Mathematical and Computer Modelling of Corrosion

Nouf Ruddah Allah Altalhi

Submitted in accordance with the requirements for the degree of Doctor of Philosophy

University of Leeds

Faculty of Engineering and Physical Sciences

School of Mathematics

Department of Applied Mathematics

December 2025

Publications

- Some of the material of Chapter 3 is included in:
  - N. R. Altalhi, D. Lesnic, and S. D. Griffiths. The boundary element method for solving rigid inclusion problems. *Proceedings of the Thirteenth UK Conference on Boundary Integral Methods (UKBIM13)*, (ed. D. Menshykov), Uni Print, University of Aberdeen, Chapter 5, pp. 41–50, 2023.
- Some of the material of Chapter 4 is included in:
  - N. R. Altalhi, D. Lesnic, and S. D. Griffiths. Solution of corrosion problems using the boundary element method. *WIT Transactions on Engineering Sciences*, Vol. 136, pp. 15–26, 2024.
- Some of the material of Chapter 6 is included in:
  - N. R. Altalhi, D. Lesnic, and S. D. Griffiths. Determination of a corroded boundary using the boundary elements method. *Proceedings of the Fourteenth UK Conference on Boundary Integral Methods (UKBIM14)*, (ed. E. Chadwick), University of Salford, Chapter 5, pp. 37–48, 2025.
- Some of the material of Chapters 5–7 is included in:
  - N. R. Altalhi, D. Lesnic, and S. D. Griffiths. Determination of boundary corrosion characteristics using the boundary element method. Submitted (in December 2025) to Engineering Analysis with Boundary Elements.

Acknowledgements

I would like to express my deepest gratitude to my supervisor, Professor Daniel Lesnic, for his outstanding supervision and continuous support. His advice throughout my PhD research, along with his willingness to share his knowledge and expertise, has been invaluable. Words cannot express my heartfelt gratitude for all the support, guidance, and time he provided during my study in Leeds.

My gratitude also extends to my co-supervisor, Dr. Stephen Griffiths for his insightful suggestions and continuous encouragement have been instrumental in shaping this research. His accessibility and readiness to assist at every juncture have made my Ph.D journey both productive and rewarding.

A special thank you to my country, Saudi Arabia, for its continuous support. I am also sincerely grateful to Taif University for their generous financial support and resources.

To my dearest parents, words cannot express the depth of my gratitude for your unwavering love, support, and sacrifice. Their belief in me has been a constant source of strength and motivation. To my mother, her unwavering belief in me and constant support have been my greatest source of strength. I dedicate this achievement to my parents as it is as much theirs as mine.

To my beloved brothers and sisters, I am deeply thankful for their presence in my life. Their love and support have been my strength pillar. A very special mention goes to my youngest brother, Abdullah. His constant support has been a source of light during the darkest times.

Finally, I would like to thank my friends, whose companionship and understanding made this journey both more enjoyable and meaningful.

Abstract

Corrosion poses a significant threat to engineering infrastructure such as pipelines and offshore constructions, especially in regions that are difficult to access and cannot be monitored directly. To prevent failures in these areas, indirect detection methods based on inverse modelling are necessary.

This thesis develops the Boundary Element Method (BEM) to discretise both direct and inverse problems related to steady-state corrosion, modelled by the Laplace's equation for the electric potential subjected to a Robin corrosion law on the corroded boundary that is not accessible to direct measurement. Inverse problems, such as reconstructing the corroded boundaries and coefficients from partial measurements, are highly ill-posed and require regularization techniques such as the Tikhonov or the truncated singular value decomposition methods to stabilize the solution in the presence of noise in the measured data.

The research addresses several specific cases of inverse problems, including Cauchy problems for data recovery with known geometry and coefficients, coefficient reconstruction, boundary identification, and joint reconstruction of both boundary and coefficients. These inverse problems are more difficult to solve than the corresponding direct problem since they are ill-posed i.e., either the existence, uniqueness or stability of solution is violated.

Numerical solutions are validated against analytical results and tested with various noise levels, demonstrating convergence and accuracy through stabilized systems solved by regularized least squares method.

Contents

List of Figures

List of Tables

Abbreviations

BEM	Boundary Element Method
EIT	Electrical Impedance Tomography
FDM	Finite Difference Method
FEM	Finite Element Method
LBC	Linear Boundary Condition
LS	Linear Least–Squares
MFS	Method of Fundamental Solutions
NHE	Normal Hydrogen Electrode
NLBC	Nonlinear Boundary Condition
PDE	Partial Differential Equation
PLTMG	Piecewise Linear Triangle Multigrid
SVD	Singular Value Decomposition
TSVD	Truncated Singular Value Decomposition

Nomenclature

A, B	Coefficient matrices
a	Constant (Chapter 2)
A^+	Moore–Penrose pseudo-inverse of matrix A
C	Space of continuous functions
C^1	Space of continuously differentiable functions

$C(r)$	Residual function
E	Error norms
$f(u)$	Nonlinear law of corrosion
F	Faraday constant (Chapter 1) and objective function (Chapter 5)
g	Given flux
g_1	Discontinuous current flux
g_2	Continuous current flux
G	Fundamental solution
$h^{(i)}$	Measured data at iteration i
$\underline{J}$	Current density vector
J_e	Current in the system
J_0	Exchange current density
L^2	Space of square integrable functions
$\underline{l}$	Vector containing lower bounds
M	Total number of boundary elements on the domain
n	Outward unit normal vector
N	Number of boundary elements
n_e	Number of electrons
p	Percent of noise
R_k	Regularisation derivative matrix
(r, θ)	Polar coordinates
SV	Singular values
S_j	j-th segment line
T	Absolute temperature
u	Potential function
$\underline{u}$	Vector containing upper boundary
U, V	Orthogonal matrices
$\underline{X}$	Variable vector
$\underline{b}$	Known matrix/vector

Greek Letters

α	Kinetic parameter (Chapter 1), symmetry parameter (Chapter 2) and scale
β	Scale
$\Gamma_i,\ \Gamma_j^i$	Sub-boundary
δ	Distance of source points to boundary
δ_{ij}	Kronecker delta symbol
Δ	Laplacian operator
Δ^2	Second-order differential operator
ϵ	Constant
η	Potential difference of $\phi_{eq} - \phi$
λ	Corrosion coefficient
μ	Regularization parameter
∇	Gradient operator
$\Omega,\ \Omega_0,\ \Omega_1$	Solution domains
$\overline{\Omega}$	Closure of domain Ω
$\partial\Omega$	Boundary of domain Ω
ϕ	Galvanic potential
ϕ_{eq}	Equilibrium galvanic potential
ρ	Random variable
σ	Conductivity
σ_n	Singular values
Σ	Diagonal matrix
Σ^+	Pseudo-inverse of matrix Σ
$\underline{\xi}$	Source points

Superscripts and Subscripts

$\cdot^{-1}$	Inverse of a matrix or operator
$\cdot^{(0)}$	Initial guess
$\cdot'$	Normal derivative
$\cdot^{\mathrm{T}}$	Transpose of a matrix
$\cdot^{++}$	Ions, positive quantities
$\cdot^{\mathrm{noisy}}$	Perturbation in data
$\cdot_{\mathrm{exact}}$	Exact solution
$\cdot_{\mathrm{max}}$	Maximum value
$\cdot_{\mathrm{min}}$	Minimum value
$\cdot_{\mathrm{num}}$	Numerical solution
$\cdot_{\mathrm{reg}}$	Regularisation of objective function
$\frac{\partial}{\partial n}$, ∂_n	Normal derivative

Chapter 1

Introduction

1.1 Motivation and background

Corrosion, fundamentally an electrochemical phenomenon, presents a persistent challenge in engineering and material science. It refers to the gradual degradation of metals due to interactions with their surrounding environment, often through chemical or electrochemical reactions. These reactions take place at the interface between a metal and an electrolyte (a fluid that conducts ions), involving oxidation at the anode, where the metal loses electrons and releases positively charged ions into the electrolyte, and a corresponding reduction at the cathode, where electrons are consumed by species like dissolved oxygen or hydrogen ions. These redox processes form a self-sustaining electrochemical circuit, resembling a galvanic cell, in which electrical energy is released due to potential differences between dissimilar metals immersed in an electrolyte [47, 52].

In buried pipeline corrosion, the corroding sections of the pipe act as anodes. The cathode, however, is not a separate metal strip but can be: (i) non-corroding areas on the same pipe where oxygen is more readily available, (ii) nearby buried metallic structures in electrical contact with the pipe, or (iii) intentionally installed sacrificial anodes used for cathodic protection. The electron transfer between these anodic and cathodic sites generates a current that highlights the dual role of electrochemical systems as both energy sources and sites of material deterioration.

The kinetics of corrosion are governed by a complex interplay of factors, including the potential difference across electrodes, the electrolyte's composition and the metal's surface properties. In this context, the buried pipeline surface itself serves as the electrode, with its potential measured relative to a reference electrode in the surrounding soil electrolyte. At the metal-electrolyte interface, the electrochemical reactions are modulated by the surface overpotential, defined as the deviation of the electrode potential from its equilibrium value, which drives the net current flow and thus the corrosion rate [43]. The Butler-Volmer equation, a fundamental tool in electrochemical theory, quantifies this relationship by expressing the current density as a non-linear function of the overpotential, with exponential terms reflecting the activation energies of the anodic and cathodic reactions [97]. This nonlinearity poses significant mathematical challenges, particularly in regions where the metal surface transitions to a passive state, often due to the formation of an oxide layer that inhibits further corrosion. Such transitions, common in metals like iron or aluminium, complicate modelling efforts, as the electrochemical activity varies spatially and temporally, necessitating sophisticated mathematical frameworks to accurately capture the system's behaviour [97].

To address these complexities, the potential model offers a simplified yet powerful approach, assuming a homogeneous electrolyte with constant ion concentrations away from the electrode surface. This assumption reduces the governing equations to the Laplace's equation for the electric potential within the electrolyte, coupled with non-linear boundary conditions derived from the Butler-Volmer formula. At the electrochemically active boundary, the current density depends exponentially on the overpotential, capturing the rapid changes in reaction rates as the potential shifts. The inactive boundary segments, portions of the pipe surface that are electrically insulated, coated, or otherwise non-reactive, while not participating in electrochemical reactions, may still conduct imposed currents from external cathodic protection systems, adding further complexity to the system [97]. The resulting non-linear boundary value problem, as formulated by Vogelius and Xu [97], transforms the potential into a variable that accounts for the external circuit's driving force, yielding a harmonic equation with a non-linear Robin boundary

condition. This framework enables the analysis of solution, particularly under varying electrochemical conditions, providing insights into the spatial distribution of corrosion along the electrode surface.

The ability to predict corrosion's spatial variability is crucial for developing effective mitigation strategies, such as cathodic protection or protective coatings. Corrosion is rarely uniform; localized forms, such as pitting or crevice corrosion, can lead to catastrophic failures in critical infrastructure like pipelines or ship hulls exposed to aggressive electrolytes. The non-linear boundary conditions in the potential model account for these variations by incorporating the effects of surface overpotential and exchange current density, which are influenced by electrolyte composition, temperature and surface characteristics [73]. Furthermore, the model's focus on the transition to passivity — where an oxide layer reduces the metal's reactivity — highlights a key aspect of corrosion control [87]. This passive state, while protective, is inherently unstable and can break down under specific conditions, leading to renewed corrosion activity. The mathematical analysis by Vogelius and Xu [97] reveals the possibility for multiple solutions in the transition region, reflecting the bi-stable nature of the electrochemical system and underscoring the complexity of corrosion dynamics.

1.1.1 Fundamentals of electrochemical corrosion

Electrochemical theory, specifically the galvanic cell concept, serves as the foundation for mathematical descriptions of corrosion. In such systems, chemical energy is converted to electrical energy via two spatially separated electrodes: an anode for oxidation and a cathode for reduction. The electrodes are immersed in an ionically conducting electrolyte and connected externally to complete the circuit [84].

1.1.1.1 Galvanic cells and electron flow

Deconinck [41] explained that an electrolyte flow of ions travels from one electrode to the other as a result of the flow of electrons in the outer circuit. Additionally, when an electrical potential is applied across the electrodes, the anode goes through oxidation

and the cathode goes through reduction. The reduction and oxidation reactions transport electrons in between electrodes, resulting in current flow in the circuit. Thus, the positive electrode is the cathode, and the negative electrode is the anode in a galvanic cell.

As a consequence of interactions between the electrode materials and the electrolyte, corrosion of the electrodes may also take place [41]. Corrosion is an unwanted phenomenon that affects many different kinds of combinations and materials. There are numerous variables, such as the potential difference between electrodes, the sort of liquid electrolyte and the electrode surface area, that impact the rate of corrosion. For instance, if we take two metals, such as copper and zinc, and submerge them partially in saltwater, then by connecting them in a galvanic cell, the electrons will flow externally through the circuit from the anode to the cathode. This occurs due to the difference in potential energy between the two electrodes in the electrochemical cell, as shown in Figure 1.1, see [102]. The cathode is the less active metal, while the anode is the more active metal. In this setup, zinc corrodes because the zinc ions Zn^{++} dissolve into the electrolyte, whereas copper remains safeguarded. When an electrical current flows, the anode eventually transforms into ions. To be precise, chemical reduction happens at the copper electrode (positive pole), while oxidation occurs at the zinc electrode (negative pole). Electrons flow automatically from zinc to copper ions in the electrolyte, which the cathode receives via the metallic connection with the anode. This reaction creates zinc ions and metallic copper, and, as a result, the cathode becomes negatively polarized and is shielded from corrosion. The quantity of zinc that corrodes is directly proportional to the amount of energy that flows through the cell.

At the zinc electrode (anode):

$$Zn(s) \longrightarrow Zn^{++} + 2e^{-} \qquad \text{oxidation.}$$

At the copper electrode (cathode), copper ions Cu^{++} are present in the electrolyte solution:

$$Cu^{++} + 2e^{-} \longrightarrow Cu(s) \qquad \text{reduction,}$$

which contains dissolved copper salt such as copper sulfate. These ions belong to the electrolyte rather than being initially released by the copper metal. They are instead reduced and deposited as metallic copper onto the cathode surface during cell operation.

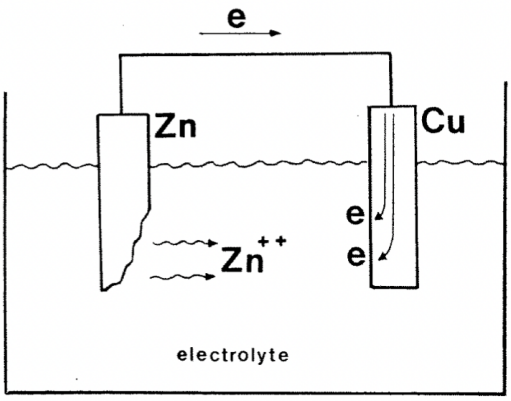

Figure 1.1: Sketch of the flow of electrons from zinc to copper and the solubility of Zn^{++} ions in the electrolyte, [102].

The concentration gradient of the zinc and copper ions near the electrodes will affect the rate of the chemical reactions and the transfer of electrons. For example, if the concentration of zinc ions near the anode is high, the rate of the oxidation reaction will be high, and the rate of the reduction reaction will be low. On the other hand, if the concentration of copper ions near the cathode is high, the rate of the reduction reaction will be high, and the rate of the oxidation reaction will be low.

1.1.1.2 General electrochemical reactions and overpotential

At the interface between electrode and electrolyte, electrochemical reactions contribute to the overall reaction rate where the oxidation and reduction reactions take place. The electron flow in the outer circuit is transformed into an ion flow in the electrolyte [42]. In general, the oxidised and reduced forms are denoted by the symbols O and R, and ne represents a specific number of electrons, so:

$$O + ne \rightleftharpoons R,$$

where the symbol $\rightleftharpoons$ indicates reversibility, meaning the reaction can proceed in either direction under different electrochemical conditions (forward for oxidation, backward for reduction).

There will be a simultaneous transfer of material and charge in the solution. In the case that an electrode is in contact with a solution, a new arrangement of solvent dipoles, ions in the solution and electrode electrons is produced at the interface. On each side of the contact surface, there are equal and opposite amounts of charge. This causes an electrical field to form, which is called the "electrical double layer."

The absolute value of the potential difference across electrode-electrolyte interphase cannot be measured. Thus, a normal hydrogen electrode (NHE) is used as the reference to make relative measurements possible [41]. The potential difference at equilibrium measured in such a cell is called the equilibrium potential or the Nernst potential E_0.

When a potential difference is applied between the electrode and a counter electrode (another strip of metal with a different Nernst potential), a current flows between these electrodes of the electrolytic cell. The galvanic potential ϕ is the electrode potential out of equilibrium (measured with a reference electrode). The difference $\phi - \phi_{eq}$ is referred to as the surface overpotential. This is discussed in greater detail in [42, 50, 87]. The driving force behind the net current (reaction) is the overpotential. The overpotential η is positive for oxidation reactions and negative for reduction reactions.

1.1.2 Mathematical formulation

The governing partial differential equations (PDEs) are established by considering a control region, Ω, within a conductive medium (electrolyte) that has a uniform conductivity σ. To satisfy the continuity of electrical charge, the net charge accumulation in Ω should be zero. Hence, this condition can be expressed as

$$\nabla \cdot \boldsymbol{J} = 0 \quad \text{in} \quad \Omega, \tag{1.1}$$

where the current density vector, $\boldsymbol{J}$, is related to the electrical potential ϕ through

$$\boldsymbol{J} = -\sigma \nabla \phi, \tag{1.2}$$

under the assumption that, for simplicity, no current is produced by any sources (or sinks). By combining equations (1.1) and (1.2), we obtain

$$-\nabla \cdot (\sigma \nabla \phi) = 0 \quad \text{in} \quad \Omega. \tag{1.3}$$

Assuming that the medium is homogeneous, the conductivity σ can be eliminated and equation (1.3) reduces to the classical Laplace equation

$$\nabla^2 \phi = 0 \quad \text{in} \quad \Omega. \tag{1.4}$$

At the electrode surface, which for buried pipelines is the outer surface of the pipe in contact with the soil electrolyte, the current density is related to the potential ϕ by

$$\boldsymbol{J} \cdot \boldsymbol{n} = -\sigma \frac{\partial \phi}{\partial n}, \tag{1.5}$$

where $\boldsymbol{n}$ is the unit outward normal (relative to the electrolyte). Concerning the boundary conditions associated with (1.4), we assume that one part of the boundary is electrochemically active, facilitating oxidation-reduction reactions, while the other part remains inactive, although a current flux can still be applied to it. At the electrochemically active metal-electrolyte boundary, due to the polarization effect, the current flux is described as a nonlinear function of the potential difference $\eta := \phi - \phi_{eq}$ given by [46]

$$J_e := \boldsymbol{J} \cdot \boldsymbol{n} = J_0 f(\phi - \phi_{eq}), \tag{1.6}$$

where J_e is the current in the system, J_0 is the exchange current density, ϕ_{eq} is the equilibrium potential of the electrode and f is a polarization response function that may be known or unknown.

Under certain ideal conditions, the expression for J_e, may be given by the so-called Butler–Volmer equation. The Butler–Volmer formula [69, 88, 97] is a mathematical expression that describes the relationship between the potential difference between the electrodes and the rate of chemical reactions that occur in the system. The equation takes into account the effects of the ionic distribution and the transfer of electrons at the electrodes and is used to model the behaviour of electrochemical systems. The equation can be written as:

$$J_e = J_0 \left[\exp\left(\frac{\alpha F}{RT}\eta\right) - \exp\left(\frac{(1-\alpha)F}{RT}\eta\right)\right], \tag{1.7}$$

where the transfer coefficient α is a kinetic parameter that in the range $(0, 1)$, F is the Faraday constant that represent the electrical charge per mole of electrons ($F \approx 96485$ C/mol), $R \approx 8.314 J/(mol\ K)$ is the universal gas constant (a fundamental constant that appears anytime energy-per-mole and temperature are related, including when dealing with solutions and solids in electrochemistry), T denotes the absolute temperature in Kelvin and $\eta := \phi - \phi_{eq}$ represents the surface overpotential (the deviation of the electrode potential from its equilibrium value), see [88]. Let $\gamma := \frac{F}{RT}$ such that (1.7) becomes

$$J_e = J_0 \Big[\exp(\alpha\gamma\eta) - \exp((1-\alpha)\gamma\eta)\Big]. \tag{1.8}$$

In the symmetric case where $\alpha = 0.5$, equation (1.8) can be simplified to

$$J_e = 2J_0 \sinh\left(\frac{\gamma\eta}{2}\right), \tag{1.9}$$

where $\sinh(x) = (e^x - e^{-x})/2$ is the hyperbolic sine function.

The Butler-Volmer equation (1.8) is often used in electrochemical theory to describe the local behaviour of a metallic electrolyte system [73]. It is applicable to all circumstances of anodic or cathodic current flow, depending on the sign and amplitude of the overpotential. In general circumstances, it might not be possible to derive an analytical expression for the non-linear function f appearing in equation (1.6).

Vogelius and Xu [97] described the steps for producing a non-linear boundary value problem from the Butler-Volmer equation. To follow their notation and non-dimensionality, where we shift by the equilibrium potential ($\phi_{eq} = 0$) and scale by defining $u := \gamma\eta = \frac{F}{RT}\eta = \frac{F}{RT}(\phi - \phi_{eq})$ (so u and ϕ represent the same physical potential up to this affine transformation), we will investigate the following mathematical model:

$$\begin{cases} \Delta u = 0 & \text{in } \Omega, \quad (1.10a) \\ \dfrac{\partial u}{\partial n} = \lambda\Big(e^{\alpha u} - e^{-(1-\alpha)u}\Big) + g & \text{in } \Gamma_0, \quad (1.10b) \\ \dfrac{\partial u}{\partial n} = g & \text{in } \partial\Omega\backslash\Gamma_0. \quad (1.10c) \end{cases}$$

where $\Gamma_0 \subset \partial\Omega$ represents the corroded portion of the boundary and $\lambda = -J_0\,\gamma/\sigma$.

1.2 Overview of modelling boundary corrosion

Corrosion represents a major challenge in engineering infrastructure, causing significant economic losses across many industries. Beyond the economic impact, corrosion-induced structural failures present serious safety concerns. The primary challenge in corrosion management is detecting and measuring corrosion damage that occurs of inaccessible locations. Consider a buried pipeline, the internal surface of a storage tank or the embedded portion of a structural foundation. In each case, critical corrosion damage develops where direct inspection becomes impossible. This creates an inverse problem: determining the extent, location and severity of hidden corrosion damage using measurements taken on accessible surfaces.

From a mathematical perspective, corrosion detection reduces to solving inverse boundary value problems for the Laplace's equation (2.1), where the electric potential u must be determined in a domain with partially unknown boundary characteristics. The corroded regions create changes that affect the electromagnetic field distribution. Moreover, the relationship between external measurements and internal geometric changes is nonlinear and ill-posed. These inverse problems differ from the more established direct problems of electrochemistry and potential theory. While computing electromagnetic fields for known

geometries is well-understood, the inverse process of determining boundary characteristics from extra measurements remains challenging. The ill-posed nature of these inverse problems requires regularization techniques and robust numerical methods to achieve stable solutions.

This thesis addressed a specific computational approach to inverse corrosion problems. The geometric setup and physical problem formulation built upon the foundational work of Vogelius and Xu [97], who established the mathematical framework for nonlinear elliptic boundary value problems in corrosion modeling. Their numerical solution for the direct problem employed the finite element method (FEM) implemented through PLTMG (Piecewise Linear Triangle Multi-Grid), utilizing sophisticated multigrid refinement techniques to handle the nonlinearity of the boundary condition. Alternatively, the BEM has found successful application in various inverse problems — including crack detection in structural mechanics [9], shape optimization in engineering design [89, 95] and parameter identification in heat transfer [78] — its potential for corrosion detection problems has remained less unexplored. This represents a significant opportunity, particularly given the BEM's natural advantages for boundary-dominated problems: reduced computational dimensionality, exact satisfaction of governing differential equations in the interior and inherent capability for handling unbounded domains typical of electromagnetic field problems. The research contribution of this thesis bridges this methodological gap by demonstrating that the BEM provides a computationally efficient alternative approach for corrosion detection. By exploiting the BEM's boundary-focused formulation, this work achieves significant computational advantages while maintaining the mathematical rigor of the original formulation. This research establishes the BEM as a viable numerical platform for inverse corrosion detection problems. The boundary-integral formulation provides a foundation for future extensions to three-dimensional problems and multiple defect scenarios. The significance extends to electrochemistry and material science, where the validated BEM framework offers a new computational tool for investigating corrosion processes.

