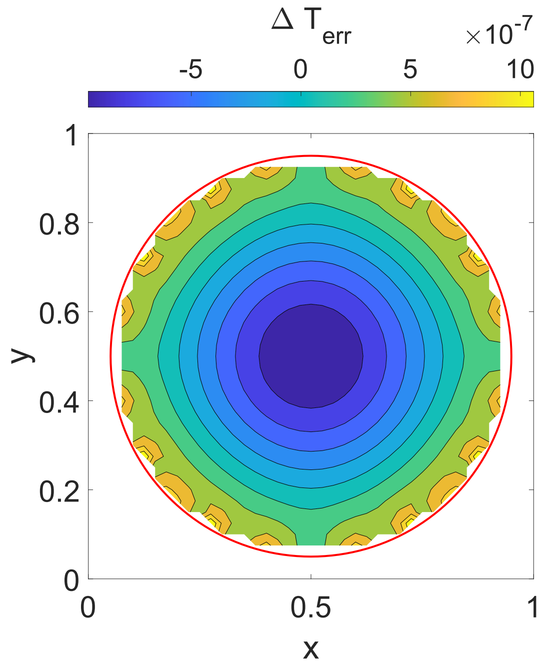
Figure B.1: Error contours for WLSQ IBM cases with transient heat conduction in a cross-section of an infinite cylinder with constant surface temperature. Figures (a)-(g) show results corresponding to each case in table 4.4.



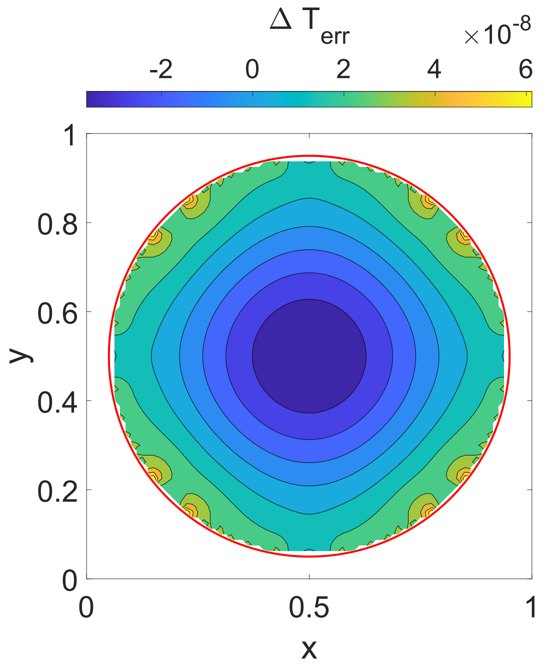
(a) Improved IBM on Case No. [1-2]



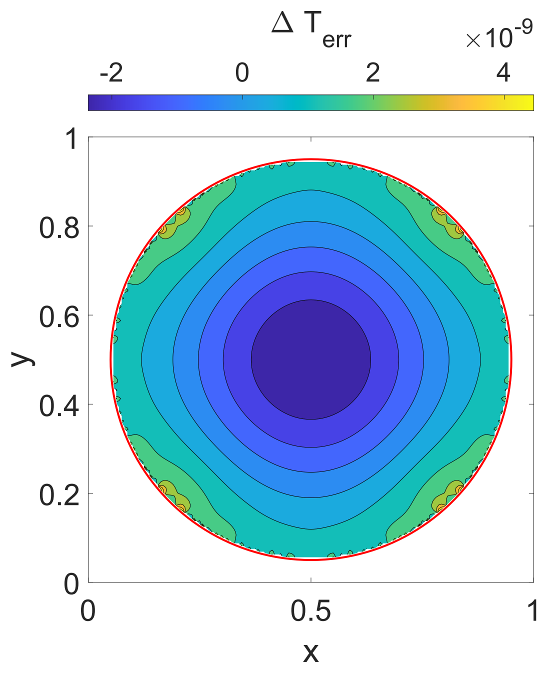
(b) Improved IBM on Case No. [2-3]