1.3 Inverse problems

Inverse problems have emerged as a crucial area of study in various scientific and engineering disciplines, offering valuable insights and solutions to complex real-world challenges [17, 75, 92]. These problems involve the reconstruction of unknown parameters or characteristics of a system based on observed or measured data, leading to significant advancements in fields such as medical imaging, geophysics, material science, remote sensing and non-destructive testing.

The concept of solving inverse problems, determining causes from observed effects, has deep historical roots and has been fundamental in various practical applications. The inherent complexity of inverse problems lies in their ill-posed nature; small variations in observed data can lead to significant differences in the inferred causes, or multiple causes can produce similar effects. This ill-posedness, first mathematically explored by Abel in 1826 through his work on mechanical problems, has profound implications on the difficulty and methods for solving these problems.

Significant developments occurred during World War II with the invention of radar and sonar, which sparked interest in inverse scattering problems. These focused on determining target locations from transmitter/receiver antennas and constructing detailed images of these targets. This period marked the beginning of new reconstruction methods that have since been applied in fields like nondestructive testing, biomedical imaging, seismology and atmospheric profile inversions. The 1980s and 1990s witnessed further advancements with the development of inversion methods in scattering and tomography [40], while qualitative methods gained traction around 1996. The early 2000s brought increased interest in addressing sparsity, leading to new regularization tools and approaches, particularly in the context of data assimilation within dynamical systems and forecasting. The integration of multiphysics-based methods has since provided deeper insights into complex systems, such as the human body and the atmosphere.

The rapid advancement of computational power and technology has further enabled the solution of large-scale inverse problems with greater accuracy, opening new avenues

for scientific research and practical applications.

Typically, inverse problems involve identifying inputs from outputs [17]. These problems often present significant challenges as they are frequently ill-posed or ill-conditioned, particularly when the underlying transformation is nonlinear or unknown, see [98].

A problem is considered ill-posed if it does not satisfy the well-posedness conditions of Hadamard: a solution must exist, be unique and depend continuously on the input data. These three criteria differentiate well-posed direct problems from ill-posed inverse problems. To clarify this distinction, consider the corrosion boundary value problem (1.10). The direct problem, which is studied in detail in Section 2.1, consists of determining the potential u in the domain Ω when the current fluxes on the sub-boundaries of $\partial\Omega$ are fully specified. Specifically, on the electrochemically active boundary Γ_0, the current flux is governed by the nonlinear Butler-Volmer kinetics and any imposed current g, while on the inactive boundary $\partial\Omega\backslash\Gamma_0$, only the imposed current flux g is present (see equations (2.2) and (2.3) in Section 2.1). In this direct problem, all parameters are known: the domain geometry Ω, the corroded boundary Γ_0, the corrosion coefficient λ and the boundary flux g. This direct problem is well-posed and can be solved numerically using standard methods of discretisation such as the boundary element method (BEM), as detailed in Chapter 2. In contrast, an inverse problem arises when one or more of the above parameters or boundary specifications are unknown and must be determined from available measurements. Such problems typically violate one or more well-posedness criteria of Hadamard and require regularization techniques for stable numerical solution. Noise in the measurement data can further exacerbate the difficulty of solving these inverse problems.

Inverse problems can generally be classified into the following types, depending on the missing or combined requirements, [5, 75]:

1. The identification of parts of the boundary of the solution domain,
2. The identification of the value of the coefficient in the boundary condition,
3. The identification of the shape, size and location of the domain,
4. The identification of the boundary conditions and/or initial conditions.

In practical applications, experimental impediments often arise when attempting to measure or enforce certain conditions. The key feature in interpreting experimental results for these problems is the need to derive results from indirect manifestations of the object that can be measured experimentally. Thus, we are dealing with inverse problems, where we must determine the causes given the observed results. In fact, inverse problems are recognized as the opposite of direct problems, yet they play one of the most crucial roles in science and engineering. Compared to direct problems, inverse problems often pose a greater challenge as they frequently exhibit non-linear characteristics and are typically ill-posed [17], meaning they do not meet the well-posedness criteria established by Hadamard, specifically, the requirements of existence, uniqueness and stability of solutions.

This thesis discusses a variety of inverse problems that arise across corrosion engineering. Some example classes are provided below:

- **Inverse Cauchy problems**

  This problem involves prescribing boundary conditions on both the solution and its normal derivative on a portion of the boundary, while the remaining part of the boundary has no prescribed conditions. The goal of the Cauchy problem is to determine the missing solution and normal derivative on this remaining, inaccessible part of the boundary. Such situations often arise in practical applications where measurements cannot be obtained on certain hostile or inaccessible regions of the boundary, yet the values on those regions are needed for the analysis or modeling of the system. In these cases, the missing boundary information can be sought using additional measurements taken on the accessible parts of the boundary, allowing the Cauchy problem to be solved and the unknown boundary values to be determined. This problem has been extensively studied, see, for example, [60, 75].

- **Inverse parameter identification problems**

  Inverse problems can also be formulated as parameter identification problems, where the aim is to determine the values of specific model parameters that control the behavior of a system, based on experimental observations or measurements. A

common example of an inverse parameter identification problem is determining the unknown thermal conductivity of a material by analyzing temperature and heat flux measurements taken at the boundary [13, 14].

- **Inverse source identification problems**

  Another class of inverse problems involves the identification of unknown sources that generate certain observed effects. These source identification problems often arise in various branches of science and engineering, such as locating the origins of seismic activity, tracing the sources of environmental pollutants, or determining the location and characteristics of tumors in medical imaging applications. The goal in these inverse problems is to infer the unknown source based on the observed data, which can provide valuable insights for scientific understanding and practical applications.

- **Inverse geometry problems**

  Inverse geometric problems focus on reconstructing the shape or geometry of an object by analyzing indirect measurements or observations. These problems model the detection of various types of defects, such as obstacles, cavities, inclusions, flaws, faults, voids and cracks. In practical applications, inverse geometric problems are explored through a range of imaging and tomography techniques, including X-ray computed tomography, ultrasound imaging, electrical impedance tomography [70], and magnetic resonance imaging [58].

1.4 Numerical methods

The numerical solution to initial/boundary value problems is obtained using techniques such as the finite element method (FEM), the finite difference method (FDM) or the boundary element method (BEM). While traditional domain-based methods like the FEM or FDM are widely used for direct problems, they require the discretisation of the entire solution domain, which becomes computationally expensive, especially for inverse problems involving unknown boundaries. Therefore, the use of the BEM in this thesis is motivated by several key advantages:

- Dimensionality reduction: The BEM reduces the dimensionality of the problem by one, requiring discretization only of the boundary $\partial\Omega$ rather than the entire domain Ω.
- Direct unknown targeting: In corrosion problems, the primary unknowns— such as the corrosion coefficient λ and the boundary shape $r(\theta)$— reside specifically on the boundary. The BEM targets these values directly without unnecessary internal calculations.
- Mathematical precision: It exactly satisfies the governing Laplace equation within the interior of the domain, providing higher accuracy for potential field problems.
- Irregular geometries: BEM is exceptionally flexible in handling the evolving and irregular shapes characteristics of corroding surfaces.

By leveraging these boundary-focused advantages, this research establishes the BEM as a robust and efficient numerical framework for identifying hidden damage in inaccessible engineering infrastructure.

1.5 Regularization methods

Consider a system of M linear equations in N unknowns, written in matrix form as

$$A\underline{X} = \underline{b}, \tag{1.11}$$

where A is the $M \times N$ coefficient matrix, $\underline{X}$ is the $N \times 1$ column vector of unknowns, and $\underline{b}$ is the $M \times 1$ right-hand-side column vector.

Several numerical methods exist to solve (1.11). We outline two common approaches: the direct inversion method for square systems and the least-squares method for overdetermined systems [75].

When $M = N$ and the coefficient matrix A is nonsingular, the exact solution of (1.11) is given directly by

$$\underline{X} = A^{-1}\underline{b}. \tag{1.12}$$

However, matrix inversion can be numerically unstable if A is ill-conditioned (i.e., has

a large condition number). In such cases, even small noise in the entries of A or $\underline{b}$ can amplify errors, leading to unbounded or highly oscillatory solutions.

When $M > N$, an exact solution generally does not exist, so the least-squares method seeks the $\underline{X}$ that minimizes the squared Euclidean norm of the residual:

$$T(\underline{X}) := \|A\underline{X} - \underline{b}\|^2. \tag{1.13}$$

The minimizer is found by first computing the gradient of T with respect to $\underline{X}$, then setting the gradient to zero,

$$A^\top(A\underline{X} - \underline{b}) = \underline{0},$$

which implies the normal system of equations

$$A^\top A\underline{X} = A^\top \underline{b}. \tag{1.14}$$

Note that although A is rectangular ($M \times N$ with typically $M > N$), the normal matrix $A^\top A$ is square ($N \times N$) and symmetric positive semi-definite. The system of equations (1.14) is solved to obtain the approximate solution in the least-squares sense given by

$$\underline{X} = (A^\top A)^{-1} A^\top \underline{b}. \tag{1.15}$$

Regularization is the mathematical process utilized to stabilize ill-conditioned systems of equations by introducing additional constraints or "prior information" into the optimization functional. Statistically, regularization addresses the bias-variance dilemma [53]. While an unregularized estimator (like ordinary least-squares) is unbiased, it suffers from extreme variance when the system is ill-conditioned; regularization introduces a controlled amount of bias to significantly reduce this variance, resulting in a more stable and generalized solution. A wide range of regularization strategies exist in the literature to combat overfitting and instability:

- Early Stopping: Extensively used in iterative learning and neural networks, this method treats the number of iterations or "stopping time" as a regularization pa-

rameter [53]. By stopping the optimization before full convergence, the algorithm captures the dominant low-frequency physical features of the solution while ignoring the high-frequency noise that typically fits later in the process.

- Lasso (Least Absolute Shrinkage and Selection Operator): This method utilizes an L^1 penalty (the sum of the absolute values of the coefficients) added to the residual sum of squares [94]. Due to the nature of the L^1 norm, the Lasso performs both continuous shrinkage and automatic variable selection simultaneously, often producing "sparse" models where negligible parameters are zeroed out [103].
- Elastic Net: This method is a generalization of the Lasso, utilising a penalty that is a convex combination of the L^1 (Lasso) and L^2 (Ridge) penalties. This hybrid approach is particularly effective when dealing with highly correlated predictors, as it maintains the sparsity of the Lasso while encouraging a "grouping effect" where correlated variables are included or excluded from the model together [103].

In this thesis, the choice of specific regularization methods is motivated by the physical requirements of the corrosion model. Truncated singular value decomposition (TSVD) is employed for linear Cauchy data recovery because it allows for a data-driven truncation of the singular value spectrum to filter noise, whilst the Tikhonov regularization is utilized for material and shape identification to enforce necessary spatial smoothness or geometric stability.

1.5.1 Tikhonov's regularisation method

The Tikhonov regularised solution of the system of equations (1.11) is given by

$$\underline{X}_{\mu} = \arg \min_{\underline{X} \in \mathbb{R}^N} \left[\|A\underline{X} - \underline{b}\|^2 + \mu \|R_k \underline{X}\|^2 \right], \tag{1.16}$$

where $\mu > 0$ is the regularization parameter to be prescribed. This reduces the instabilities produced by the ordinary least-squares method when $\mu = 0$. In (1.16), R_k represents the regularizing derivative matrix of order k, where $k = 0, 1, 2, ...$, chosen depending on

the desired smoothness of the solution, e.g.,

$$R_0 = I_N \in \mathbb{R}^{N\times N},$$

$$R_1 = \begin{pmatrix} 1 & -1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & -1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & -1 \end{pmatrix} \in \mathbb{R}^{(N-1)\times N},$$

$$R_2 = \begin{pmatrix} 1 & -2 & 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & -2 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & -2 & 1 \end{pmatrix} \in \mathbb{R}^{(N-2)\times N}.$$

Minimizing (1.16) yields explicitly, the Tikhonov regularised solution [25]

$$\underline{X}_\mu = (A^{\mathrm{T}}A + \mu R_k^{\mathrm{T}} R_k)^{-1} A^{\mathrm{T}} \underline{b}. \tag{1.17}$$

One of the criteria for choosing the regularization parameter μ is the L-curve criterion [75], which is based on plotting, on a log-log graph, the residual norm $\|A\underline{X}_\mu - \underline{b}\|$ versus the solution norm $\|R_k X_\mu\|$ for various values of $\mu > 0$ and choosing μ at the corner of the resulting L-curve.

1.5.2 Singular value decomposition (SVD)

The singular value decomposition (SVD) is a fundamental method in linear algebra widely employed in solving linear systems and matrix approximations. Singular values are non-negative real numbers that quantify the "strength" or "significance" of the principal components of a linear transformation, representing the scaling factors along the principal axes [54]. They are typically obtained as the square roots of the eigenvalues of $A^{\mathrm{T}}A$ (or AA^{T}). Given a matrix $A \in \mathbb{R}^{M\times N}$, this can be decomposed as [55]:

$$A = U\Sigma V^{\mathrm{T}}, \tag{1.18}$$

where $U \in \mathbb{R}^{M\times M}$ and $V \in \mathbb{R}^{N\times N}$ are orthogonal matrices containing the eigenvectors of AA^{T} and $A^{\mathrm{T}}A$, respectively, and $\Sigma \in \mathbb{R}^{M\times N}$ is a diagonal matrix with singular values

$$\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_n \geq 0.$$

By introducing (1.18) into the system of equations (1.11), one obtains:

$$||A\underline{X} - \underline{b}|| = ||U\Sigma V^{\mathrm{T}}\underline{X} - \underline{b}|| = ||\Sigma\underline{c} - \underline{d}||, \tag{1.19}$$

where $\underline{c} = V^{\mathrm{T}}\underline{X}$ and $\underline{d} = U^{\mathrm{T}}\underline{b}$. Define Σ^{+} as the diagonal matrix whose diagonal entries are the inverse singular values

$$\sigma_i^{+} = \begin{cases} \sigma_i^{-1}, & \text{if } \sigma_i \neq 0, \\ 0, & \text{otherwise.} \end{cases}$$

The solution to the system of equations (1.11) is then given by:

$$\underline{\tilde{X}} = A^{+}\underline{b}, \tag{1.20}$$

where A^{+} denotes the Moore–Penrose pseudo-inverse of A, which generalizes the matrix inverse to singular and non-square matrices and can be expressed via SVD as $A^{+} = V\Sigma^{+}U^{\mathrm{T}}$ [54]. Equation (1.20) can be written in the form

$$\underline{X} = \sum_{i=1}^{N} \sigma_i^{+}(\underline{u}_i^{\mathrm{T}} \cdot \underline{b})\,\underline{v}_i, \tag{1.21}$$

where u_i and v_i represent the column vectors of the matrices U and V, respectively .

1.5.3 Truncated Singular Value Decomposition (TSVD)

Although the SVD provides a stable framework for solving linear systems, in ill-posed problems the smallest singular values amplify noise, leading to unstable solutions. To overcome this, the truncated SVD (TSVD) method eliminates the influence of the smallest

singular values by truncating the expansion (1.21) at a chosen parameter j, [75], to yield

$$\underline{X}_j = \sum_{i=1}^{j} \sigma_i^{-1} (\underline{u}_i^{\mathrm{T}} \cdot \underline{b})\, \underline{v}_i. \tag{1.22}$$

where j is the truncation parameter.

By neglecting small singular values, TSVD stabilizes the solution, mitigates the effect of noise and provides a low-rank approximation of the matrix A, making it especially effective in regularization of ill-posed inverse problems.

1.6 Aim and objectives of thesis

The primary aim of this research is to develop and validate a unified computational framework using the BEM for the simultaneous identification of unknown boundary geometries and spaced-dependent material characteristics in inaccessible corroded infrastructure.

The specific objectives to achieve this unified framework are:

- Platform Validation: To establish the BEM as a superior numerical platform over the domain discretisation methods for solving both direct and inverse problems for the Laplace's equation and compare its performance against meshless methods.
- Data Recovery: To solve the inverse Cauchy problem for reconstructing missing potential and flux distributions on inaccessible surfaces.
- Property Identification: To develop a methodology for identifying the space-dependent corrosion coefficient on known geometries using the Tikhonov regularization method.
- Geometric Reconstruction: To implement a data fusion technique for the unique reconstruction of unknown corroded boundary shapes.
- Joint Identification (The Unified Framework): To integrate the geometric and material identification sub-problems into a single, stable optimization framework using model reduction via Fourier trigonometric polynomials to overcome the ill-posedness of the joint inverse problem.

1.7 Structure of thesis

This thesis investigates the mathematical and computational methods for modeling corrosion in buried pipelines, specifically focusing on both direct and inverse problems. The main motivation for this research is the need to identify and characterize corrosion damage in portions of pipelines that are physically inaccessible to direct measurement. This is crucial for maintaining the structural integrity and operational safety of critical infrastructures, such as those in oil, water and gas distribution networks. The thesis comprises eight chapters, structured as follows: Chapter 2 examines the Direct Problem for modeling corrosion within a doubly-connected annular domain, which represents a buried cylindrical pipe. Given the symmetry of this configuration, the analysis focuses on only a quarter of the geometry, where the inner boundary signifies the corroded pipe surface undergoing electrochemical reactions. The governing Laplace equation is numerically solved using both the BEM and the meshless Method of Fundamental Solutions (MFS). To model the corrosion process, both linear and nonlinear Robin-type boundary conditions are applied to the corroded boundary, while a prescribed current flux is imposed on the accessible outer boundary through strategically placed electrodes. This foundational formulation supports the inverse problems explored in Chapters 4–7. Chapter 3 expands this analysis to an annular domain problem where a rigid, perfectly conducting inclusion forms the inner core. Here, the Dirichlet boundary value problem for the Laplace equation is solved using the BEM, with the potential assuming a constant value on the inner boundary. If this constant is unknown, an additional physical constraint that the integral of the current flux over the outer boundary must be zero, is enforced to guarantee solution's uniqueness. This chapter thus offers a fundamental understanding for addressing perfectly conducting boundaries within corrosion contexts. Chapter 4 investigates the Cauchy inverse problem for the Laplace equation, addressing the realistic scenario where direct measurements on the corroded boundary are not feasible. The objective is to determine the unknown potential and flux distributions on this inaccessible corroded boundary, utilizing over-prescribed Cauchy data, which includes both potential and flux measurements

on the accessible boundary. Discretization using the BEM results in an overdetermined system of linear equations that is highly ill-conditioned, a characteristic inherent to the ill-posed nature of this problem. The TSVD is employed as a regularization technique to stabilize the numerical solution, particularly in the presence of measurement noise. Chapter 5 shifts its focus to determining the space-dependent corrosion coefficient on the corroded boundary, based on potential measurements taken elsewhere on the accessible portion of the boundary. This inverse problem operates under the assumption that the corroded boundary's geometry is known, but the corrosion coefficient, which governs the electrochemical reaction rate, remains unknown. The BEM discretization yields a nonlinear inverse problem, which is then reformulated as a least-squares minimization problem incorporating Tikhonov regularization to ensure solution's stability. Chapter 6 addresses the geometric inverse problem, aiming to reconstruct the unknown shape of the corroded boundary when the corrosion coefficient is a known parameter. The corroded boundary is parameterized using polar coordinates, and an iterative optimization algorithm, grounded in nonlinear least-squares minimization, is utilized to progressively refine the boundary geometry. To ensure the uniqueness and stability of this reconstruction, at least two linearly independent pairs of Cauchy data from the accessible boundary are required. The most complicate inverse problem is presented in Chapter 7, which involves the simultaneous reconstruction of both the space-dependent corrosion coefficient and the unknown shape of the corroded boundary. An iterative optimization framework, integrating shape derivatives and sensitivity analysis, is developed to concurrently update both parameters using over-prescribed potential measurements from the accessible boundary. To mitigate the severe ill-posedness associated with the full-dimensional formulation, model reduction through trigonometric Fourier series parametrization is applied, thereby decreasing the number of unknowns and enhancing computational efficiency. Finally, Chapter 8 synthesizes the main findings from all preceding chapters and presents the overall conclusions of this thesis. It discusses the practical implications of the developed methodologies for non-destructive corrosion assessment and proposes directions for future research.

Chapter 2

Direct problem

The focus of this chapter is to solve the direct problem (1.3) for determining the potential distribution u in the region of interest Ω using the Boundary Element Method (BEM). The BEM reduces the dimensionality of the problem by one by requiring only the boundary $\partial\Omega$ to be discretised. Despite the absence of an exact solution for our problem (1.3), we will compare the BEM numerical results with those obtained using the method of fundamental solution (MFS).

A brief outline of this chapter is as follows. In Section 2.1, we introduce the mathematical formulation of the direct problem. The BEM is explained in Section 2.2, while Section 2.3 describes the MFS. In Section 2.4, we compare some numerical results gathered by solving the linear boundary condition problem with those obtained by solving the nonlinear boundary condition problem for various (constant, variable, positive or negative) corrosion coefficient λ. In Section 2.5, we present the conclusions of the chapter.

2.1 Mathematical formulation of the direct problem

Recapping (1.3), we consider the electrical potential u satisfying the Laplace equation in a two-dimensional bounded homogeneous electrolyte domain Ω. In Figure 2.1, the boundary Γ_0 is the first quarter of the unit circle, whilst the remaining boundary part consists of the horizontal and vertical portions $\Gamma_1, \Gamma_2, \Gamma_3$ and Γ_4. This geometry is in fact a symmetric reduction of the full circular pipe of unit radius centered at the origin that

is concealed inside the square duct $(-2,2) \times (-2,2)$. Then, due to symmetry, the flux $g = 0$ on $\Gamma_1 \cup \Gamma_3$. Also, by assuming that the top wall is adiabatic we have that $g = 0$ on Γ_3.

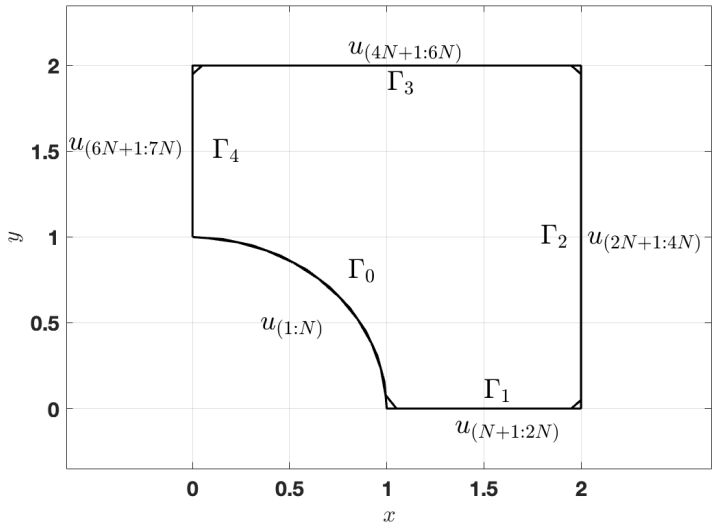

Figure 2.1: The solution domain Ω, with primary unknowns computed over its boundary $\partial\Omega = \bigcup_{i=0}^{4} \Gamma_i$, using $7N$ constant boundary elements.

The governing equation is given by

$$\Delta u = 0 \qquad \text{in} \quad \Omega, \tag{2.1}$$

subject to the boundary conditions

$$\frac{\partial u}{\partial n} = \lambda f(u) + g \quad \text{on} \quad \Gamma_0, \tag{2.2}$$

$$\frac{\partial u}{\partial n} = g \qquad \text{on} \quad \Gamma = \partial\Omega \setminus \Gamma_0, \tag{2.3}$$

where $\underline{n}$ denotes the outward unit normal vector to the boundary $\partial\Omega$, g is given flux defined on the boundary $\partial\Omega$, λ represents the coefficient of corrosion (due to the chemical transfer at the corroded boundary Γ_0) and $f(u)$ is a nonlinear law of corrosion, usually given by the Butler-Volmer law

$$f(u) = e^{\alpha u} - e^{-(1-\alpha)u}, \tag{2.4}$$

where $\alpha \in (0,1)$ is a symmetry parameter. In case $\alpha = \frac{1}{2}$, expression (2.4) yields

$$f(u) = 2\sinh\left(\frac{u}{2}\right). \tag{2.5}$$

Equation (2.5) may be linearised as

$$f(u) \approx u + .., \tag{2.6}$$

for $|u| \ll 1$, in which case the boundary condition (2.2) becomes of Robin type given by

$$\frac{\partial u}{\partial n} = \lambda u + g \quad \text{on} \quad \Gamma_0. \tag{2.7}$$

In the next two sections we describe two numerical methods that can be employed for solving the direct problem (2.1)–(2.3)

2.2 The Boundary Element Method (BEM)

Boundary elements are typically connected with direct formulations in which the unknowns of the problem are physical variables such as potentials or fluxes, which are more appealing to users than indirect formulations, which were commonly employed before 1978 and contain concepts such as sources or dipoles. In direct problems, one aims to ascertain the answer to a boundary value problem governed by a partial differential equation in a domain subject to predetermined boundary conditions.

The BEM may be considered to have evolved and become a powerful instrument for investigating engineering problems [68]. In addition, the BEM has been effectively used for a wide range of inverse problems [60, 92, 93].

Since BEM reduces the number of unknowns, it has an advantage over traditional domain discretisation techniques [74]. Furthermore, the BEM can be used to accurately model irregular geometries and non-uniform boundary conditions, which makes it a powerful tool in various engineering and scientific applications in electromagnetics, heat transfer and acoustics. BEM can also be combined with optimization methods,

making it possible to solve nonlinear and ill-posed inverse problems.

In this section, the Laplace equation (2.1) subjected to the boundary conditions (2.2) and (2.3) is being tackled in two dimensions using the BEM. First, the Laplace equation (2.1) is transformed into the boundary integral equation, [22, 74, 75]:

$$\eta(\underline{p})u(\underline{p}) = \int_{\partial\Omega} \Big[G(\underline{p},\underline{p}')\frac{\partial u}{\partial n}(\underline{p}') - u(\underline{p}')\frac{\partial G}{\partial n}(\underline{p},\underline{p}')\Big]ds, \quad \underline{p} \in \overline{\Omega}, \tag{2.8}$$

where $\eta(\underline{p}) = 1$ if $\underline{p} \in \Omega$ and $\eta(\underline{p}) = 0.5$ if $\underline{p} \in \partial\Omega$ (smooth), and G is the fundamental solution of the two-dimensional Laplace equation given by

$$G(\underline{p},\underline{p}') = -\frac{1}{2\pi}\ln|\underline{p} - \underline{p}'|. \tag{2.9}$$

Then, applying (2.8) at $\underline{p} \in \partial\Omega = \Gamma_0 \cup \Gamma_1 \cup \Gamma_2 \cup \Gamma_3 \cup \Gamma_4$ and using the boundary conditions (2.2) and (2.3) results in

$$\frac{1}{2}u(\underline{p}) = \int_{\Gamma_0} G(\underline{p},\underline{p}')\Big(\lambda(\underline{p}')f(u(\underline{p}')) + g(\underline{p}')\Big)\, ds(\underline{p}') + \int_{\partial\Omega\setminus\Gamma_0} G(\underline{p},\underline{p}')\, g(\underline{p}')\, ds(\underline{p}') - \int_{\partial\Omega} u(\underline{p}')\frac{\partial G}{\partial n(\underline{p}')}(\underline{p},\underline{p}')ds(\underline{p}'), \quad \underline{p} \in \partial\Omega. \tag{2.10}$$

We discretise anticlockwise the boundary $\partial\Omega = \Gamma_0 \cup \Gamma_1 \cup \Gamma_2 \cup \Gamma_3 \cup \Gamma_4$ of the solution domain Ω into a series of $M = 7N$ boundary elements, see Figure 2.1, which are straight line segments $S_j = [\underline{p}_{j-1}, \underline{p}_j]$ for $j = \overline{1, M}$, where $\underline{p}_M = \underline{p}_0 = (0, 1)$. Over each small boundary element S_j, the unknown Dirichlet data u is assumed constant and to take the value at its midpoint (boundary element node), $\underline{\tilde{p}}_j = (\underline{p}_j + \underline{p}_{j-1})/2$ for $j = \overline{1, M}$, namely,

$$u(\underline{p}) \approx u(\underline{\tilde{p}}_j) =: u_j, \quad \underline{p} \in S_j, \quad j = \overline{1, M}. \tag{2.11}$$

We also approximate the Neumann data g piece-wisely as

$$g(\underline{p}) \approx g(\underline{\tilde{p}}_j) =: g_j, \quad \underline{p} \in S_j, \quad j = \overline{1, M}, \tag{2.12}$$

and the corrosion coefficient

$$\lambda(\underline{p}) \approx \lambda(\underline{\tilde{p}}_j) =: \lambda_j, \quad \underline{p} \in S_j, \quad j = \overline{1, M}. \tag{2.13}$$

Denote

$$A_j(\underline{p}) := \int_{S_j} G(\underline{p}, \underline{p}')ds(\underline{p}'), \quad B_j(\underline{p}) := \int_{S_j} \frac{\partial G(\underline{p}, \underline{p}')}{\partial n(\underline{p}')}ds(\underline{p}'), \quad \underline{p} \in \overline{\Omega}, \quad j = \overline{1, M}. \tag{2.14}$$

The approximation of the boundary $\partial\Omega$ by straight lines enables us to evaluate analytically the integrals $A_j(\underline{p})$ and $B_j(\underline{p})$ in (2.14) for every $\underline{p} \in \overline{\Omega}$ and their expressions are given by [75]:

$$A_j(\underline{p}) = A(\underline{p}, \underline{p}_{j-1}, \underline{p}_j) = -\frac{1}{2\pi} \begin{cases} h(\ln(h) - 1) & \text{if } ab = 0, \\ a\cos(\beta)\ln(a/b) - h(1 - \ln(b)) + a\,\psi\sin(\beta) & \text{if } ab \neq 0, \end{cases} \tag{2.15}$$

$$B_j(\underline{p}) = B(\underline{p}, \underline{p}_{j-1}, \underline{p}_j) = \frac{1}{2\pi} \begin{cases} 0 & \text{if } ab = 0 \text{ or } \underline{p} \in \overline{\underline{p}_{j-1}, \underline{p}_j}, \\ \psi\ \mathrm{sign}(\alpha_{j-1}(\underline{p}) - \alpha_j(\underline{p})) & \text{if } y \in [y_{j-1}, y_j], \\ \psi\ \mathrm{sign}(\alpha_j(\underline{p}) - \alpha_{j-1}(\underline{p})) & \text{otherwise}, \end{cases} \tag{2.16}$$

where sign is the signum function, $a = |\underline{p} - \underline{p}_{j-1}|$, $b = |\underline{p} - \underline{p}_j|$, $h = |\underline{p}_j - \underline{p}_{j-1}|$, $\alpha_{j-1}(\underline{p}) \in [0, \pi]$ and $\alpha_j(\underline{p}) \in [0, \pi]$ are the angles between the positive x-axis and the straight lines $\overline{\underline{p}\underline{p}_{j-1}}$ and $\overline{\underline{p}\underline{p}_j}$ in the upper half plane, respectively, and the angles $\psi \in [0, \pi]$ and $\beta \in [0, \pi]$ are given by

$$\psi = \arccos\left(\frac{a^2 + b^2 - h^2}{2ab}\right), \quad \beta = \arccos\left(\frac{a^2 + h^2 - b^2}{2ah}\right), \tag{2.17}$$

see Figure 2.2. Collocating (2.10) at the boundary element nodes $\underline{p} = \underline{\tilde{p}}_i$ for $i = \overline{1, M}$,

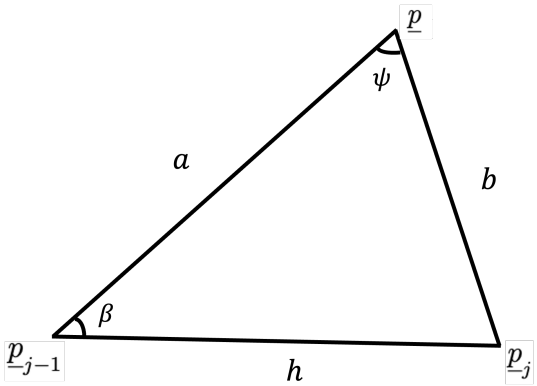

Figure 2.2: Triangle involved in calculating the BEM coefficients $A_j(\underline{p})$ and $B_j(\underline{p})$.

and using (2.11) and (2.12), we obtain

$$\sum_{j=1}^{M} A_{ij} g_j + \sum_{j=1}^{N} A_{ij} \lambda_j f(u_j) + \sum_{j=1}^{M} B_{ij} u_j = 0, \quad i = \overline{1, M}, \tag{2.18}$$

where

$$A_{ij} = A_j(\underline{\tilde{p}}_i), \quad B_{ij} = -B_j(\underline{\tilde{p}}_i) - \frac{1}{2}\delta_{ij} \quad \text{for} \quad i = \overline{1, M}, \quad M = 7N, \tag{2.19}$$

and δ_{ij} is the Kronecker delta symbol.

The system of nonlinear algebraic equations (2.18) is solved using the MATLAB command `fsolve` to determine the Dirichlet values u_j for $j = \overline{1, M}$. In case the boundary law (2.5) linearizes as (2.6) then the resulting system of linear algebraic equations

$$\sum_{j=1}^{N} (A_{ij}\lambda_j + B_{ij}) u_j + \sum_{j=N+1}^{M} A_{ij} u_j = -\sum_{j=1}^{M} A_{ij}\, g_j, \quad i = \overline{1, M}, \tag{2.20}$$

is solved using the LU decomposition method.

2.3 The Method of Fundamental Solution (MFS)

A meshless approach, namely the method of fundamental solution (MFS), see [64], may also be employed to solve the direct problem given by Eqs.(2.1)–(2.3).

The essential idea behind the MFS is to approximate the solution using a linear combination of fundamental solutions of the governing equation with reference to source locations situated outside the solution domain, as explained in [20]. The MFS differs from the commonly used BEM based on boundary integral equations in that the source points are positioned outside the solution domain. There appear to be a few similarities in the literature between the BEM and the MFS, as mentioned in [3, 23]. In the MFS, we seek a solution to (2.1) as a linear combination of non-singular fundamental solutions [35, 64, 66],

$$u(\underline{p}) = \sum_{j=1}^{M} c_j G(\underline{p}, \underline{\xi}_j), \quad \underline{p} \in \overline{\Omega}, \tag{2.21}$$

where $(\underline{\xi}_j)_{j=\overline{1,M}}$ are so-called source points (or 'singularities') selected outside the domain $\overline{\Omega}$ and $(c_j)_{j=\overline{1,M}}$ are unknown coefficients to be determined by imposing the boundary conditions (2.2) and (2.3). Since $\underline{\xi}_j \notin \overline{\Omega}$ for $j = \overline{1, M}$, the right-hand side of (2.21) represents a linear combination of harmonic functions and hence it is a harmonic function which satisfies the Laplace's equation (2.1) in Ω. Based on (2.21), collocating (2.2) and (2.3) at the boundary points $(\tilde{p}_i)_{i=\overline{1,M}}$ we obtain

$$\sum_{j=1}^{M} c_j \frac{\partial G}{\partial n(\underline{p})}(\underline{\tilde{p}}_i, \underline{\xi}_j) = \frac{\partial u}{\partial n}(\tilde{p}_i) = g_i + \lambda_i f\Big(\sum_{j=1}^{M} c_j G(\underline{\tilde{p}}_i, \underline{\xi}_j)\Big), \quad i = \overline{1, N}, \tag{2.22}$$

$$\sum_{j=1}^{M} c_j \frac{\partial G}{\partial n(\underline{p})}(\underline{\tilde{p}}_i, \underline{\xi}_j) = \frac{\partial u}{\partial n}(\tilde{p}_i) = g_i, \quad i = \overline{(N+1), M}, \tag{2.23}$$

where

$$\frac{\partial G}{\partial n(\underline{p})}(\underline{\tilde{p}}_i, \underline{\xi}_j) = \nabla_{\underline{p}} G(\underline{\tilde{p}}_i, \underline{\xi}_j) \cdot \underline{n}(\underline{\tilde{p}}_i) = \frac{\partial G}{\partial x}(\underline{\tilde{p}}_i, \underline{\xi}_j) n_x(\underline{\tilde{p}}_i) + \frac{\partial G}{\partial y}(\underline{\tilde{p}}_i, \underline{\xi}_j) n_y(\underline{\tilde{p}}_i) \tag{2.24}$$

and

$$\frac{\partial G}{\partial x}(\underline{\tilde{p}}_i, \underline{\xi}_j) = -\frac{1}{2\pi} \frac{(\tilde{x}_i - \zeta_j)}{|\underline{\tilde{p}}_i - \underline{\xi}_j|^2}, \quad \frac{\partial G}{\partial y}(\underline{\tilde{p}}_i, \underline{\xi}_j) = -\frac{1}{2\pi} \frac{(\tilde{y}_i - \eta_j)}{|\underline{\tilde{p}}_i - \underline{\xi}_j|^2}, \tag{2.25}$$

where $\underline{\tilde{p}}_i = (\tilde{x}_i, \tilde{y}_i)$, $\underline{\xi}_j = (\zeta_j, \eta_j)$ and $\underline{n} = (n_x, n_y)$. In case of the linear boundary condition

(2.6), equations (2.22) simplify as

$$\sum_{j=1}^{M} c_j \left[\frac{\partial G}{\partial n(\underline{p})}(\underline{\tilde{p}}_i, \underline{\xi}_j) - \lambda_i \, G(\underline{\tilde{p}}_i, \underline{\xi}_j) \right] = g_i, \quad i = \overline{1, N}. \tag{2.26}$$

2.4 Numerical results for the direct problem

On the boundary Γ_0, we define the points $\underline{p}_j = (r(\theta_j)\cos(\theta_j), r(\theta_j)\sin(\theta_j))$ for $j = \overline{0, N}$, with $\theta_j = \frac{\pi(N-j)}{2N}$, where $r(\theta_0) = r(\frac{\pi}{2}) = 1$ and $r(\theta_N) = r(0) = 1$, representing a quarter-circle arc of unit radius.

On the vertical wall Γ_2, we impose a current flux over an electrode of length 2ϵ centered at $(2, \frac{1}{3})$ which is drawn out over another electrode of length 2ϵ centered at $(2, \frac{2}{3})$, where $\epsilon > 0$ is a small number, typically $\epsilon = 0.1$. Therefore, on Γ_2, we initially impose the discontinuous flux

$$g_1(2, y) = \begin{cases} 0 & \text{if} \quad 0 \le y < \frac{1}{3} - \epsilon, \\ \frac{1}{2\epsilon} & \text{if} \quad \frac{1}{3} - \epsilon \le y \le \frac{1}{3} + \epsilon, \\ 0 & \text{if} \quad \frac{1}{3} + \epsilon < y < \frac{2}{3} - \epsilon, \\ \frac{-1}{2\epsilon} & \text{if} \quad \frac{2}{3} - \epsilon \le y \le \frac{2}{3} + \epsilon, \\ 0 & \text{if} \quad \frac{2}{3} + \epsilon < y \le 2. \end{cases} \tag{2.27}$$

Expression (2.27) signifies that an electric current is injected through the electrode centred at $y = \frac{1}{3}$ of length 2ϵ and drawn out through the electrode centred at $y = \frac{2}{3}$ also of length 2ϵ.

We shall also investigate imposing the continuous flux

$$g_2(2, y) = y(2 - y), \quad y \in [0, 2], \tag{2.28}$$

prescribed on Γ_2. Both fluxes (2.27) and (2.28) satisfy lateral continuity as we approach the corners (2,0) and (2,2) from either sides. We also take, for simplicity, $g = 0$ on Γ_0 in

which case the boundary condition (2.2) becomes

$$\frac{\partial u}{\partial n} = \lambda f(u) \quad \text{on} \quad \Gamma_0. \tag{2.29}$$

Although the exact solution is not available to compare with the numerical results to determine their accuracy and the order of convergence, we have employed both the BEM and the MFS to compute and compare the numerical solutions. Eight examples explore the cases $\lambda(\theta) = \pm 1$ and $\lambda(\theta) = \pm \sin(\theta)$, where $\theta \in \left[0, \frac{\pi}{2}\right]$, for the fluxes (2.27) and (2.28), testing the convergence as N increases.

We have employed the BEM to solve the direct problem for the geometry depicted in Figure 2.1 for different numbers of boundary elements $M = 7N$. The boundary $\partial\Omega$ is uniformly discretised anti-clockwise starting from the point (0,1) with N boundary elements on each of the boundaries Γ_0, Γ_1 and Γ_4 and $2N$ boundary elements on each of the boundaries Γ_2 and Γ_3.

In the MFS, the M source points $(\underline{\tilde{\xi}}_j)_{j=\overline{1,M}}$ are placed on a dilated boundary at a distance $\delta > 0$ from $\partial\Omega$, as shown in Figure 2.3. There are many ways to choose a suitable value of $\delta > 0$, as discussed in various references [33, 34] on the MFS subject, but herein we only mention that the value of $\delta > 0$ depends on how far the solution of the problem (2.1)–(2.3) can be analytically continued in the exterior of Ω and, in our numerical simulations, we take a small value of δ such as 0.2. We also note that Chen et al. [32] emphasized that the ill-conditioning of the MFS grows as the distance δ between the source points $(\underline{\xi}_j)_{j=1,M}$ and the boundary increases.

For the linear boundary condition (LBC)

$$\frac{\partial u}{\partial n} = \lambda u \quad \text{on} \quad \Gamma_0, \tag{2.30}$$

we apply the `linsolve` command to obtain the numerical solution of the system of linear algebraic equations (see (2.20))

$$\sum_{j=1}^{N}(A_{ij}\lambda_j + B_{ij})u_j + \sum_{j=N+1}^{M} B_{ij}u_j = -\sum_{j=2N+1}^{4N} A_{ij}g_j, \quad i = \overline{1, M}, \tag{2.31}$$

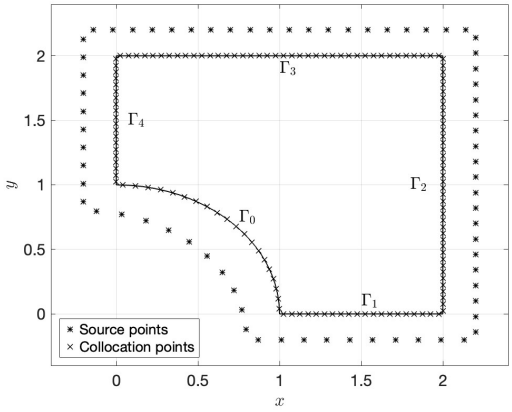

Figure 2.3: The geometry of both the BEM and MFS, with $M = 7N$, indicating the boundary nodes and the source points, when $N = 10$.

in case of the BEM, or (see (2.23))

$$\sum_{j=1}^{M} c_j \frac{\partial G}{\partial n(\underline{p})}(\underline{\tilde{p}}_i, \underline{\xi}_j) = \begin{cases} g_i & \text{for} \quad i = \overline{2N+1, 4N}, \\ 0 & \text{otherwise.} \end{cases} \tag{2.32}$$

and (2.26), in case of the MFS.

For the nonlinear boundary condition (NLBC)

$$\frac{\partial u}{\partial n} = 2\lambda \sinh\left(\frac{u}{2}\right) \quad \text{on} \quad \Gamma_0, \tag{2.33}$$

we use `fsolve` command to obtain the numerical solution of the system of nonlinear algebraic equations (see (2.18))

$$\sum_{j=1}^{M} B_{ij} u_j + 2 \sum_{j=1}^{N} A_{ij}\ \lambda_j \sinh\left(\frac{u_j}{2}\right) = -\sum_{j=2N+1}^{4N} A_{ij}\ g_j, \quad i = \overline{1, M}, \tag{2.34}$$

in case of the BEM or (see (2.22))

$$\sum_{j=1}^{M} c_j \frac{\partial G}{\partial n(\underline{p})}(\underline{\tilde{p}}_i, \underline{\xi}_j) - 2\lambda_i \sinh\left(\frac{1}{2} \sum_{j=1}^{M} c_j G(\underline{\tilde{p}}_i, \underline{\xi}_j)\right) = g_i, \quad i = \overline{1, N}, \tag{2.35}$$

and (2.32), in case of the MFS.

Results are given at selected points in Ω and along Γ_2, with figures visualizing conver-

gence and solution behavior. We focus on comparing LBC and NLBC outcomes in the BEM and MFS, assessing the convergence of the numerical methods.

2.4.1 Examples with the discontinuous current flux (2.27) and $\lambda(\theta) = \pm 1$

2.4.1.1 Example 1: Discontinuous current flux (2.27) and $\lambda(\theta) = 1$

As demonstrated in Table 2.1, increasing N (or M) leads to increased agreement between the BEM and MFS results. For instance, the solutions at the point (1.5,0.5) for the BEM and MFS in the LBC case when $N = 10$ are 0.0993 and 0.0752, respectively, whereas the solutions when $N = 270$ are 0.0841 and 0.0841, respectively. This indicates that as the number of degrees of freedom increases, the results in numerically modelling and predicting the behaviour of the system under consideration become more accurate and reliable. The convergence study in Table 2.1 demonstrates that the BEM and MFS are in very good agreement. To visualize the spatial distribution of the solution, a contour

Table 2.1: The numerical BEM and MFS solutions for the LBC (2.30) and for the NLBC (2.33) on Γ_0, at arbitrary test points in Ω, for various $N \in \{10, 30, 90, 270\}$, when $\lambda(\theta) = 1$ and g on Γ_2 is defined by (2.27).

	N	u_{LBC}		u_{NLBC}	
		BEM	MFS	BEM	MFS
$(1.5, 0.5)$	10	0.0993	0.0752	0.0994	0.0753
	30	0.0840	0.0823	0.0840	0.0847
	90	0.0841	0.0842	0.0841	0.0842
	270	0.0841	0.0841	0.0841	0.0841
$(1.5, 1.5)$	10	-0.1043	-0.0758	-0.1044	-0.0759
	30	-0.0866	-0.0777	-0.0867	-0.0860
	90	-0.0866	-0.0866	-0.0866	-0.0867
	270	-0.0866	-0.0866	-0.0866	-0.0866
$(0.5, 1.5)$	10	-0.0955	-0.0676	-0.0956	-0.0676
	30	-0.0799	-0.0622	-0.0800	-0.0793
	90	-0.0799	-0.0800	-0.0800	-0.0801
	270	-0.0799	-0.0798	-0.0800	-0.0800

plot is shown in Figure 2.4.

The convergence of the solutions with the increase in N (or M) is also illustrated in

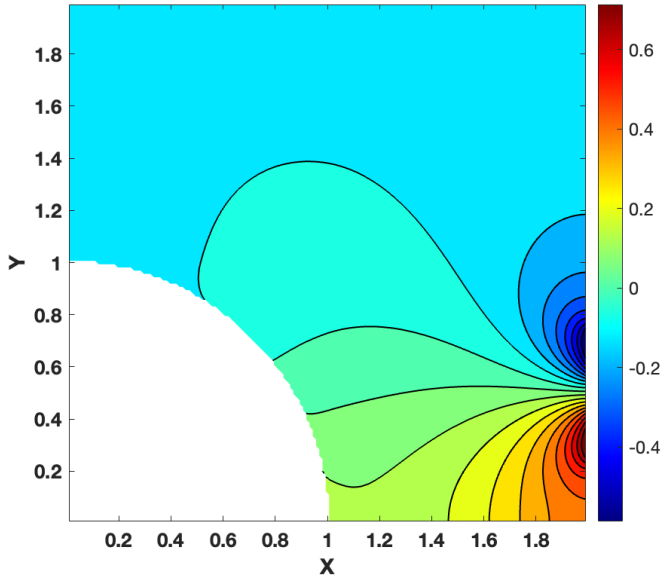

Figure 2.4: Contour plot of the numerical solution computed using the BEM over the domain Ω for the NLBC (2.33) on Γ_0, with $N = 30$, $\lambda = 1$ and g on Γ_2 defined by (2.27).

Figure 2.5. The combination of numerical results and graphical analysis validates the reliability of the BEM and MFS numerical solutions for solving the direct problem.

Additionally, the performance comparison in Figure 2.6 shows that the BEM results for the unspecified boundary data become indistinguishable for $N \in \{30, 90, 270\}$. Furthermore, Table 2.2 (see also Figure 2.6(d)) illustrates the convergence of the BEM numerical solution for $u|_{\Gamma_2} =: h$, as N increases. This will help in simulating the data on Γ_2 that is measured in order to compensate for the lack of knowledge of the missing boundary values (in Chapter 4), unknown coefficient λ (in Chapter 5) and unknown corroded boundary Γ_0 (in Chapter 6) in the inverse problems that will be analysed in the following chapters.