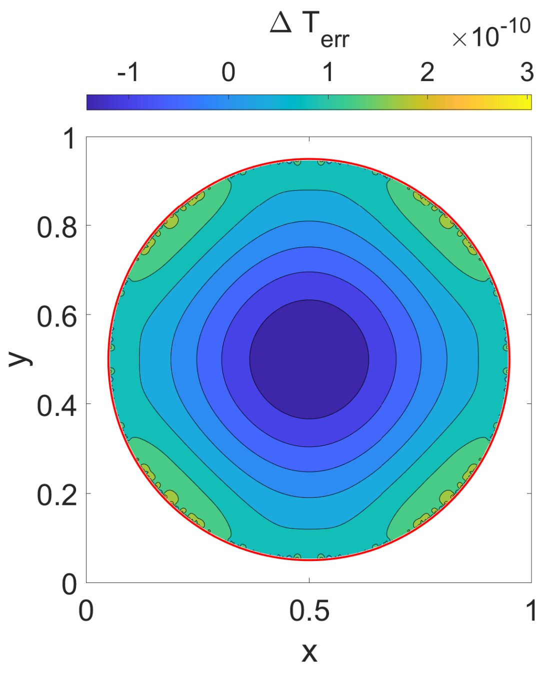
(c) Improved IBM on Case No. [3-4]



(d) Improved IBM on Case No. [4-5]



(e) Improved IBM on Case No. [5-6]



(f) Improved IBM on Case No. [6-7]

Figure B.2: Error contours for Improved IBM with transient heat conduction in a cross-section of an infinite cylinder with constant surface temperature. Figures (a)-(f) show results corresponding to each pair of cases in table 4.4.

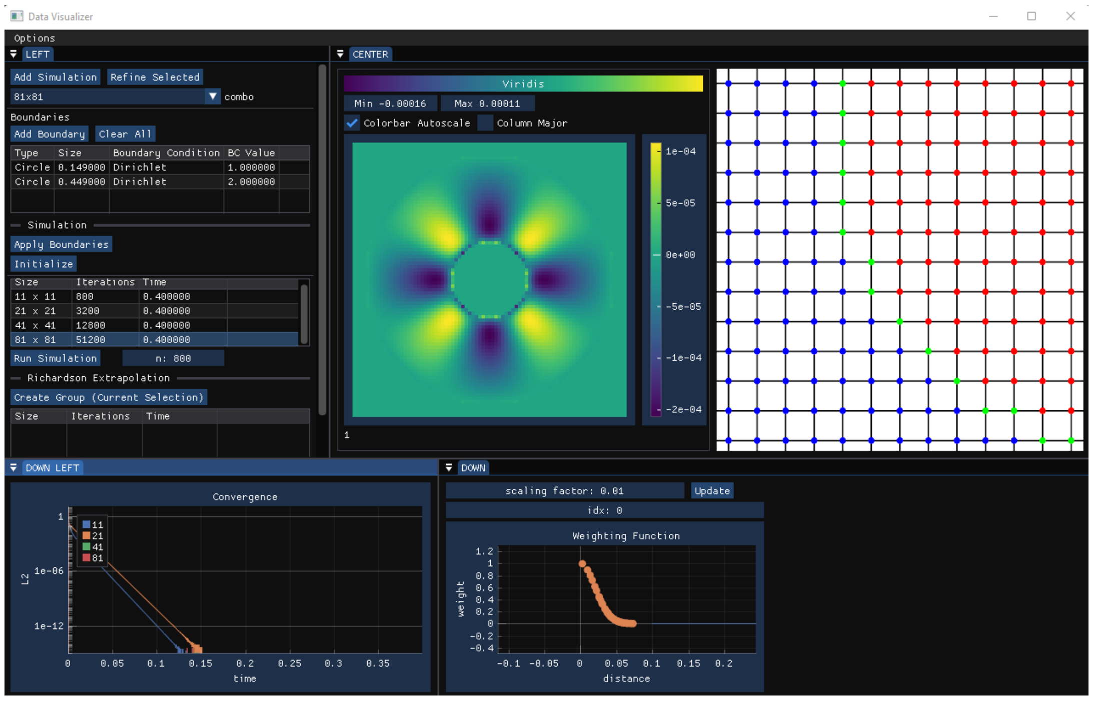
C Application Overview

The application which has been developed is structured in two parts. The first of which is the IBM Application containing the WLSQ IBM method and solver config. And a second part in the form of a data viewer which provides a GUI for setting up, running, and analyzing the simulations in real-time. The main classes of the IBM Application are:

1. GeometrySDF
2. CartGrid
3. Schemes
4. Solver

Which are responsible for the immersed boundaries, the cartesian grid, the FTCS scheme, and organizing the simulation tasks. The main libraries utilized in this area are

1. Eigen - Linear algebra
2. HDF5 - Hierarchical Data Format version 5
3. HighFive - High-level interface for HDF5



Type	Size	Boundary Condition	BC Value
Circle	0.149000	Dirichlet	1.000000
Circle	0.449000	Dirichlet	2.000000

Size	Iterations	Time
11 x 11	800	0.400000
21 x 21	3200	0.400000
41 x 41	12800	0.400000
81 x 81	51200	0.400000

Size	Iterations	Time

Figure C.0: View of the GUI window for running the simulations, in this case running the steady-state benchmark with Dirichlet boundaries. The application has two representations of the field, one on the right to inspect the geometry of the immersed boundary, and another on the left to view the error distribution.

The data viewer consists of a main UI layer, built on the following graphics libraries

1. OpenGL - Rendering API
2. ImGui - Immediate mode GUI framework
3. ImPlot - Plotting library for ImGui

An example of the application is presented in figure C.0 where an example simulation is performed, with the view focused on the $N = 81$ grid.

The application including all source codes is available at

https://github.com/TerjeTL/ImprovedIBM

D Source Code

```cpp
void FTCS_Scheme::Update(double dt, double r)
{
    BoundaryCondition();

    phi_old = m_mesh_grid->GetPhiMatrix();
    Eigen::MatrixXd& phi = m_mesh_grid->GetPhiMatrixRef();

    auto grid_extents = m_mesh_grid->GetMeshSize();

    #ifdef MT_ON
        #pragma omp parallel for num_threads(4)
    #endif
    for (int j = 1; j < grid_extents.first-1; j++)
    {
        for (int i = 1; i < grid_extents.second-1; i++)
        {
            // skip if node is a ghost point/inactive
            if (m_mesh_grid->GetCellFlag(i,j) != 0)
            {
                continue;
            }

            phi(j, i) = phi_old(j, i) + 1.0 * r * (phi_old(j, i + 1)
                        - 2 * phi_old(j, i) + phi_old(j, i - 1)
                        + phi_old(j + 1, i) - 2 * phi_old(j, i)
                        + phi_old(j - 1, i));
        }
    }

    euclidian_norm = (phi - phi_old).squaredNorm();
}
```

Source Code 1: Function implementation of the FTCS scheme.

```cpp
//--------------------
// WLSQ Main Update
//--------------------
// This function updates all ghost point values using the WLSQ method. Prerequisites for
// this method is that the wlsq data of each ghost point has been previously initialized
// with WLSQInit, and that WLSQUpdateGeometry has been called if there have been any changes
// to the configuration such as the weight function or boundary geometries.
void CartGrid::WeightedLeastSquaresMethod()
{
    for (auto& [ij, wlsq] : m_wlsq_data)
    {
        // Update vector of phi values corresponding to the node selection
        int n = 0;
        for (auto [i, j] : wlsq.m_num_stencil_reduced) // pre-sliced numerical stencil
        {
            wlsq.m_phi_vec_reduced[n] = phi_matrix(j, i);
            n++;
        }

        double linear_comb_sum = wlsq.m_M_boundary.dot(wlsq.m_phi_vec_reduced);
        wlsq.m_gp_val = wlsq.m_bc_term - linear_comb_sum * wlsq.m_M_den;
    }
}
```