2.4.1.2 Example 2: Discontinuous current flux (2.27) and $\lambda(\theta) = -1$

Figure 2.7 shows that the BEM results for the unspecified boundary data overlap and become visually indistinguishable for $N \in \{30, 90, 270\}$ while Table 2.3 demonstrates that, as N increases, solutions from both the BEM and MFS converge to similar values at all points tested in Ω. It is important to note that in Table 2.3 the result for the NLBC (2.33) of the MFS with $M = 270$, could not be obtained due to extended computational

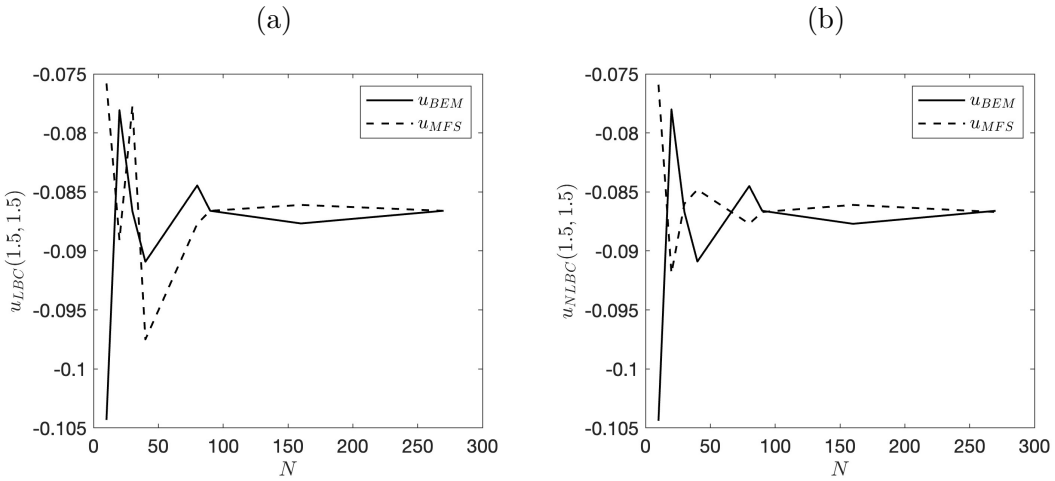

Figure 2.5: Comparing the convergence of the MFS and BEM numerical solutions at the point $(1.5, 1.5)$ for (a) the LBC (2.30) and (b) the NLBC (2.33) on Γ_0, for various $N \in \{10, 20, 30, 40, 80, 90, 160, 270\}$, when $\lambda(\theta) = 1$ and g on Γ_2 is defined by (2.27).

Table 2.2: The y-coordinates of the midpoints on Γ_2 and the BEM numerical solutions for $u|_{\Gamma_2} = h$ obtained with various $N \in \{10, 30, 90, 270\}$ for the NLBC (2.33) on Γ_0 and g on Γ_2 defined by (2.27).

y	$h_{N=10}$	$h_{N=30}$	$h_{N=90}$	$h_{N=270}$
0.05	0.5608	0.4590	0.4591	0.4591
0.15	0.6461	0.5169	0.5170	0.5170
0.25	0.9281	0.7863	0.7864	0.7864
0.35	0.8214	0.8011	0.8012	0.8012
0.45	0.3110	0.3468	0.3469	0.3469
0.55	-0.0697	-0.1445	-0.1444	-0.1444
0.65	-0.5754	-0.5951	-0.5950	-0.5950
0.75	-0.6717	-0.5720	-0.5719	-0.5719
0.85	-0.3716	-0.2881	-0.2880	-0.2880
0.95	-0.2560	-0.2060	-0.2059	-0.2059
1.05	-0.2014	-0.1641	-0.1640	-0.1640
1.15	-0.1698	-0.1393	-0.1392	-0.1392
1.25	-0.1500	-0.1234	-0.1234	-0.1234
1.35	-0.1368	-0.1129	-0.1129	-0.1129
1.45	-0.1279	-0.1058	-0.1058	-0.1058
1.55	-0.1218	-0.1009	-0.1008	-0.1008
1.65	-0.1176	-0.0975	-0.0975	-0.0975
1.75	-0.1148	-0.0953	-0.0952	-0.0952
1.85	-0.1130	-0.0939	-0.0939	-0.0939
1.95	-0.1121	-0.0932	-0.0932	-0.0932

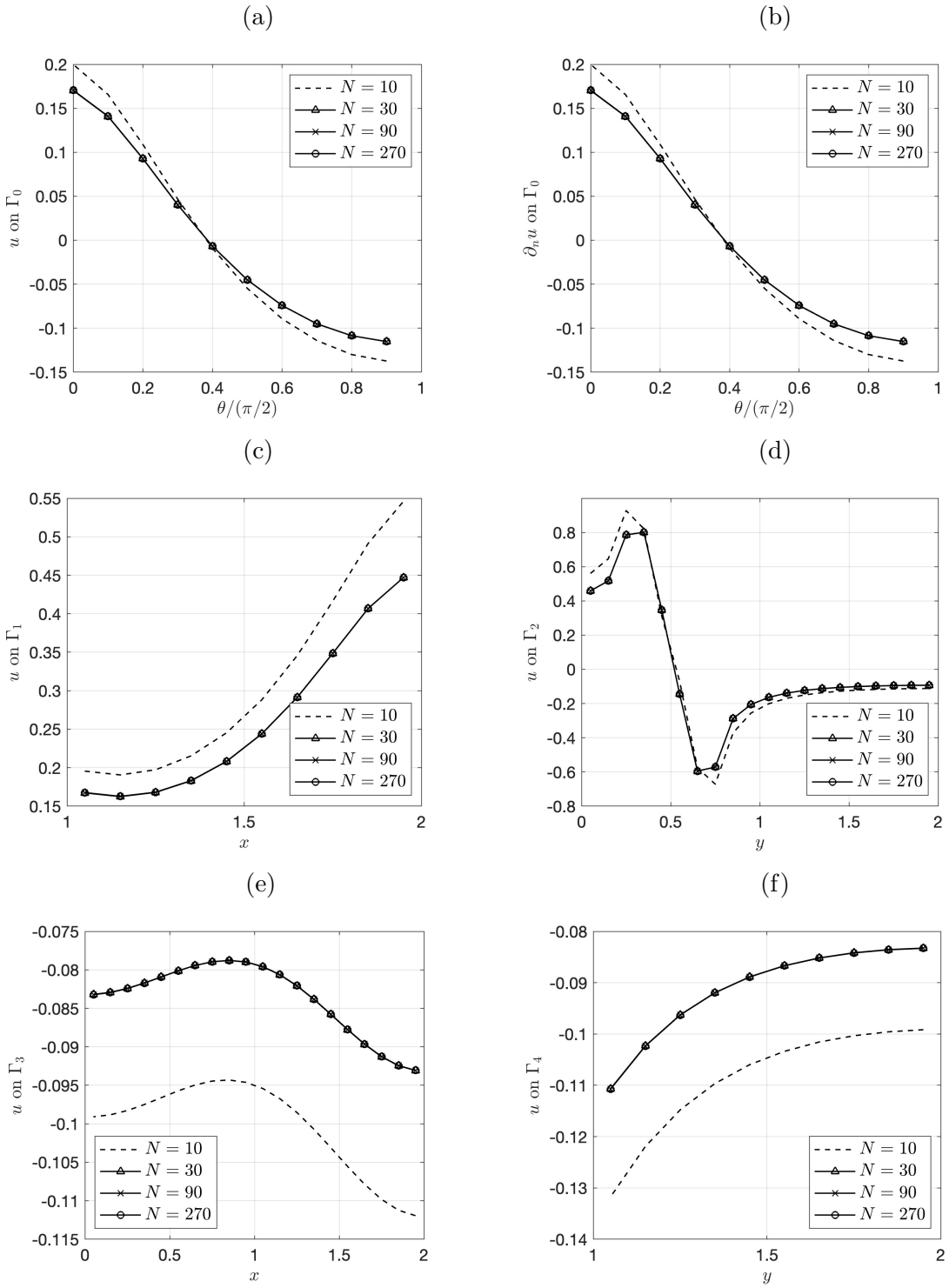

Figure 2.6: Comparison between the BEM values of u over $\partial\Omega$ and $\partial_n u$ over Γ_0 for various $N \in \{10, 30, 90, 270\}$, for the LBC (2.30), when $\lambda(\theta) = 1$ and g on Γ_2 is defined by (2.27).

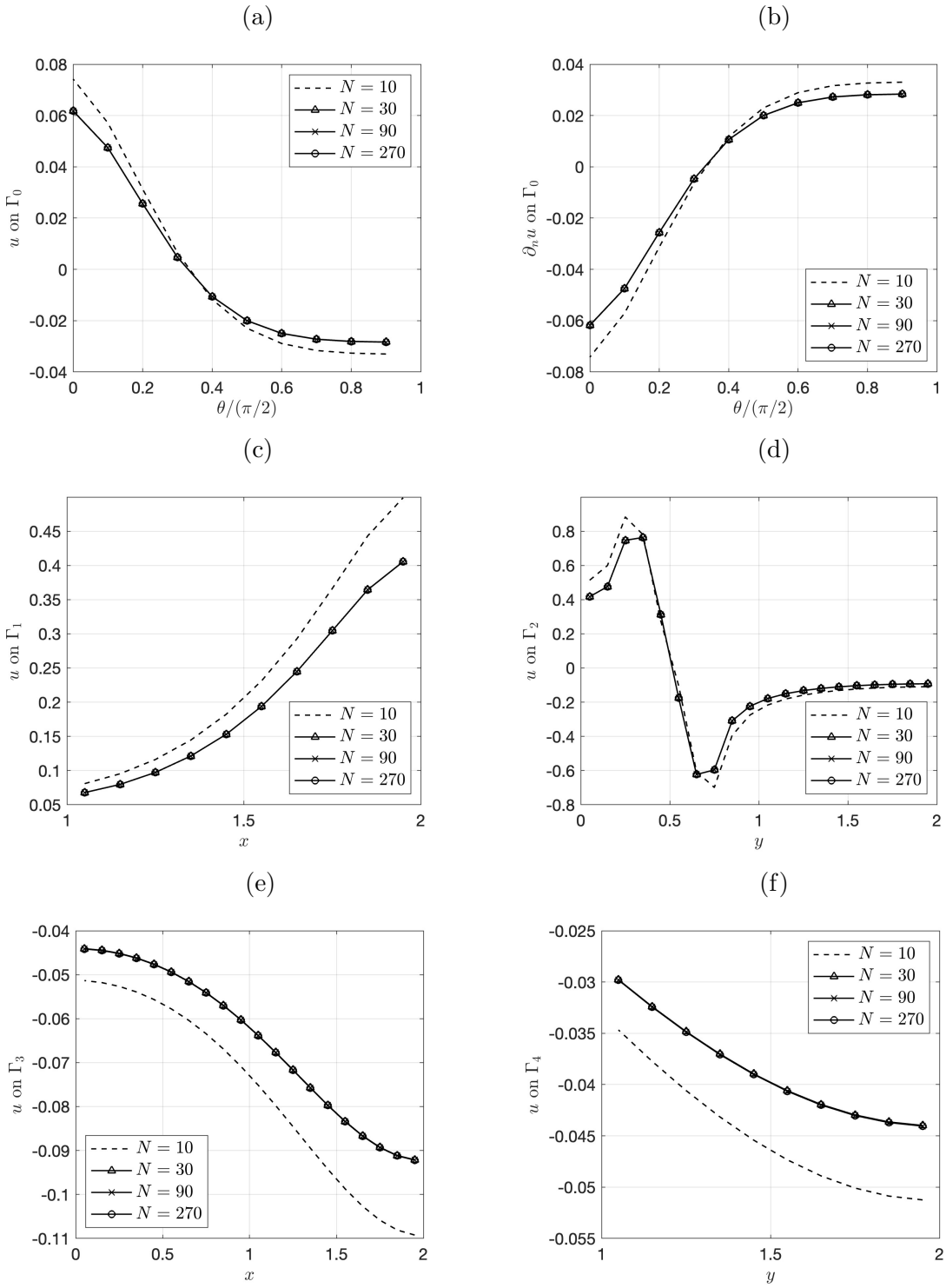

Figure 2.7: Comparison between the BEM values of u over $\partial\Omega$ and $\partial_n u$ over Γ_0 for various $N \in \{10, 30, 90, 270\}$, for the LBC (2.30), when $\lambda(\theta) = -1$ and g on Γ_2 is defined by (2.27).

processing times, preventing the completion of calculations in a reasonable time-frame.

Table 2.3: The numerical BEM and MFS solutions for the LBC (2.30) and for the NLBC (2.33) on Γ_0, at arbitrary test points in Ω, for various $N \in \{10, 30, 90, 270\}$, when $\lambda(\theta) = -1$ and g on Γ_2 is defined by (2.27).

	N	u_{LBC}		u_{NLBC}	
		BEM	MFS	BEM	MFS
(1.5, 0.5)	10	0.0575	0.0353	0.0547	0.0353
	30	0.0469	0.0196	0.0451	0.0462
	90	0.0469	0.0470	0.0451	0.0468
	270	0.0469	0.0469	0.0450	
(1.5, 1.5)	10	-0.1014	-0.0824	-0.1031	-0.0824
	30	-0.0856	-0.1062	-0.0865	-0.0861
	90	-0.0857	-0.0856	-0.0865	-0.0857
	270	-0.0857	-0.0857	-0.0866	
(0.5, 1.5)	10	-0.0511	-0.0433	-0.0507	-0.0433
	30	-0.0437	-0.0592	-0.0428	-0.0442
	90	-0.0437	-0.0438	-0.0429	-0.0438
	270	-0.0437	-0.0438	-0.0429	

2.4.2 Examples with the continuous current flux (2.28) and $\lambda(\theta) = \pm 1$

2.4.2.1 Example 3: Continuous current flux (2.28) and $\lambda(\theta) = 1$

The third case considers the continuous flux function (2.28) imposed on Γ_2, along with $\lambda(\theta) = 1$ on Γ_0. This smooth flux distribution differs from the previous piecewise constant function by providing a continuous current variation along Γ_2. Table 2.4 shows the excellent agreement between the numerical solutions obtained by both the BEM and MFS at selected points within Ω for the linear (2.30) and nonlinear (2.33) boundary conditions on Γ_0. Also, Figure 2.8 shows that the BEM achieves convergent and accurate results across different mesh sizes, including $N \in \{10, 30, 90, 270\}$. Furthermore, the convergence of the BEM solution for $u|_{\Gamma_2} =: h$ with increasing N, as depicted in Table 2.5 and Figure 2.8(d), reinforces the convergence and accuracy of the solution along Γ_2. Figure 2.9 presents the contour plot of the numerical solution over the domain Ω.

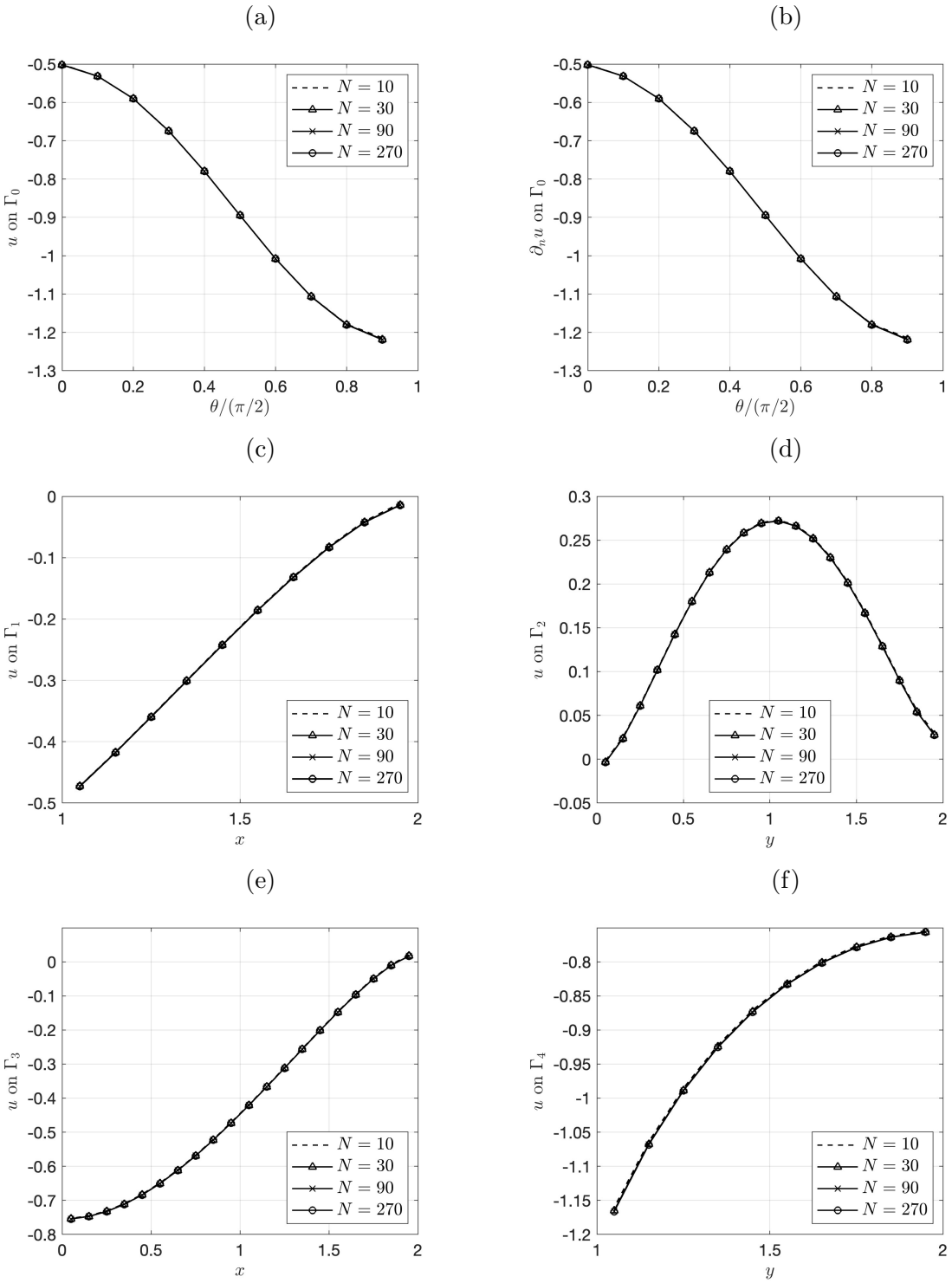

Figure 2.8: Comparison between the BEM values of u over $\partial\Omega$ and $\partial_n u$ over Γ_0 for various $N \in \{10, 30, 90, 270\}$, for the LBC (2.30), when $\lambda(\theta) = 1$ and g on Γ_2 is defined by (2.28).

Table 2.4: The numerical BEM and MFS solutions for the LBC (2.30) and for the NLBC (2.33) on Γ_0, at arbitrary test points in Ω, for various $N \in \{10, 30, 90, 270\}$, when $\lambda(\theta) = 1$ and g on Γ_2 is defined by (2.28).

	N	u_{LBC}		u_{NLBC}	
		BEM	MFS	BEM	MFS
(1.5, 0.5)	10	-0.1888	-0.1905	-0.1411	-0.1417
	30	-0.1896	-0.1900	-0.1417	-0.1418
	90	-0.1899	-0.1900	-0.1418	-0.1419
	270	-0.1900	-0.1900	-0.1419	
(1.5, 1.5)	10	-0.1568	-0.1585	-0.1249	-0.1258
	30	-0.1581	-0.1585	-0.1261	-0.1263
	90	-0.1584	-0.1585	-0.1264	-0.1264
	270	-0.1585	-0.1585	-0.1264	
(0.5, 1.5)	10	-0.7350	-0.7371	-0.7196	-0.7214
	30	-0.7371	-0.7376	-0.7219	-0.7224
	90	-0.7376	-0.7376	-0.7224	-0.7225
	270	-0.7376	-0.7377	-0.7225	

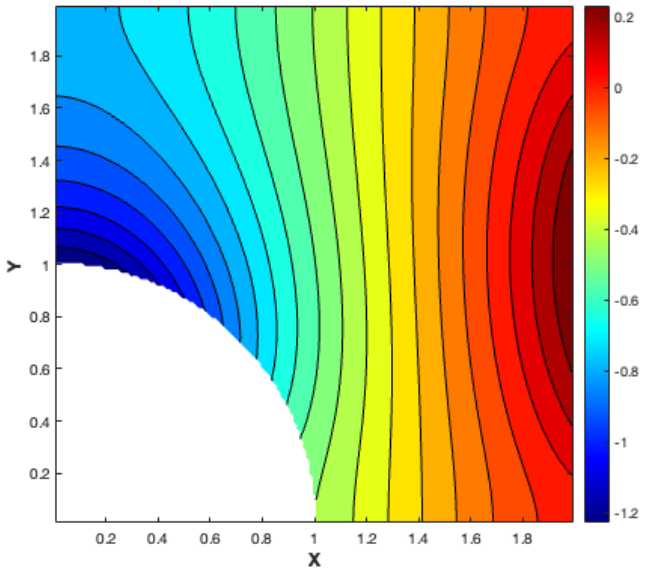

Figure 2.9: Contour plot of the numerical solution computed using the BEM over the domain Ω for the NLBC (2.33) on Γ_0, with $N = 30$, $\lambda = 1$ and g on Γ_2 defined by (2.28).

Table 2.5: The y-coordinates of the midpoints on Γ_2 and the BEM numerical solutions for $u|_{\Gamma_2} = h$ obtained with various $N \in \{10, 30, 90\}$ for the LBC (2.30) and g on Γ_2 is defined by (2.28).

y	$h_{N=10}$	$h_{N=30}$	$h_{N=90}$
0.05	-0.0017	-0.0032	-0.0034
0.15	0.0249	0.0237	0.0235
0.25	0.0619	0.0609	0.0607
0.35	0.1027	0.1019	0.1017
0.45	0.1433	0.1426	0.1424
0.55	0.1810	0.1803	0.1801
0.65	0.2137	0.2131	0.2129
0.75	0.2406	0.2395	0.2393
0.85	0.2592	0.2586	0.2584
0.95	0.2703	0.2696	0.2694
1.05	0.2729	0.2722	0.2720
1.15	0.2670	0.2663	0.2661
1.25	0.2529	0.2521	0.2519
1.35	0.2310	0.2301	0.2299
1.45	0.2023	0.2013	0.2011
1.55	0.1681	0.1669	0.1667
1.65	0.1302	0.1289	0.1287
1.75	0.0915	0.0900	0.0898
1.85	0.0559	0.0541	0.0539
1.95	0.0299	0.0279	0.0276

2.4.2.2 Example 4: Continuous current flux (2.28) and $\lambda(\theta) = -1$

Table 2.6 presents the numerical solutions of the BEM and the MFS for both the LBC (2.30) and the NLBC (2.33) on Γ_0, evaluated at selected points within the domain Ω for different values of $N \in \{10, 30, 90, 270\}$ when $\lambda(\theta) = -1$ and the current flux g on Γ_2 is given by (2.28). At the point $(1.5, 0.5)$, for instance, the BEM and MFS solutions for the LBC (2.30) start from 1.4124 and 1.4130, respectively, for $N = 10$, and converge closely to 1.4117 and 1.4118 by $N = 270$. Similarly, for the NLBC (2.33), the numerical solutions at point $(1.5, 0.5)$ also show consistent convergence between BEM and MFS, as N increases. A similar trend is observed at the other two points, $(1.5, 1.5)$ and $(0.5, 1.5)$, where for both the LBC and NLBC, the numerical solutions converge to nearly identical results for both the BEM and MFS, especially at higher values of N. However, achieving convergence for the NLBC at the highest discretization level of $N = 270$ is computationally intensive, requiring significantly more time than the linear case.

Table 2.6: The numerical BEM and MFS solutions for the LBC (2.30) and for the NLBC (2.33) on Γ_0, at arbitrary test points in Ω, for various $N \in \{10, 30, 90, 270\}$, when $\lambda(\theta) = -1$ and g on Γ_2 is defined by (2.28).

	N	u_{LBC}		u_{NLBC}	
		BEM	MFS	BEM	MFS
(1.5, 0.5)	10	1.4124	1.4130	1.3871	1.3870
	30	1.4112	1.4119	1.3859	1.3865
	90	1.4116	1.4118	1.3863	1.3864
	270	1.4117	1.4118	1.3864	
(1.5, 1.5)	10	1.5390	1.5401	1.5149	1.5153
	30	1.5381	1.5389	1.514	1.5147
	90	1.5386	1.5388	1.5145	1.5146
	270	1.5387	1.5388	1.5146	
(0.5, 1.5)	10	1.0566	1.0577	1.0337	1.0341
	30	1.0558	1.0565	1.0329	1.0335
	90	1.0562	1.0564	1.0333	1.0334
	270	1.0563	1.0564	1.0334	

In Figure 2.10, the BEM produces accurate and convergent results for the unspecified boundary data under various mesh sizes, including $N \in \{10, 30, 90, 270\}$. Finally, Figure 2.11 shows that, the LBC and NLBC boundary conditions produce similar solutions for $u|_{\Gamma_0}$ and $u|_{\Gamma_2}$ in cases $\lambda(\theta) = \pm 1$ for the discontinuous (2.27) and continuous (2.28) fluxes. This is to be expected since the linear and nonlinear functions (2.6) and (2.5) are very close to each other for $|u| \leq 1$.

2.4.3 Examples with the discontinuous current flux (2.27) and $\lambda(\theta) = \pm \sin(\theta)$

Having established the reliability and convergence behavior of the BEM and MFS for constant values of $\lambda(\theta) = \pm 1$ under both discontinuous and continuous current flux boundary conditions on Γ_2, it is natural to explore how these numerical methods perform when the boundary parameter λ exhibits spatial variation. In many physical scenarios, such as corrosion processes, this boundary parameter may not remain uniform but instead depend on position along the boundary. To investigate this further and assess the adaptability of the BEM and MFS to more realistic situations, we extend our analysis to include cases where λ varies sinusoidally as $\lambda(\theta) = \pm \sin(\theta)$, with θ representing the angular coordi-

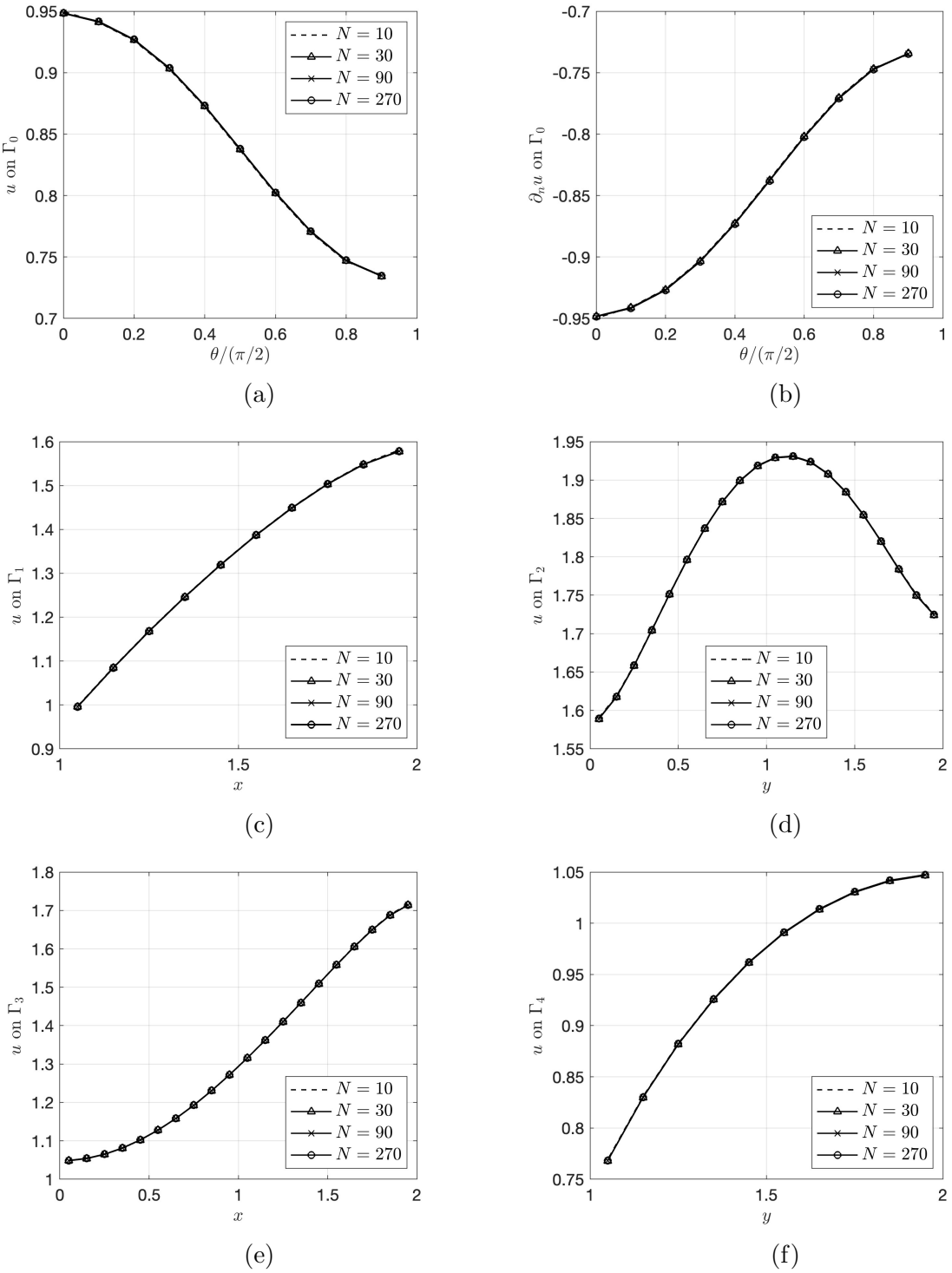

Figure 2.10: Comparison between the BEM values of u over $\partial\Omega$ and $\partial_n u$ over Γ_0 for various $N \in \{10, 30, 90, 270\}$, for the LBC (2.30), when $\lambda(\theta) = -1$ and g on Γ_2 is defined by (2.28).

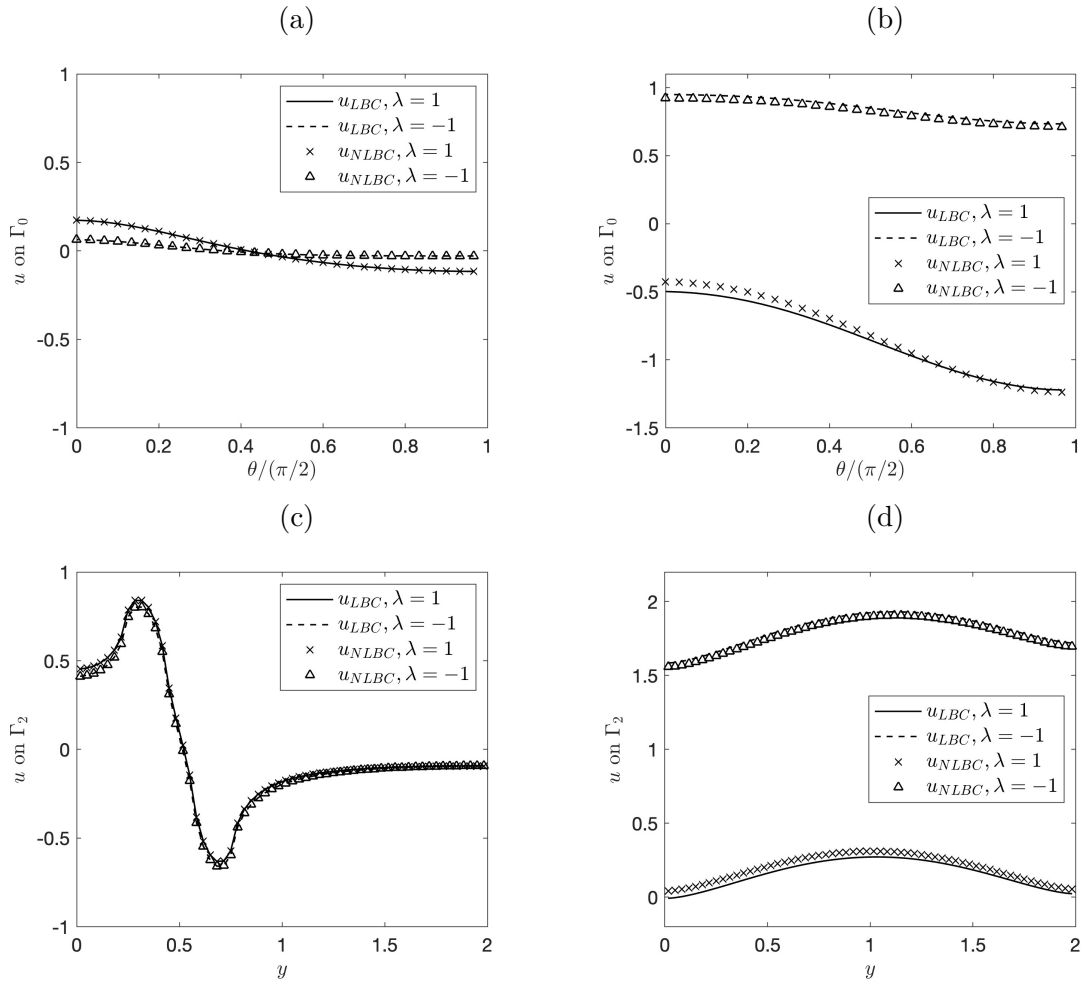

Figure 2.11: Comparison of the BEM numerical solutions for u on $\Gamma_0 \cup \Gamma_2$ obtained with $N = 30$ for the LBC (2.30) and NLBC (2.33) in the direct problem when $\lambda(\theta) = \pm 1$ and (a, c) Examples 1 and 2 with the discontinuous current flux (2.27) and (b, d) Examples 3 and 4 with the continuous current flux (2.28).

Table 2.7: The numerical BEM and MFS solutions for the LBC (2.30) and for the NLBC (2.33) on Γ_0, at arbitrary test points in Ω, for various $N \in \{10, 30, 90, 270\}$, when $\lambda(\theta) = \sin(\theta)$ and g on Γ_2 is defined by (2.27).

	N	u_{LBC}		u_{NLBC}	
		BEM	MFS	BEM	MFS
(1.5, 0.5)	10	0.1082	0.0813	0.1082	0.0813
	30	0.0906	0.0909	0.0906	0.0905
	90	0.0905	0.0903	0.0905	0.0903
	270	0.0904	0.0904	0.0904	
(1.5, 1.5)	10	-0.0666	-0.0499	-0.0666	-0.0499
	30	-0.0565	-0.0460	-0.0565	-0.0570
	90	-0.0570	-0.0575	-0.0570	-0.0575
	270	-0.0572	-0.0574	-0.0572	
(0.5, 1.5)	10	-0.0318	-0.0238	-0.0318	-0.0238
	30	-0.0287	-0.0098	-0.0287	-0.0297
	90	-0.0297	-0.0305	-0.0297	-0.0304
	270	-0.0300	-0.0303	-0.0300	

nate along the boundary Γ_0. This introduces a spatially oscillating boundary condition, providing a richer test of the numerical methods' accuracy and robustness.

To this end, we present four additional examples, mirroring the structure of the previous analysis.

2.4.3.1 Example 5: Discontinuous current flux (2.27) and $\lambda(\theta) = \sin(\theta)$

Table 2.7 shows very good agreement between BEM and MFS results at selected interior points in Ω. Also, Figure 2.12 shows the convergence of the BEM for $u|_{\partial\Omega}$ with increasing the number of boundary elements $N \in \{30, 90, 270\}$. The contour plot illustrating the numerical solution u is shown in Figure 2.13.

2.4.3.2 Example 6: Discontinuous current flux (2.27) and $\lambda(\theta) = -\sin(\theta)$

As previously obtained in Table 2.7, Table 2.8 shows the very good agreement obtained between the BEM and MFS solutions at selected points in Ω. Also, Figure 2.14 illustrates the convergence of the BEM solution for $u|_{\partial\Omega}$ as the number of boundary elements $N \in \{30, 90, 270\}$ increase.

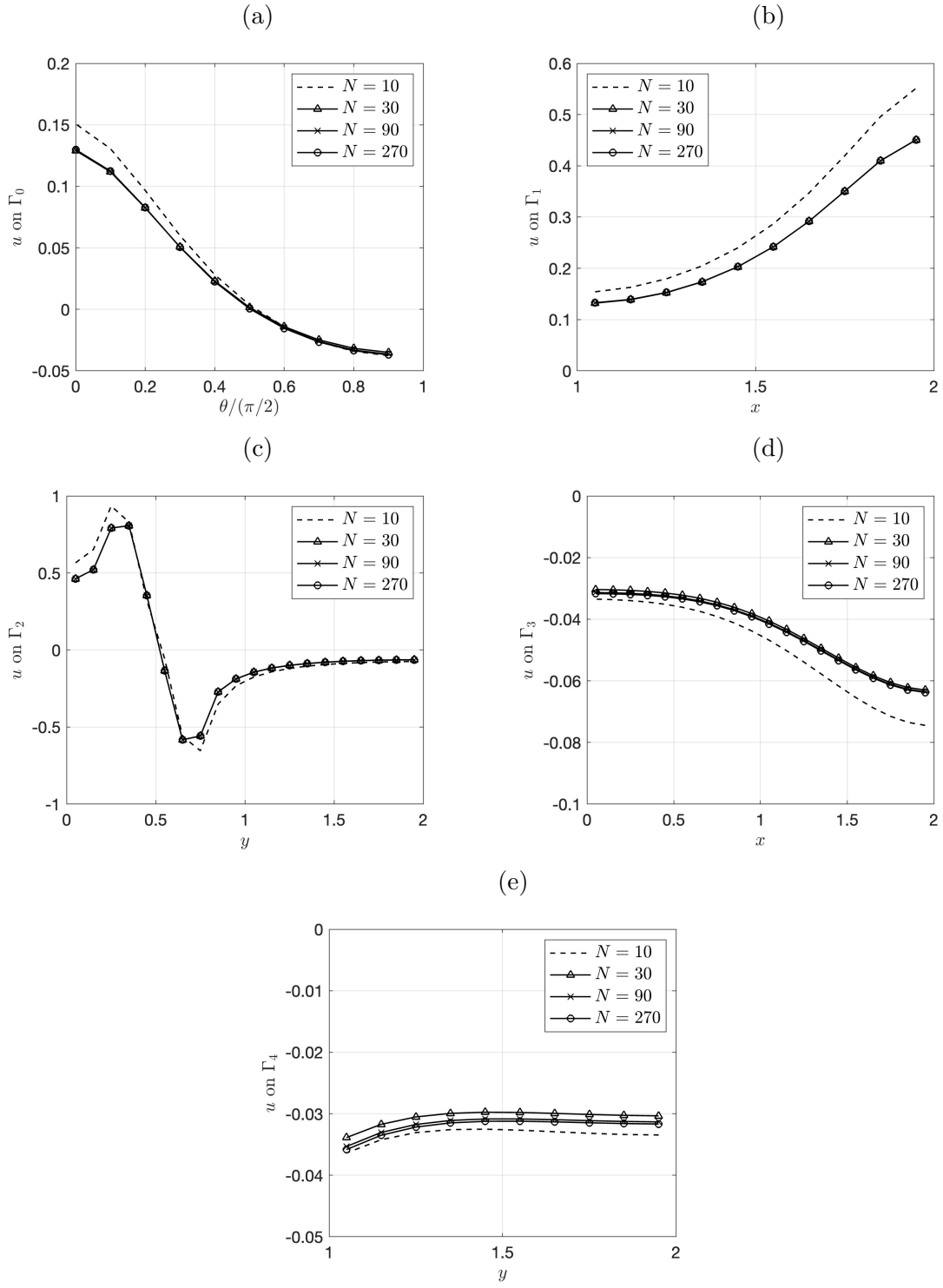

Figure 2.12: Comparison between the BEM values of u over $\partial\Omega$ for various $N \in \{10, 30, 90, 270\}$, for the LBC (2.30), when $\lambda(\theta) = \sin(\theta)$ and g on Γ_2 is defined by (2.27).

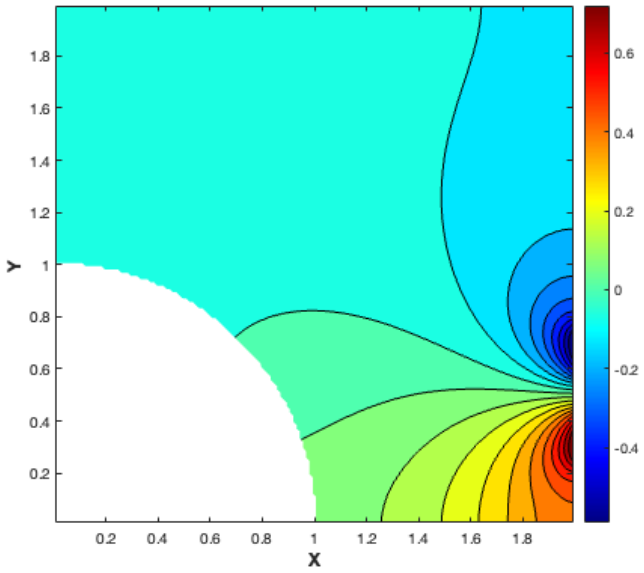

Figure 2.13: Contours of the potential u obtained from the numerical solution of the BEM over the domain Ω for the NLBC (2.33) on Γ_0, with $N = 30$, $\lambda(\theta) = \sin(\theta)$ and g on Γ_2 defined by (2.27).

Table 2.8: The numerical BEM and MFS solutions for the LBC (2.30) and for the NLBC (2.33) on Γ_0, at arbitrary test points in Ω, for various $N \in \{10, 30, 90, 270\}$, when $\lambda(\theta) = -\sin(\theta)$ and g on Γ_2 is defined by (2.27).

	N	u_{LBC}		u_{NLBC}	
		BEM	MFS	BEM	MFS
(1.5, 0.5)	10	0.0939	0.0526	0.0939	0.0526
	30	0.0732	0.0300	0.0732	0.0681
	90	0.0720	0.0712	0.0720	0.0712
	270	0.0717	0.0714	0.0717	
(1.5, 1.5)	10	-0.0724	-0.0690	-0.0724	-0.0689
	30	-0.0649	-0.0988	-0.0649	-0.0691
	90	-0.0659	-0.0665	-0.0659	-0.0666
	270	-0.0662	-0.0664	-0.0662	
(0.5, 1.5)	10	-0.0280	-0.0329	-0.0280	-0.0329
	30	-0.0274	-0.0539	-0.0274	-0.0308
	90	-0.0282	-0.0286	-0.0282	-0.0287
	270	-0.0284	-0.0285	-0.0284	

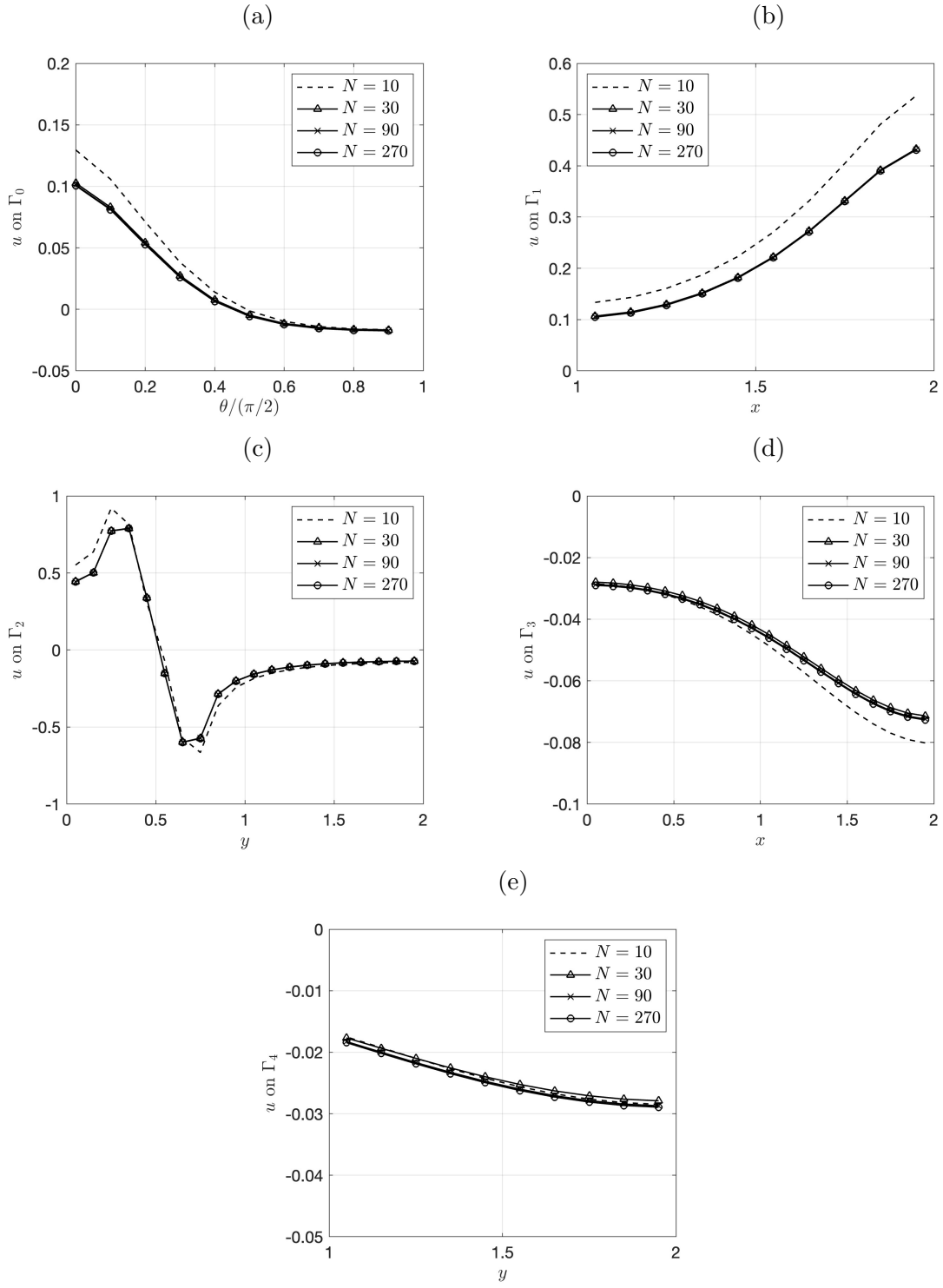

Figure 2.14: Comparison between the BEM values of u over $\partial\Omega$ for various $N \in \{10, 30, 90, 270\}$, for the LBC (2.30), when $\lambda(\theta) = -\sin(\theta)$ and g on Γ_2 is defined by (2.27).

Table 2.9: The numerical BEM and MFS solutions for the LBC (2.30) and NLBC (2.33) on Γ_0, at various points in Ω, for various $N \in \{10, 30, 90, 270\}$, when $\lambda(\theta) = \sin(\theta)$ and g on Γ_2 is defined by (2.28).

	N	u_{LBC}		u_{NLBC}	
		BEM	MFS	BEM	MFS
(1.5, 0.5)	10	-0.4264	-0.3436	-0.2709	-0.2182
	30	-0.3846	-0.3584	-0.2443	-0.2278
	90	-0.3726	-0.3639	-0.2366	-0.2312
	270	-0.3687	-0.3658	-0.2342	
(1.5, 1.5)	10	-0.5021	-0.4050	-0.3590	-0.2937
	30	-0.4546	-0.4235	-0.3276	-0.3067
	90	-0.4405	-0.4302	-0.3181	-0.3113
	270	-0.4359	-0.4324	-0.3151	
(0.5, 1.5)	10	-1.1926	-1.0798	-1.0668	-0.9872
	30	-1.1391	-1.1026	-1.0304	-1.0044
	90	-1.1228	-1.1106	-1.0189	-1.0102
	270	-1.1174	-1.1133	-1.0151	

2.4.4 Examples with the continuous current flux (2.28) and $\lambda(\theta) = \pm\sin(\theta)$

2.4.4.1 Example 7: Continuous current flux (2.28) and $\lambda(\theta) = \sin(\theta)$

Table 2.9 shows the good agreement between the BEM and MFS results at interior points inside the solution domain Ω. Also, Figure 2.15 illustrates the convergence of the BEM, demonstrating improved accuracy with a progressively refined mesh achieved by increasing the number of boundary elements $N = 10$ to $N = 270$.

2.4.4.2 Example 8: Continuous current flux (2.28) and $\lambda(\theta) = -\sin(\theta)$

Table 2.10 shows the good agreement between the BEM and MFS for u at selected points inside Ω. Also, Figure 2.16 demonstrates the BEM's convergence showing enhanced accuracy as the mesh is progressively refined by increasing the number of boundary elements from $N = 10$ to $N = 270$.