Source Code 2: Function implementation of the WLSQ .

```cpp
void CartGrid::WLSQUpdateGeometry()
{
    // Keep track of maximum condition number
    double cond_max = 0.0;

    // Perform the update on all ghost points
    for (auto& [ij, wlsq] : m_wlsq_data)
    {
        const int i = ij.first;
        const int j = ij.second;

        wlsq.ghost_point = { i, j };

        // Obtain the immersed boundary and normal direction to the
↪   boundary
        size_t parent_sdf = ghost_point_parent_sdf(j, i);
        Eigen::Vector2d world_loc = GetWorldCoordinate(Eigen::Vector2d{
↪   i, j });
        Eigen::Vector2d unit_normal =
↪   immersed_boundaries.at(parent_sdf)->GetNormal(world_loc.x(),
↪   world_loc.y());

        // Define the desired stencil size
        const int stencil_size_required = 35;

        ConstructNumericalStencil(wlsq, stencil_size_required);

        // Construct the Vandermonde matrix, starting with GP as first
↪   row
        wlsq.m_vandermonde = ConstructVandermonde(3, wlsq, wlsq.m_pos);

        // Construct the diagonal weight matrix
        wlsq.m_weight = ConstructWeightMatrix(m_weight_scaling, wlsq,
↪   wlsq.m_pos);

        // Compute the pseudoinverse (WV)^+
        Eigen::MatrixXd w_v_product = wlsq.m_weight * wlsq.m_vandermonde;
        auto pseudo_inv = PseudoInverseSVD(w_v_product);
        cond_max = std::max(cond_max, pseudo_inv.second);

        // Compute the final weight matrix
        wlsq.m_M = pseudo_inv.first * wlsq.m_weight;

        // Initialize related vars
        wlsq.m_phi_vec = Eigen::VectorXd::Zero(wlsq.m_active_nodes_num +
↪   1);
        wlsq.m_bc_type = GetBoundaryCondition(i, j);
        wlsq.m_unit_normal = unit_normal;
        wlsq.m_bc_value = GetBoundaryPhi(i, j);

        // Optimization measures (pre-slicing data)
        wlsq.m_num_stencil_reduced = std::vector<std::pair<int,
    ↪   int>>(wlsq.m_numerical_stencil.begin() + 1,
    ↪   wlsq.m_numerical_stencil.end());
        wlsq.m_phi_vec_reduced = wlsq.m_phi_vec(Eigen::seq(1,
    ↪   Eigen::placeholders::last));
        Eigen::VectorXd M_0 = wlsq.m_M(0, Eigen::seq(1,
    ↪   Eigen::placeholders::last));
        Eigen::VectorXd M_1 = wlsq.m_M(1, Eigen::seq(1,
    ↪   Eigen::placeholders::last));
        Eigen::VectorXd M_2 = wlsq.m_M(2, Eigen::seq(1,
    ↪   Eigen::placeholders::last));

        // Pre-compute terms for the main update function
        if (wlsq.m_bc_type == BoundaryCondition::Dirichlet)
        {
            wlsq.m_M_boundary = M_0;
            wlsq.m_M_den = 1.0 / wlsq.m_M(0, 0);
            wlsq.m_bc_term = wlsq.m_bc_value * wlsq.m_M_den;
        }
        else if (wlsq.m_bc_type == BoundaryCondition::Neumann)
        {
            auto n_x = wlsq.m_unit_normal.x();
            auto n_y = wlsq.m_unit_normal.y();

            wlsq.m_M_boundary = n_x * M_1 + n_y * M_2;
            wlsq.m_M_den = 1.0 / (n_x * wlsq.m_M(1, 0) + n_y *
    ↪   wlsq.m_M(2, 0));
            wlsq.m_bc_term = wlsq.m_bc_value * wlsq.m_M_den;
        }
    }

    //printf("Condition Number: %g\n\n", cond_max);
}
```

Source Code 3: Function implementation of the WLSQ Construction.

```cpp
// method for calculating the pseudo-Inverse as recommended by Eigen developers
template<typename MatrixType>
std::pair<MatrixType, double> PseudoInverseSVD(const MatrixType& mat, double epsilon = std::numeric_limits<double>::epsilon())
{
    Eigen::JacobiSVD< MatrixType > svd(mat ,Eigen::ComputeThinU | Eigen::ComputeThinV);

    double tolerance = epsilon * std::max(mat.cols(), mat.rows()) * svd.singularValues().array().abs()(0);
    double condition_number = svd.singularValues()(0) / svd.singularValues()(svd.singularValues().size() - 1);

    return { svd.matrixV() * (svd.singularValues().array().abs() > tolerance).select(svd.singularValues().array().inverse(), 0).matrix().asDiagonal() * svd.matrixU().adjoint(), condition_number };
}
```

Source Code 4: Function implementation SVD pseudoinverse.