Finally, Figure 2.17 demonstrates the agreement between numerical solutions for u on $\Gamma_0 \cup \Gamma_2$ obtained using LBC and NLBC methods. Both boundary condition approaches produce close results for the variable parameter cases $\lambda(\theta) = \pm\sin(\theta)$ in both cases the

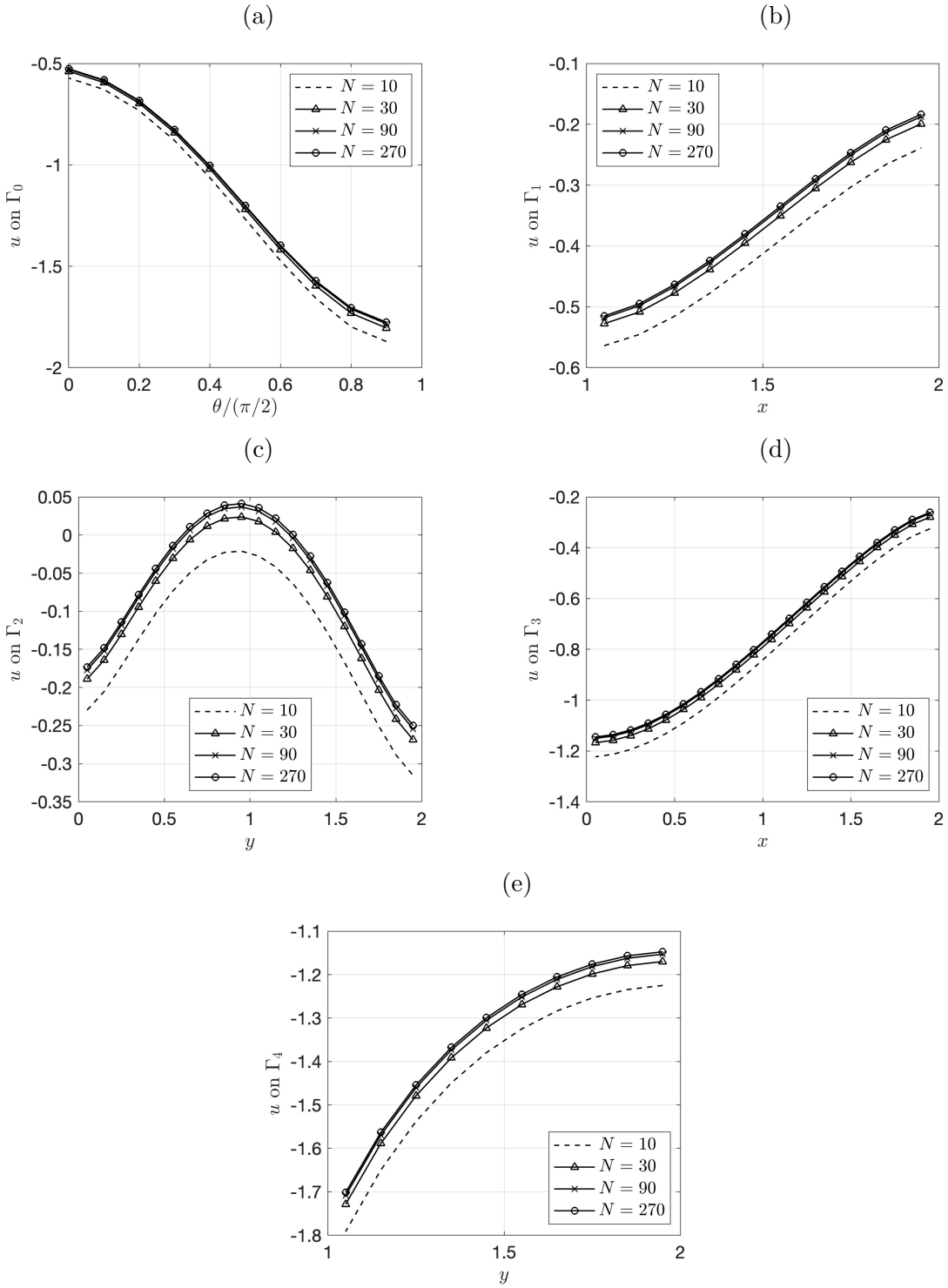

Figure 2.15: Comparison between the BEM values of u over $\partial\Omega$ for various $N \in \{10, 30, 90, 270\}$, for the LBC (2.30), when $\lambda(\theta) = \sin(\theta)$ and g on Γ_2 is defined by (2.28).

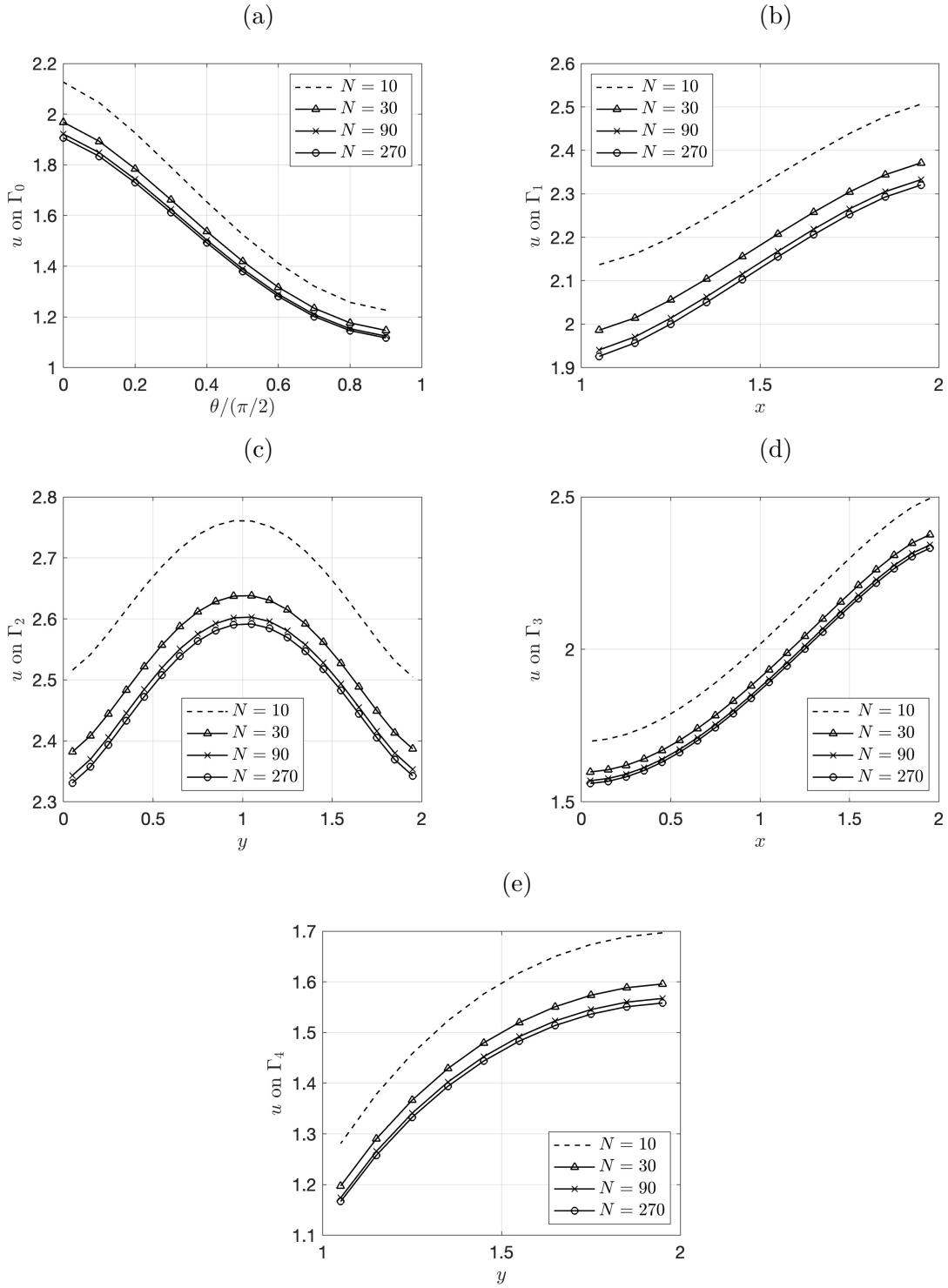

Figure 2.16: Comparison between the BEM values of u over $\partial\Omega$ for various $N \in \{10, 30, 90, 270\}$, for the LBC (2.30), when $\lambda(\theta) = -\sin(\theta)$ and g on Γ_2 is defined by (2.28).

Table 2.10: The numerical BEM and MFS solutions for the LBC (2.30) and for the NLBC (2.33) on Γ_0, at various test points in Ω, for various $N \in \{10, 30, 90, 270\}$, when $\lambda(\theta) = -\sin(\theta)$ and g on Γ_2 is defined by (2.28).

	N	u_{LBC}		u_{NLBC}	
		BEM	MFS	BEM	MFS
(1.5, 0.5)	10	2.3208	2.0579	2.1965	1.9666
	30	2.1897	2.1069	2.0824	2.0101
	90	2.1516	2.1244	2.0492	2.0255
	270	2.1394	2.1304	2.0386	
(1.5, 1.5)	10	2.3168	2.0854	2.1986	1.9994
	30	2.2002	2.1277	2.0988	2.0365
	90	2.1668	2.1430	2.0701	2.0497
	270	2.1561	2.1482	2.0610	
(0.5, 1.5)	10	1.7158	1.5145	1.6044	1.4341
	30	1.6134	1.5507	1.5184	1.4654
	90	1.5844	1.5639	1.4939	1.4766
	270	1.5752	1.5684	1.4862	

discontinuous (2.27) and continuous (2.28) fluxes.

2.5 Summary

In conclusion, the BEM (as well as the MFS) has shown to be a useful strategy for solving a direct problem in the field of corrosion. The outcomes of the BEM and the MFS for both linear and nonlinear boundary conditions on Γ_0 demonstrate how effective these approaches are.

The numerical results across the eight examples in Section 2.4 demonstrate that both methods are in very good agreement and successfully capture the potential distribution under diverse conditions, including discontinuous (2.27) and continuous (2.28) current fluxes imposed on Γ_2 and varying Robin corrosion coefficient $\lambda(\theta) = \pm 1$ and $\lambda(\theta) = \pm\sin(\theta)$ on Γ_0. For both linear (LBC) given by (2.30) and nonlinear boundary conditions (NLBC) given by (2.33), the BEM consistently achieves convergence, as the number of boundary elements N increases, as evidenced by the tight agreement in solutions. The MFS also converges effectively for LBC and higher N, but its performance for NLBC at higher discretizations presents challenges due to computational limitations, underscoring BEM's superior reliability, which is further employed in the remaining of the thesis. The

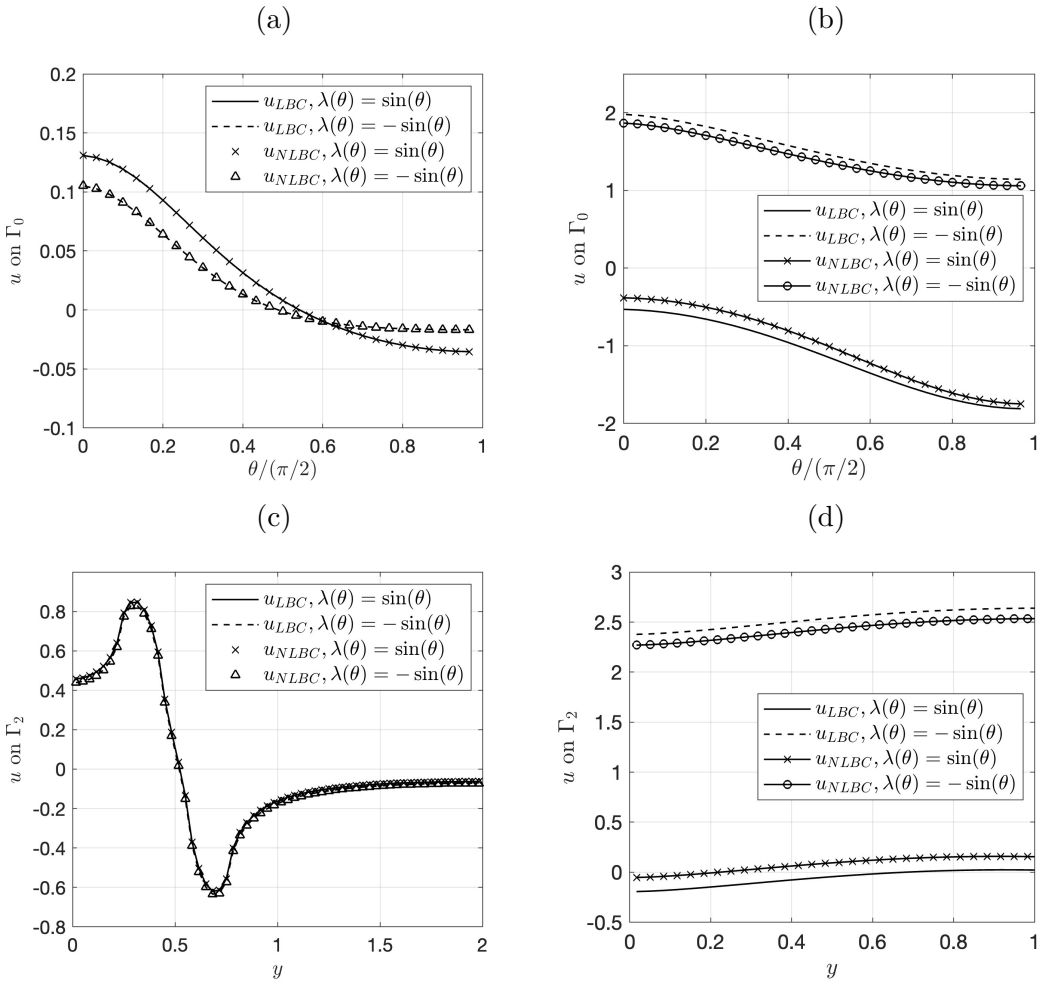

Figure 2.17: Comparison of the BEM numerical solutions for u: (a) on $\Gamma_0 \cup \Gamma_2$ obtained with $N = 30$ for the LBC (2.30) and NLBC (2.33) in the direct problem when $\lambda(\theta) = \pm\sin(\theta)$ and (a,c) Examples 5 and 6 with the discontinuous current flux (2.27) and (b,d) Examples 7 and 8 with the continuous current flux (2.28).

MATLAB computational programs for the BEM and MFS are included in Section A.2 of Appendix A.

Chapter 3

The BEM in an annular domain

In this chapter, we consider electrostatics (or steady-state heat conduction) probing fields in a domain containing a rigid inclusion, e.g., the domain outside an underground pipe, that can be described by a Dirichlet problem for the Laplace equation in an annulus in which on the inner boundary the solution takes an unknown, but constant, value [6, 49]. The extra unknown represented by the constant value of the electric potential on the inner boundary is determined by imposing that the integral of the current flux over the outer boundary of the annular domain vanishes. For such a problem, the BEM is well-suited and it is the purpose of this chapter to implement numerically this boundary integral method for solving the aforementioned inverse problem in an annular domain in order to pave the way for the numerical investigation in subsequent chapters.

3.1 Introduction

One of the most crucial engineering topics for the nuclear power plant's safety administration is corrosion in pipes [101]. Corrosion can lead to leaks, ruptures, and other forms of damage in pipes, which can cause safety hazards and costly repairs. Detecting and monitoring corrosion is, therefore, essential for ensuring the integrity of the pipes and the safety and efficiency of nuclear power plants.

Electrostatics or heat conduction imaging can be used to detect corrosion in pipes by evaluating the electric field or temperature distribution caused by corrosion-induced

changes in the material. This requires resolving Laplace's equation in an approximately annular domain with corrosion deviations and imposing boundary conditions on the inner and outer boundaries to estimate the electric potential or temperature distribution.

Our numerical simulations show that the BEM can accurately determine the current flux and the electric potential on the inner boundary and inside the solution domain, providing a fast and accurate direct solver for the nonlinear minimization that is usually employed in the detection of corrosion [49, 68], see later on Chapters 5–7.

This chapter is organised as follows. In Section 3.2, we formulate the mathematical problem under investigation, whilst in Section 3.3, the BEM is implemented. Section 3.4 discusses the numerically obtained results in order to demonstrate the convergence and accuracy of the BEM. Finally, conclusions are included in Section 3.5.

3.2 Mathematical formulation

Let $\Omega_1 \subset \Omega_0$ be two-dimensional simply-connected bounded domains with sufficiently smooth boundaries $\partial\Omega_1$ and $\partial\Omega_0$, see Fig. 3.1. The solution domain $\Omega := \Omega_0 \backslash \Omega_1$ is the annular domain in between $\partial\Omega_0$ and $\partial\Omega_1$. Let us assume that Ω is a body with constant conductivity. In domain Ω, the electrostatic field is taken into consideration by seeking a harmonic function u, representing the electrical potential, satisfying the Laplace's equation

$$\Delta u = 0 \qquad \text{in} \quad \Omega, \tag{3.1}$$

subject to the Dirichlet boundary conditions

$$u = f \qquad \text{on} \quad \partial\Omega_0, \tag{3.2}$$

$$u = a \qquad \text{on} \quad \partial\Omega_1, \tag{3.3}$$

where f is a given Dirichlet data on the outer boundary $\partial\Omega_0$ and a is a constant. Equation (3.3) models the fact that Ω_1 is a (perfectly conducting) rigid inclusion (while noting that an adiabatic zero-Neumann boundary condition on $\partial\Omega_1$ models a cavity). If a is given,

then the problem in Eqs. (3.1)-(3.3) has a unique solution u. However, if the constant a is not known then the additional condition

$$\int_{\partial\Omega_0} \frac{\partial u}{\partial n} ds = 0, \tag{3.4}$$

ensures the uniqueness of solution (u, a) to the problem in Eqs. (3.1)-(3.4), see [49]. The BEM numerical approximations of the problem in Eqs. (3.1)-(3.3) with a known, or Eqs. (3.1)-(3.4) with a unknown, will be illustrated in Section 3.3.

In electrical impedance tomography (EIT), the obstacle Ω_1 is also unknown and in order to compensate for this missing information one measures the current flux g given by

$$\frac{\partial u}{\partial n} = g \qquad \text{on} \quad \partial\Omega_0. \tag{3.5}$$

This more difficult nonlinear inverse problem of EIT, [21, 49, 67], will be investigated later on in Chapter 6 in the context of boundary corrosion detection.

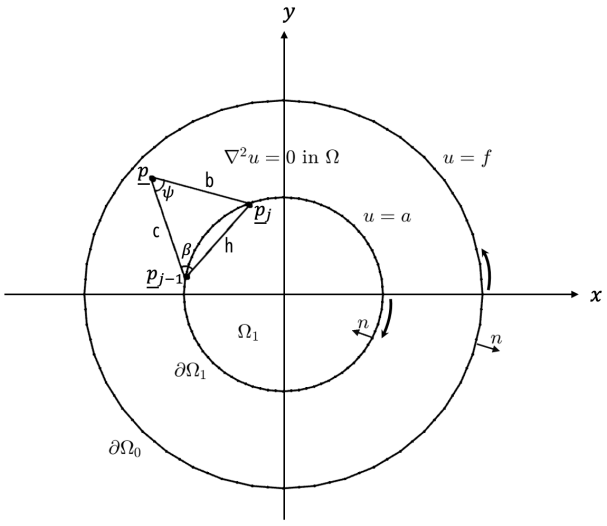

Figure 3.1: The boundary value problem in Eqs. (3.1)-(3.3) and the BEM discretisation of the boundary $\partial\Omega = \partial\Omega_0 \cup \partial\Omega_1$.

3.3 Boundary element method (BEM)

In [49], the author employed a single-layer representation of the harmonic function u that was further differentiated in the tangential and normal directions at the boundary

in order to apply the conditions (3.3) and (3.4). In our study, we use a combined single-double layer classical representation, based on the Green's formula, which avoids these differentiations and enables direct applications of conditions (3.3) and (3.4).

Application of the BEM to Laplace's equation (3.1) in two-dimensions, [79], results in (2.8). Applying (2.8) at $\underline{p} \in \partial\Omega = \partial\Omega_0 \cup \partial\Omega_1$ and using the boundary conditions (3.2) and (3.3) results in

$$\int_{\partial\Omega} G(\underline{p},\underline{p}')\frac{\partial u}{\partial n}(\underline{p}')ds(\underline{p}') - \int_{\partial\Omega_0} f(\underline{p}')\frac{\partial G}{\partial n(\underline{p}')}(\underline{p},\underline{p}')ds(\underline{p}') \tag{3.6}$$
$$-a\int_{\partial\Omega_1} \frac{\partial G}{\partial n(\underline{p}')}(\underline{p},\underline{p}')ds(\underline{p}') = \begin{cases} \frac{1}{2}f(\underline{p}) & \text{if } \underline{p} \in \partial\Omega_0, \\ \frac{1}{2}a & \text{if } \underline{p} \in \partial\Omega_1. \end{cases}$$

We discretise the boundary $\partial\Omega = \partial\Omega_0 \cup \partial\Omega_1$ of the solution domain $\Omega = \Omega_0 \backslash \Omega_1$ into a series of $M = 2N$ boundary elements. The outer boundary $\partial\Omega_0$ is discretised anti-clockwise into a series of N straight line segments $\Gamma_j^0 = [\underline{p}_{j-1}, \underline{p}_j]$ for $j = \overline{1,N}$, where $\underline{p}_N = \underline{p}_0$, while the inner boundary $\partial\Omega_1$ is discretised clockwise into a series of N straight line segments $\Gamma_{j-N}^1 = [\underline{p}_j, \underline{p}_{j+1}]$ for $j = \overline{N+1,2N}$, where $\underline{p}_{2N+1} = \underline{p}_{N+1}$. Let us denote

$$\Gamma_j = \begin{cases} \Gamma_j^0 & \text{for } j = \overline{1,N}, \\ \Gamma_j^1 & \text{for } j = \overline{N+1,2N}. \end{cases}$$

Over each small boundary element Γ_j, the unknown Neumann data $\partial u/\partial n$ is assumed constant and takes the value at the midpoint (boundary element node)

$$\tilde{\underline{p}}_j = \begin{cases} (\underline{p}_j + \underline{p}_{j-1})/2 & \text{for } j = \overline{1,N}, \\ (\underline{p}_{j+1} + \underline{p}_j)/2 & \text{for } j = \overline{N+1,2N}, \end{cases}$$

namely,

$$\frac{\partial u}{\partial n}(\underline{p}) \approx \frac{\partial u}{\partial n}(\tilde{\underline{p}}_j) =: u_j' \qquad \text{for} \quad \underline{p} \in \Gamma_j, \quad j = \overline{1,2N}. \tag{3.7}$$

We also approximate the Dirichlet data f in (3.2) by a piece-wise constant function as

$$f(\underline{p}) \approx f(\underline{\tilde{p}}_j) =: f_j \qquad \text{for} \quad \underline{p} \in \Gamma_j^0, \quad j = \overline{1, N}. \tag{3.8}$$

Denote

$$A_j(\underline{p}) := \int_{\Gamma_j} G(\underline{p}, \underline{p}') ds(\underline{p}'), \quad B_j(\underline{p}) := \int_{\Gamma_j} \frac{\partial G}{\partial n(\underline{p}')}(\underline{p}, \underline{p}') ds(\underline{p}'), \quad \underline{p} \in \overline{\Omega}, \quad j = \overline{1, 2N}. \tag{3.9}$$

With (3.7) and (3.8), the boundary integral equation (3.6) is approximated as

$$\sum_{j=1}^{2N} A_j(\underline{p}) u_j' - \sum_{j=1}^{N} B_j(\underline{p}) f_j - a \sum_{j=N+1}^{2N} B_j(\underline{p}) = \begin{cases} \dfrac{1}{2} f(\underline{p}) & \text{if } \underline{p} \in \partial\Omega_0, \\ \dfrac{1}{2} a & \text{if } \underline{p} \in \partial\Omega_1. \end{cases} \tag{3.10}$$

Collocating (3.10) at $\underline{p} = \underline{\tilde{p}}_i$ for $i = \overline{1, 2N}$, we obtain

$$\sum_{j=1}^{2N} A_{ij} u_j' + \sum_{j=1}^{2N} B_{ij} f_j = 0, \qquad i = \overline{1, 2N}. \tag{3.11}$$

where

$$\begin{gathered} A_{ij} := A_j(\underline{\tilde{p}}_i), \quad B_{ij} := -\frac{1}{2}\delta_{ij} - B_j(\underline{\tilde{p}}_i), \quad i, j = \overline{1, 2N}, \\ f_j = f(\underline{\tilde{p}}_j), \quad j = \overline{1, N} \quad \text{and} \quad f_j = a, \quad j = \overline{N+1, 2N}, \end{gathered} \tag{3.12}$$

and δ_{ij} the Kronecker delta symbol.

In the case a is known, the system of (3.11) can be written in compact form as

$$A\underline{u'} = -B\underline{f} \tag{3.13}$$

and solved using the LU decomposition method.

In the case a is unknown, the system of (3.11) can be written in the form

$$\sum_{j=1}^{2N} A_{ij}u'_j + a\sum_{j=N+1}^{2N} B_{ij} = -\sum_{j=1}^{N} B_{ij}f_j, \qquad i = \overline{1,2N}. \tag{3.14}$$

Equation (3.4) is also employed in the discretized form,

$$\sum_{j=1}^{N} h_j u'_j = 0, \tag{3.15}$$

where $h_j := \int_{\Gamma_j} ds(\underline{p}')$ for $j = \overline{1,N}$. The new system of linear (3.14) and (3.15) is solved using the LU decomposition method to find the Neumann values u'_j for $j = \overline{1,2N}$ and the constant a. Once these have been found accurately, the following discretisation of the integral (3.10)

$$u(\underline{p}) = \sum_{j=1}^{2N} A_j(\underline{p})u'_j - \sum_{j=1}^{N} B_j(\underline{p})f_j - a\sum_{j=N+1}^{2N} B_j(\underline{p}), \quad \underline{p} \in \Omega, \tag{3.16}$$

provides explicitly the solution $u(\underline{p})$ at any point $\underline{p}$ inside the solution domain Ω.

3.4 Numerical results and discussion

Let Ω_0 and Ω_1 be the circles of radii R_0 and R_1, where $0 < R_1 = 0.5 < R_0 = 1$ centred at the origin, as shown in Figure 3.1.

3.4.1 Example when a is known

With the boundary conditions (3.2) and (3.3) given by

$$u(r,\theta)|_{r=R_0=1} = f(R_0,\theta) = 3\cos(\theta) \qquad \text{on} \quad \partial\Omega_0, \tag{3.17}$$

$$u(r,\theta)|_{r=R_1=0.5} = a = 0 \qquad \text{on} \quad \partial\Omega_1, \tag{3.18}$$

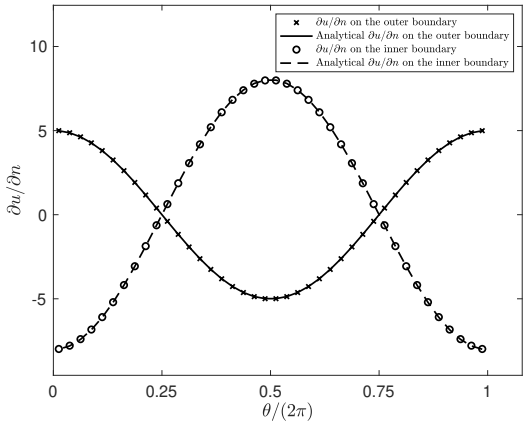

Figure 3.2: The analytical (3.20) and numerical normal derivatives on the outer and inner boundaries, as functions of $\theta/(2\pi)$, obtained using the BEM with $N = 40$, for the problem given by Eqs. (3.1), (3.17) and (3.18).

it is easy to see that the problem given by Eqs. (3.1), (3.17) and (3.18) has the analytical solution

$$u(r,\theta) = 4r\cos(\theta) - \frac{\cos(\theta)}{r}, \quad R_1 < r < R_0, \quad \theta \in [0, 2\pi). \tag{3.19}$$

From (3.19), the exact flux is given by

$$\left.\frac{\partial u}{\partial n}\right|_{r=R_0} = 4\cos(\theta) + \frac{\cos(\theta)}{R_0^2}, \quad \left.\frac{\partial u}{\partial n}\right|_{r=R_1} = -\left(4\cos(\theta) + \frac{\cos(\theta)}{R_1^2}\right), \quad \theta \in [0, 2\pi). \tag{3.20}$$

As shown in Figure 3.2, the agreement between the BEM numerical solution and the exact solution for the normal derivative is excellent.

In order to choose the right number of boundary elements for a specific problem, it is crucial to know how fast the numerical solution reaches the true solution, as the number of boundary elements increases. In Table 3.1, the number of boundary elements used determines the accuracy of the numerical solution obtained using the BEM. For instance, at $(-0.3, 0.6)$, the numerical solution becomes more accurate since the absolute error $\{0.0153, 0.0049, 0.0012, 0.0003\}$ decreases as the number of boundary elements increases from $N = 10$ to 20, 40 and 80, as shown in Table 3.1. The solutions become more accurate as more boundary elements are used.

It is illustrative to examine the convergence behaviour of the BEM using a plot, see Figure 3.3(a), and calculate order of convergence of the BEM. The computational order

	u_N				u_{exact}
	$N = 10$	$N = 20$	$N = 40$	$N = 80$	
$u(0.8, 0.0)$	1.8793	1.9363	1.9467	1.9491	1.9500
$u(0.6, 0.3)$	1.0296	1.0568	1.064	1.0660	1.0666
$u(-0.3, 0.6)$	-0.5179	-0.5284	-0.5320	-0.5330	-0.5333

Table 3.1: The exact (3.19) and numerical BEM solutions obtained with various numbers of boundary elements $N \in \{10, 20, 40, 80\}$, for the problem given by Eqs. (3.1), (3.17) and (3.18).

of convergence is defined as [51],

$$\gamma \approx \frac{\ln\left(\frac{e_{new}}{e_{old}}\right)}{\ln\left(\frac{N_{new}}{N_{old}}\right)}, \tag{3.21}$$

where e_{new} and e_{old} denote the absolute errors between the exact solution and the numerical solutions obtained with N_{new} and N_{old} boundary elements, respectively. For our problem we have calculated the order of convergence for $u(0.8, 0.0)$ and obtained $\gamma \in \{2.3699, 2.0625, 1.9965\}$ for $\{10, 20, 40\} \ni N_{old} = N_{new}/2$, see Figure 3.3(a). This indicates that the order of convergence is quadratic.

The L^2-norm of error gives a measure of the overall accuracy of the numerical approximation, see Figure 3.3(b). To compute the L^2-norm numerically, we use a Gaussian quadrature technique over a triangulated mesh that covers the domain Ω using the PDE Toolbox in MATLAB. The sum of the approximations over all mesh elements yields the L^2-norm squared of the error. The formula for the L^2-norm of a function $U(x, y)$ over a domain Ω is

$$||U||_{L^2(\Omega)} = \left(\int_\Omega U^2(x, y) dx dy \right)^{1/2}. \tag{3.22}$$

By using (3.21), the estimated values of γ $\{4.7044, 1.8012, 2.1346\}$ for $\{10, 20, 40\} \ni N_{old} = N_{new}/2$ are obtained for the $L^2(\Omega)$ errors presented in Figure 3.3(b), indicating again the quadratic order of convergence.

3.4.2 Example when a is unknown

Consider now the same problem given by (3.1), (3.3), (3.4) and (3.17) in the annular domain $\Omega = \{(r, \theta)\} |\ 0.5 = R_1 < r < R_0 = 1,\ \theta \in [0, 2\pi)\}$, but now the constant a is unknown. Then, (3.14) and (3.15) can be written as the linear system of algebraic

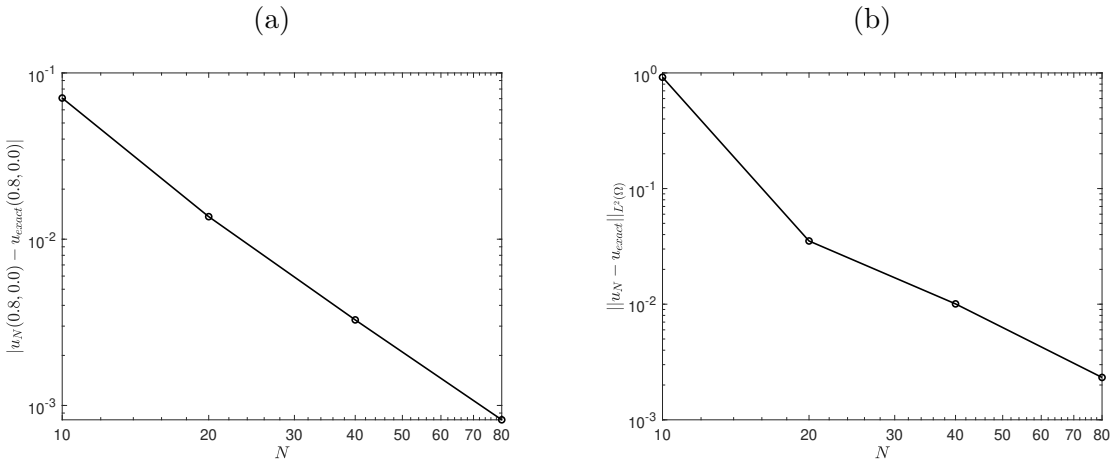

Figure 3.3: The convergence of the BEM at (a) the point $(0.8, 0.0)$ and (b) in $L^2(\Omega)$ for the problem given by Eqs. (3.1), (3.17) and (3.18).

equations

$$\tilde{A}\begin{pmatrix} \underline{u}' = (u'_j)_{j=\overline{1,2N}} \\ a \end{pmatrix} = \underline{C}, \tag{3.23}$$

where the matrix $\tilde{A}$ is given by

$$\tilde{A}_{ij} = \begin{cases} A_{ij} & \text{if } \; i = \overline{1,2N},\ j = \overline{1,2N}, \\ h_j & \text{if } \; i = 2N+1,\ j = \overline{1,N}, \\ 0 & \text{if } \; i = 2N+1,\ j = \overline{N+1,2N+1}, \\ 0 & \text{if } \; i = \overline{1,N},\ j = 2N+1, \\ -1 & \text{if } \; i = \overline{N+1,2N},\ j = 2N+1, \end{cases}$$

and

$$\underline{C} = (C_i)_{i=\overline{1,2N+1}} \text{ with } C_i = -\sum_{j=1}^{N} B_{ij} f_j \text{ for } i = \overline{1,2N} \text{ and } C_{2N+1} = 0.$$

Table 3.2 shows the numerical BEM solutions at the points: $(0.8, 0.0), (0.6, 0.3)$, and $(-0.3, 0.6)$ converging to the exact solution (3.19). The numerically retrieved values of the constant a are also close to the exact value of zero. The estimated values of $\gamma \in \{4.6420, 1.8231, 2.1155\}$ for $\{10, 20, 40\} \ni N_{old} = N_{new}/2$ are obtained for the $L^2(\Omega)$ errors, indicating the quadratic order of convergence. These values are similar to those

	u_N				u_{exact}
	$N = 10$	$N = 20$	$N = 40$	$N = 80$	
$u(0.8, 0.0)$	1.9281	1.9504	1.9503	1.9501	1.9500
$u(0.6, 0.3)$	1.1169	1.0803	1.0701	1.0675	1.0666
$u(-0.3, 0.6)$	-0.5623	-0.5401	-0.5350	-0.5337	-0.5333
a	-4.7E-15	5.6E-14	-3.9E-13	-1.0E-12	0

Table 3.2: The exact (3.19) and numerical BEM solutions obtained with various numbers of boundary elements $N \in \{10, 20, 40, 80\}$ for the problem given by Eqs. (3.1), (3.3), (3.4) and (3.17).

obtained for the problem of the previous section 3.4.1 when a was known.

We consider now an example in which an analytical solution is not available. Here we take the Dirichlet data (3.2) given by, [49],

$$u(r,\theta)|_{r=R_0=1} = f(R_0,\theta) = 10\left(\exp\left[-4\sin^2(\theta/2)\right] - \exp\left[-4\cos^2(\theta/2)\right]\right), \quad \theta \in [0, 2\pi). \quad (3.24)$$

The numerical results presented in Table 3.3 show the convergence of the BEM numerical solution.

	u_N			
	$N = 10$	$N = 20$	$N = 40$	$N = 80$
$u(0.8, 0.0)$	6.5303	6.2997	6.2097	6.1858
$u(0.6, 0.3)$	3.5701	3.2147	3.1353	3.1157
$u(-0.3, 0.6)$	-1.4536	-1.2937	-1.2592	-1.2507
a	-1.4E-15	1.2E-13	-1.2E-12	-4.4E-12

Table 3.3: The numerical BEM solutions obtained with various numbers of boundary elements $N \in \{10, 20, 40, 80\}$ for the problem given by Eqs. (3.1), (3.3), (3.4) and (3.24).

3.5 Summary

The BEM has been applied to solving the Dirichlet problem for the Laplace equation in an annular domain in the case where the inner core represents a rigid inclusion. The constant value a of the potential on the boundary of this perfectly conducting hole may be known or unknown. In the case where a is unknown, an extra condition that the integral of the flux vanishes over the outer boundary of the solution domain has been

enforced. Numerical results have been presented and discussed showing that the BEM is convergent as the number of boundary elements increases and that the convergence is of quadratic order. The MATLAB computational program for the BEM is included in Section A.3 of Appendix A.

Chapter 4

Inverse Cauchy problem

Returning to the quarter-pipe problem investigated in Chapter 2, this chapter focuses on resolving inverse Cauchy problems for the Laplace equation using the BEM, while considering the presence and absence of noise in the input data. In these problems, the objective is to determine the unspecified boundary values of the potential and the flux on the inaccessible (possibly corroded) boundary from the Cauchy data available elsewhere on accessible portions of the boundary [7]. The computational process involves the application of numerical techniques such as the least-squares (LS) method or the singular value decomposition (SVD). The SVD aids in understanding the structure of the system matrix and identifies the components most affected by noise. To evaluate the solution's stability, noise is deliberately introduced into the input data. It is found that the truncated singular value decomposition (TSVD) exhibits noise resilience and stability, thereby ensuring high precision in numerical approximations across varying degrees of data imperfections. Through the dedicated selection of singular values in the TSVD, the approach demonstrates notable improvements in solution's accuracy, particularly in noise reduction. The research meticulously quantifies the impact of noise on solution's accuracy and stability, offering nuanced insights into the optimization of numerical methods for effectively tackling inverse and ill-posed Cauchy problems under diverse conditions.

4.1 Introduction

We consider inverse problems associated with the Laplace equation, focusing on the linear Cauchy problem when the potential u and the flux $\partial_n u$ are partially unspecified along certain sections of the boundary $\partial\Omega$ of the solution domain Ω. This scenario arises in physical situations where certain parts of a specimen's boundary are inaccessible for measurement, e.g., buried or hidden, or in contact with a hostile environment. The missing boundary information is compensated for by over-prescribed boundary conditions (Cauchy data) on accessible friendly sub-portions, providing a way to overcome the information deficit. Solving these problems is crucial in various applications such as in combustion chambers or corrosion engineering, where reconstructing unknown boundary conditions from partial boundary measurements is essential for understanding and optimising the system's performance. Although uniqueness of solution holds due to the unique analytic continuation for the elliptic Laplace's equation, the Cauchy problems are still ill-posed by violating the continuous dependence on the input data, i.e., small errors in the input measured data cause large errors in the output solution. Therefore, solving these inverse problems necessitates the development and implementation of regularization methods, see Section 1.5.2, that can provide stable and accurate solutions. The BEM is particularly useful in discretising these inverse boundary value problems in which all the unknowns are associated with the boundary only.

In addressing linear Cauchy problems, the ordinary linear least-squares (LS) method may be utilized as a preliminary solution to the inverse problem. However, due to the ill-posed nature of Cauchy problems, as mentioned in [15], the direct application of the LS method often leads to solutions that are highly sensitive to noise in the data. This limitation necessitates the adoption of regularization techniques, among which the TSVD is particularly noteworthy.

The TSVD is a powerful technique employed in the field of inverse problems and numerical analysis to stabilize linear ill-posed problems and attenuate the impact of noise in the solution. The application of the TSVD involves the decomposition of the system

matrix into its singular values and vectors, followed by the truncation of small singular values to alleviate the effects of noise, as described in Section 1.5. By retaining only the most significant singular values and associated vectors, TSVD provides a stable solution given by equation (1.22). The application of the TSVD within the BEM framework provides a good approach to tackling the complexities of inverse Cauchy problems. By systematically incorporating regularization, this study mitigates the challenges posed by data imperfections, achieving solutions that are both accurate and stable across various scenarios, including those afflicted by noise.

The subsequent sections of this chapter are organized as follows. Section 4.2 provides an overview of the mathematical formulation of the inverse Cauchy problem. In Section 4.3, we illustrate the computational methodologies for solving the Cauchy problem based on the BEM discretisation. Section 4.4 presents numerical results and discussions for linear and nonlinear boundary conditions under both noise-free and noisy data circumstances. Finally, Section 4.5 offers concluding remarks.

4.2 Mathematical formulation of the Cauchy problem

The investigated inverse Cauchy linear problem is illustrated in Figure 4.1 and formulated by equations (4.1)–(4.4) below. We consider the Laplace equation in the domain Ω,

$$\Delta u = 0 \qquad \text{in} \quad \Omega, \tag{4.1}$$

subject to the boundary conditions

$$u = h \qquad \text{on} \quad \Gamma_2, \tag{4.2}$$

$$\frac{\partial u}{\partial n} = g \quad \text{on} \quad \Gamma_2, \tag{4.3}$$

$$\frac{\partial u}{\partial n} = 0 \quad \text{on} \quad \Gamma_1 \cup \Gamma_3 \cup \Gamma_4, \tag{4.4}$$

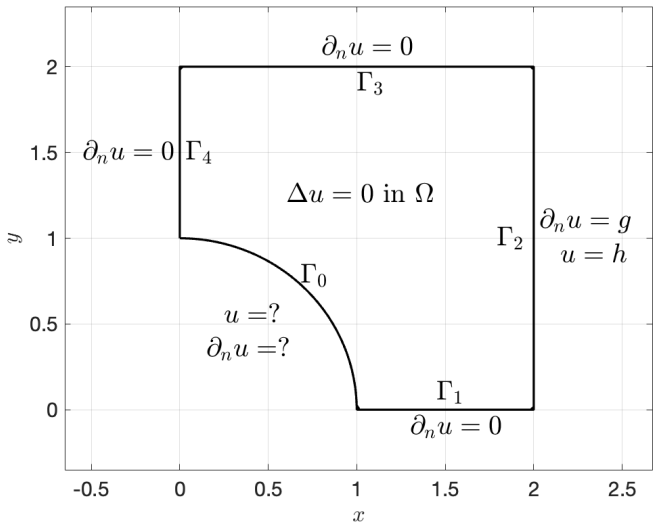

Figure 4.1: Cauchy problem formulation.

when $\underline{n}$ is the outward unit normal vector to the boundary $\partial\Omega = \bigcup_{i=1}^{4} \Gamma_i$. Remark that the boundary Γ_2 is overprescribed by supplying on it both the Dirichlet and Neumann data (4.2) and (4.3), whilst the presumed inaccessible boundary Γ_0 is underprescribed since no boundary condition/data is specified on it.

4.3 The boundary element method (BEM)

Compared to conventional domain discretisation techniques such as the finite-difference method (FDM) or the finite element method (FEM), the BEM has an advantage in inverse boundary value problems where the solution is sought mainly on the boundary. The BEM can also be combined with numerical optimization methods, making it possible to solve nonlinear and ill-posed inverse problems. Additionally, BEM can be used to accurately model irregular geometries and non-linear boundary conditions, which makes it a powerful numerical tool in various engineering and scientific applications such as electromagnetics, heat transfer and acoustics.

In this section, the Laplace equation (4.1) is being tackled in two dimensions using the BEM. This method divides the problem domain into two parts: the internal region Ω where the solution is unknown, and the boundary $\partial\Omega$ where some boundary conditions are prescribed. As described in Chapter 2 the Laplace equation (4.1) is transformed into

the boundary integral equation (2.8) via the application of Green's formula.

Applying the boundary conditions (4.2)–(4.4) in the integral equation (2.8) results in

$$\eta(\underline{p})u(\underline{p}) = \int_{\Gamma_0} G(\underline{p},\underline{p}')\frac{\partial u}{\partial n}(\underline{p}')\,ds(\underline{p}') + \int_{\Gamma_2} G(\underline{p},\underline{p}')\,g(\underline{p}')\,ds(\underline{p}') \\ - \int_{\partial\Omega\setminus\Gamma_2} u(\underline{p}')\frac{\partial G}{\partial n(\underline{p}')}(\underline{p},\underline{p}')ds(\underline{p}') - \int_{\Gamma_2} h(\underline{p}')\frac{\partial G}{\partial n(\underline{p}')}(\underline{p},\underline{p}')ds(\underline{p}'), \quad \underline{p}\in\overline{\Omega}. \quad (4.5)$$

We discretise anti-clockwise the boundary $\partial\Omega$ of the solution domain Ω into a series of $M = 7N$ straight-line boundary elements, $S_j = [\underline{p}_{j-1}, \underline{p}_j]$ for $j = \overline{1, M}$, where $\underline{p}_M = \underline{p}_0 = (0, 1)$. Over each small boundary element S_j, the Dirichlet data u is assumed constant and take its value at the midpoint (boundary element node), where $\underline{\tilde{p}}_j = (\underline{p}_j + \underline{p}_{j-1})/2$ for $j = \overline{1, M}$, namely,

$$u(\underline{p}) \approx u(\underline{\tilde{p}}_j) =: u_j, \quad \forall \underline{p}\in S_j, \quad j = \overline{1, 2N}\cup\overline{4N+1, M}. \quad (4.6)$$

$$u(\underline{p}) \approx u(\underline{\tilde{p}}_j) = h(\underline{\tilde{p}}_j) =: h_j, \quad \forall \underline{p}\in S_j, \quad j = \overline{2N+1, 4N}. \quad (4.7)$$

We also approximate the Neumann data by a piece-wise constant function as

$$\frac{\partial u}{\partial n}(\underline{p}) \approx \frac{\partial u}{\partial n}(\underline{\tilde{p}}_j) =: u'_j, \quad \forall \underline{p}\in S_j, \quad j = \overline{1, N}. \quad (4.8)$$

$$\frac{\partial u}{\partial n}(\underline{p}) \approx \frac{\partial u}{\partial n}(\underline{\tilde{p}}_j) = g(\underline{\tilde{p}}_j) =: g_j, \quad \forall \underline{p}\in S_j, \quad j = \overline{2N+1, 4N}. \quad (4.9)$$

With the approximations (4.6)−(4.9), the integral equation (4.5) yields

$$\eta(\underline{p})u(\underline{p}) = \sum_{j=1}^{N} A_j(\underline{p})u'_j + \sum_{j=2N+1}^{4N} A_j(\underline{p})g_j - \sum_{j=1}^{2N} B_j(\underline{p})u_j \\ - \sum_{j=2N+1}^{4N} B_j(\underline{p})h_j - \sum_{j=4N+1}^{M} B_j(\underline{p})u_j, \quad \underline{p}\in\overline{\Omega}, \quad (4.10)$$

where the coefficient matrices $A_j(\underline{p})$ and $B_j(\underline{p})$ defined in (2.8) are evaluated analytically, as described in Chapter 2.

Collocating equation (4.10) at the boundary element midpoint nodes $\underline{p} = \underline{\tilde{p}}_i$ for $i =$

$\overline{1,M}$, we obtain

$$\sum_{j=1}^{N} A_{ij}u'_j + \sum_{j=2N+1}^{4N} A_{ij}g_j + \sum_{j=1}^{2N} B_{ij}u_j + \sum_{j=2N+1}^{4N} B_{ij}h_j + \sum_{j=4N+1}^{M} B_{ij}u_j = 0,$$

$$i = \overline{1,M}, \quad (4.11)$$

where

$$A_{ij} = A_j(\underline{\tilde{p}}_i), \quad B_{ij} = -B_j(\underline{\tilde{p}}_i) - \frac{1}{2}\delta_{ij} \quad \text{for} \quad i = \overline{1,M}, \quad M = 7N. \quad (4.12)$$

The system of equations (4.11) can be rewritten as

$$\sum_{j=1}^{N} A_{ij}u'_j + \sum_{j=1}^{2N} B_{ij}u_j + \sum_{j=4N+1}^{M} B_{ij}u_j = -\sum_{j=2N+1}^{4N} A_{ij}g_j - \sum_{j=2N+1}^{4N} B_{ij}h_j,$$

$$i = \overline{1,M}. \quad (4.13)$$

In a matrix form, the system of equations (4.13) can be written as

$$K\underline{X} = \underline{b}, \quad (4.14)$$

where

$$K_{ij} = \begin{cases} A_{ij}, & \text{for } j = \overline{1,N}, \\ B_{i(j-N)}, & \text{for } j = \overline{N+1,3N}, \\ B_{i(j+N)}, & \text{for } j = \overline{3N+1,M-N}, \end{cases} \qquad i = \overline{1,M}, \quad (4.15)$$

and

$$X_j = \begin{cases} u'_j & \text{for } j = \overline{1,N}, \\ u_{j-N} & \text{for } j = \overline{N+1,3N}, \\ u_{j+N} & \text{for } j = \overline{3N+1,(M-N)}, \end{cases} \quad (4.16)$$

$$b_i = -\sum_{j=2N+1}^{4N} A_{ij}g_j - \sum_{j=2N+1}^{4N} B_{ij}h_j, \qquad i = \overline{1,M}. \quad (4.17)$$

This presents on over-determined system of $M = 7N$ linear equations with $6N$ unknowns.

The least-squares solution of this system is given by

$$\underline{X} = (K^\top K)^{-1} K^\top \underline{b}. \tag{4.18}$$

In practice, the data h in (4.2), where the assumed form of h will be specified in Section 4.4, is measured experimentally and therefore it is contaminated with noise. In our numerical implementation, we simulate this noise by perturbing the data h as

$$h^{noisy}(\mathrm{p}) = (1 + \rho\ \mathrm{p})h, \tag{4.19}$$

where p represents the percentage of noise and ρ is a random variable drawn from a uniform distribution in $[-1, 1]$. This noise induces a corresponding perturbation $\underline{b}^{noisy}$ in the vector $\underline{b}$ defined in (4.17), such that (4.18) recasts as

$$\underline{X}(\mathrm{p}) = (K^\top K)^{-1} K^\top \underline{b}^{noisy}(\mathrm{p}). \tag{4.20}$$

An alternative to the least-squares method is the singular value decomposition (SVD) method that allows for the analysis of the significance of data $\underline{b}$ through the spectrum of singular values of the matrix K. These singular values, when normalized by the largest singular value, manifest a pronounced decline. This observed pattern is a key factor in applying the truncated singular value decomposition (TSVD) method. The TSVD harnesses this information to truncate the spectrum of singular values appropriately. Retaining the larger singular values signifies holding onto the essential data characteristics, while the smaller ones, generally associated with noise and instabilities, are neglected.

4.4 Numerical results and discussion

We consider the Neumann data (4.3) with the function g given by (2.27) with $\epsilon = 0.1$. The Dirichlet data (4.2) on Γ_2 is fabricated from solving the direct problem given by equations (2.5), (4.1), (4.3) and (4.4), using the BEM with $N = 90$, i.e., with $M = 7N = 630$ boundary elements, as described in Section 2.2. However, we retain only one-third of

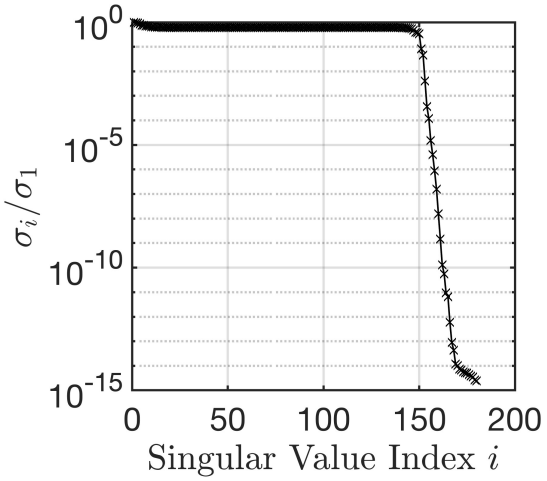

Figure 4.2: Normalized singular values of the matrix K.

this data on Γ_2 and solve the inverse problem given by equations (4.1)–(4.4) using the BEM with $N = 30$, i.e., with $M = 7N = 210$ boundary elements. Employing different discretisations for the direct and inverse problems is essential to prevent committing an inverse crime [65], where discretisation errors in the direct and inverse problems may cancel each other out, leading to an artificially accurate solution.

Since the Cauchy problem (4.1)–(4.4) is ill-posed, the resulting system of equations (4.13) is ill-conditioned. This is reflected in the behaviour of the normalised singular values of the matrix K (of dimension $M \times (M - N) = 210 \times 180$) displayed in Figure 4.2. The normalised singular values start off at a maximum value of 1 (since they are normalized by the largest singular value) and exhibit a steep decline, eventually spanning several orders of magnitude after about 150 singular values. This characteristic decay pattern is indicative of the presence of both significant and negligible components in the data structure. It suggests that the matrix from which these singular values are derived may be well-conditioned in the higher index range but becomes increasingly ill-conditioned as the index increases. Such a plot is crucial in the context of regularization, where one may truncate the spectrum to mitigate the effects of noise amplification, retaining only the most significant singular values for obtaining a stable solution [56].

We consider the case of a linear law (2.7), i.e. $f(u) = u$, and take $\lambda = -1$ such that

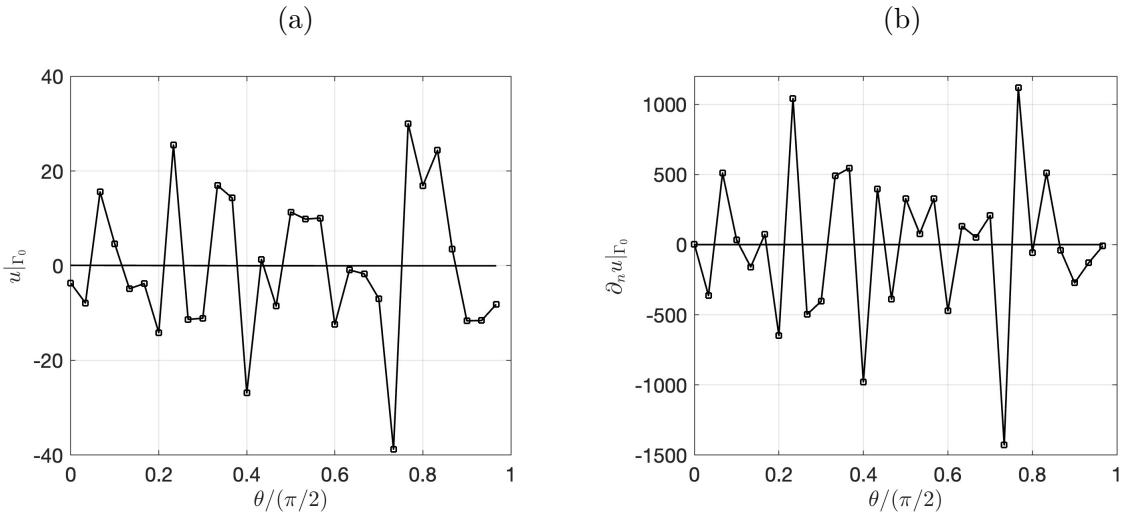

Figure 4.3: The BEM least-squares solution (4.18) for (a) u on Γ_0 and (b) the normal derivative $\partial_n u$ on Γ_0 of the inverse problem ($-\square-$) for p = 0 noiseless data in comparison with the BEM solutions of the direct linear problem (——).

the linear Robin boundary condition

$$\frac{\partial u}{\partial n} + u = 0 \quad \text{on } \Gamma_0, \tag{4.21}$$

applies to fabricate the Dirichlet data (4.2) by solving the direct well-posed linear problem given by equations (4.1), (4.3), (4.4) and (4.21). On solving this direct problem using the BEM it also provides the numerical values for u and $\partial_n u$ on Γ_0, which will be used to compare with the numerical results obtained by solving the inverse problem.

4.4.1 Results for p = 0 (no noise)

Figure 4.3 presents the numerical BEM least-squares solutions (4.18) of the inverse problem with noise-free data, i.e. p = 0 in (4.19), for the potential u and its normal derivative $\partial_n u$ on Γ_0. Both solutions exhibit highly oscillatory and unbounded behaviour, with $u|_{\Gamma_0}$ ranging from approximately -40 to 40 in Figure 4.3(a), and $\partial_n u|_{\Gamma_0}$ from -1500 to 1100 in Figure 4.3(b), indicating the high instability of the ill-posed inverse problem. An alternative to the least-squares method is the SVD. The numerically obtained results shown in Figure 4.4(a) for u on Γ_0 and in Figure 4.4(b) for $\partial_n u$ on Γ_0 reveal a significant amplification in oscillations compared to those in Figure 4.3, ranging from -2×10^6 to

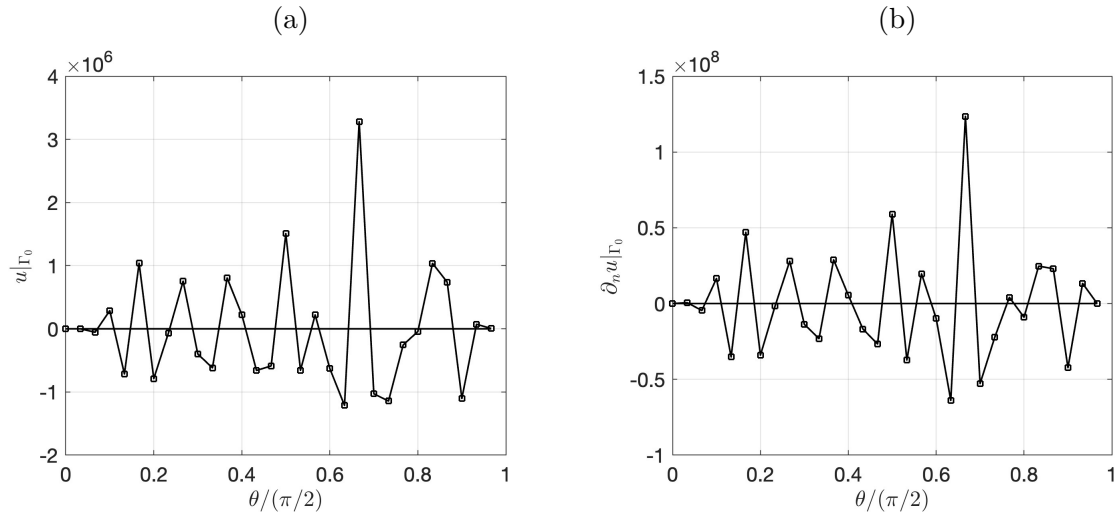

Figure 4.4: The BEM solutions for (a) u on Γ_0 and (b) the normal derivative $\partial_n u$ on Γ_0 of the inverse problem $(-\square-)$ for p = 0 noiseless data obtained using the untruncated SVD, in comparison with the BEM solutions of the direct linear problem (——).

4×10^6 and from -10^8 to 2×10^8, respectively.

From Figure 4.4, clearly the SVD has to be truncated by retaining only the most significant singular values to avoid the huge noise amplification. Figure 4.5 and Table 4.1 highlight the differences between the exact solutions (BEM solutions of the direct linear problem) and those obtained using the TSVD for different levels of truncation [57]. We observe that the lowest errors in the solution for $u|_{\Gamma_0}$ and $\partial_n u|_{\Gamma_0}$ are 0.0093 and 0.0436, respectively, achieved when the truncation index is set to 154 or 153. This truncation index produces the best balance between accuracy and stability, as confirmed by applying the L-curve method [16]. We illustrate the obtained L-curve in Figure 4.6, which identifies the corner region around 154 as the appropriate regularization parameter that mitigates the accuracy (residual) and stability (solution norm).

Figure 4.7 illustrates the TSVD solutions truncated at the indices 155, 154 and 153. Figures 4.7(a), 4.7(c) and 4.7(e) display the potential u on Γ_0 at these truncation indices, whilst Figures 4.7(b), 4.7(d) and 4.7(f) show the corresponding normal derivative $\partial_n u$ on Γ_0. The results for u on Γ_0 illustrated in Figures 4.7(a), 4.7(c) and 4.7(e) closely align with the direct linear problem solutions, demonstrating enhanced robustness against the ill-posedness of the inverse problem. On the other hand, the results for $\partial_n u$ on Γ_0 illustrated in Figures 4.7(b), 4.7(d) and 4.7(f), while remaining stable, are less accurate than those

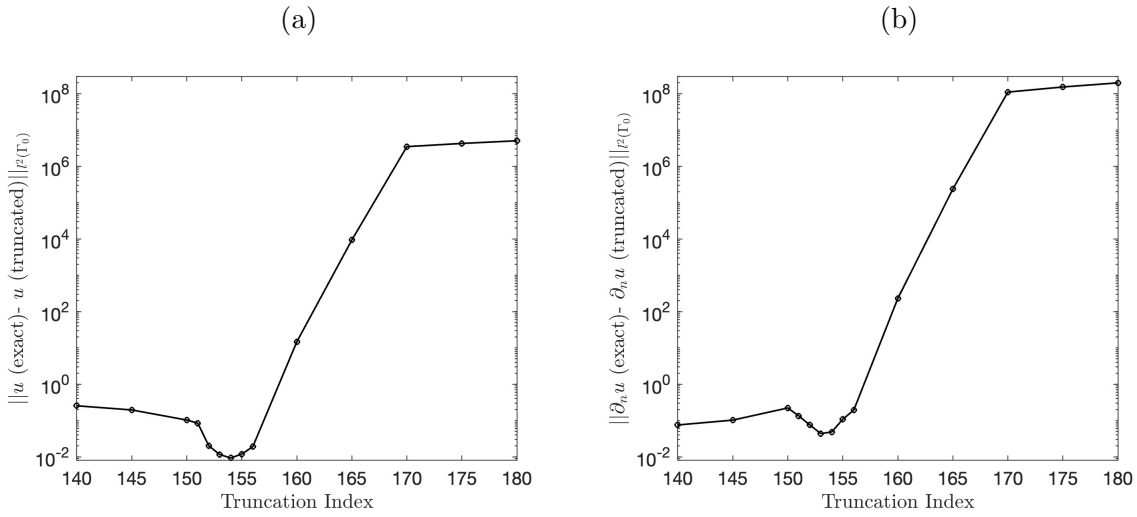

Figure 4.5: The discrete $\ell^2(\Gamma_0)$ norm of the differences between the exact (BEM solutions of the direct linear problem) and the TSVD solutions for (a) u on Γ_0 and (b) $\partial_n u$ on Γ_0, displayed at different truncation indices, for the inverse problem with p = 0 noiseless data.

Table 4.1: The error norms $E_u = ||u_{exact} - u_{num}||_{\ell^2(\Gamma_0)}$ and $E_{\partial_n u} = ||\partial_n u_{exact} - \partial_n u_{num}||_{\ell^2(\Gamma_0)}$ versus the number of retained singular values, for the inverse problem with p = 0 noiseless data.

SV	E_u	$E_{\partial_n u}$
160	14.707	230.1
155	0.0119	0.1084
154	0.0093	0.0484
153	0.0115	0.0436
152	0.0201	0.0755
151	0.0844	0.1337
150	0.1029	0.2215
145	0.1952	0.1024
140	0.2540	0.0752

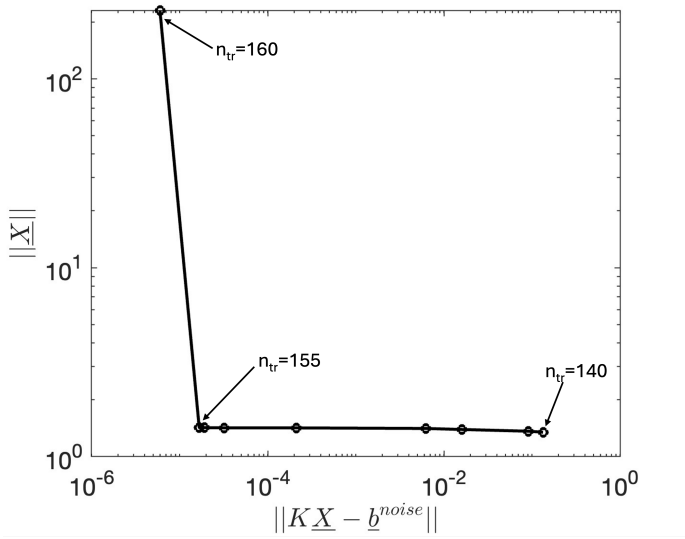

Figure 4.6: The L-curve illustrating the trade-off between the solution norm and the residual norm for the inverse problem with p = 0 noiseless data.

for u on Γ_0. This is somewhat expected since derivatives reduce the regularity of their belonging spaces compared to the function itself. Further improvement in stability and accuracy of $\partial_n u$ may be achieved through some a posteriori further regularization of the direct problem given by equations (4.1), (4.3), (4.4) and the just obtained $u|_{\Gamma_0}$, see [77].

Table 4.2 shows the values $u|_{\Gamma_0}$, $\partial_n u|_{\Gamma_0}$ and the ratio $(\partial_n u)/u$ on Γ_0. This ratio is one way to also estimate the corrosion coefficient λ, see (2.7), in case it is unknown. The numerical results presented in the last column of Table 4.2 shows that the ratio $(\partial_n u)/u$ on Γ_0 is quite far from the true value of $\lambda = -1$, see (2.5) and (4.4). In Chapter 5, we will elaborate more on the reconstruction of λ from the Cauchy data (4.2) and (4.3).

Finally, on plotting the obtained $u|_{\Gamma_0}$ versus $\partial_n u|_{\Gamma_0}$, as shown in Figure 4.8, we can simply infer the corrosion law $f(u)$, in case it is unknown, [36]. From this figure it can be seen that the exact linear law with $\lambda = -1$ given by $f(u) = -u$ is reasonably estimated by the TSVD truncated at $n_{tr} = 153$, whilst $n_{tr} = 154$ yields a multivalued unphysical relationship between the Dirichlet and Neumann data on Γ_0. More developed methods for estimating the nonlinear law $f(u)$ on Γ_0 from the Cauchy data on Γ_2 can be found in [80].

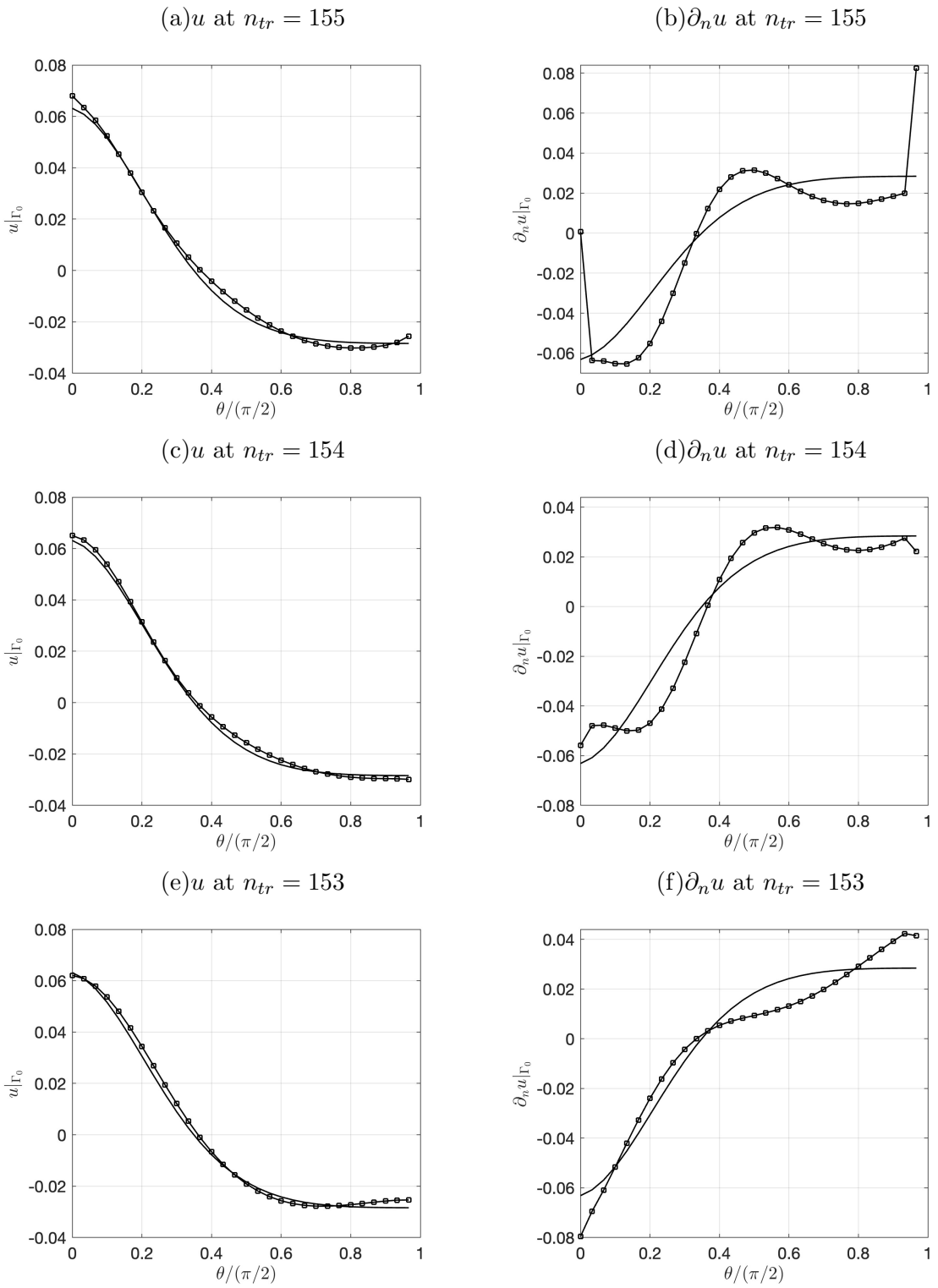

Figure 4.7: The BEM solutions ($-\square-$) for u and $\partial_n u$ on Γ_0 of the inverse problem with p = 0 noiseless data obtained using the TSVD truncated at 155, 154 and 153 singular values, in comparison with the BEM solutions of the direct linear problem (——).

Table 4.2: The values of $\lambda = (\partial_n u)/u$ for the inverse problem with p = 0 noiseless data, obtained using the TSVD with $n_{tr} = 154$. The first column indicates the position of the boundary element node $\tilde{p}_i$ on Γ_0 for $i = \overline{1, 30}$. The exact value is $\lambda = -1$.

i	u	$\partial_n u$	$(\partial_n u)/u$
1	-0.0299	0.0222	-0.7413
2	-0.0297	0.0276	-0.9298
3	-0.0296	0.0255	-0.8617
4	-0.0295	0.0239	-0.8101
5	-0.0294	0.0229	-0.7806
6	-0.0290	0.0226	-0.7766
7	-0.0285	0.0229	-0.8010
8	-0.0278	0.0238	-0.8558
9	-0.0268	0.0253	-0.9422
10	-0.0256	0.0272	-1.0600
11	-0.0242	0.0291	-1.2064
12	-0.0224	0.0309	-1.3758
13	-0.0204	0.0319	-1.5586
14	-0.0182	0.0316	-1.7404
15	-0.0156	0.0297	-1.9023
16	-0.0127	0.0257	-2.0199
17	-0.0094	0.0194	-2.0585
18	-0.0056	0.0109	-1.9320
19	-0.0012	0.0005	-0.4228
20	0.0039	-0.0109	-2.8277
21	0.0097	-0.0224	-2.3088
22	0.0163	-0.0329	-2.0140
23	0.0236	-0.0413	-1.7476
24	0.0314	-0.0470	-1.4949
25	0.0394	-0.0497	-1.2632
26	0.0470	-0.0500	-1.0637
27	0.0539	-0.0489	-0.9068
28	0.0595	-0.0477	-0.8020
29	0.0633	-0.0479	-0.7570
30	0.0651	-0.0559	-0.8584

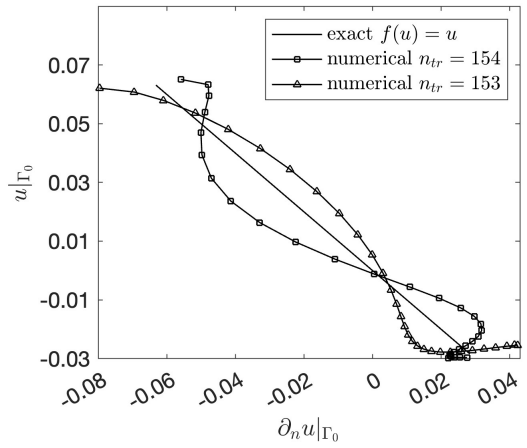

Figure 4.8: The numerical corrosion law obtained using the TSVD with $n_{tr} = 154$ and 153, no noise, in comparison with the exact expression $\lambda f(u) = -u$.

4.4.2 Results for p = 1%, 3% and 5% noise

In this subsection, we analyse the impact on the numerical results for $u|_{\Gamma_0}$ and $\partial_n u|_{\Gamma_0}$ obtained when adding noise levels of p = 1%, 3% and 5% to the observed data on Γ_2, as defined in (4.19).

Figures 4.9 and 4.10 present the numerical BEM least-squares and untruncated SVD solutions for the inverse problem with noise levels of p = 1%, 3% and 5% compared to the direct problem BEM solutions. As the noise level increases from 1% to 3%, the amplitude of oscillations in both u and $\partial_n u$ approximately doubles, and from 3% to 5%, it doubles again, resulting in an overall fourfold increase. As previously obtained by comparing Figures 4.3 and 4.4 for p = 0 noiseless data, the results obtained using the untruncated SVD are much more unstable than those obtained using the least-squares method. The escalation in oscillatory behaviour with rising noise levels underlines the increasing instability of the ill-posed inverse problem.

Figure 4.11 demonstrates that a truncation index of 152 in the TSVD yields the smallest errors for $u|_{\Gamma_0}$, as shown in Figure 4.11(a), and for $\partial_n u|_{\Gamma_0}$, as depicted in Figure 4.11(b). The error norms for $u|_{\Gamma_0}$ are 0.0290, 0.0595 and 0.0931 for noise levels of p = 1%, 3% and 5% respectively, while those for $\partial_n u|_{\Gamma_0}$ are 0.0774, 0.1140 and 0.1689 for the same noise levels, with 152 representing the minimum error, see Table 4.3. This truncation index provides an effective balance between accuracy and stability across all noise levels

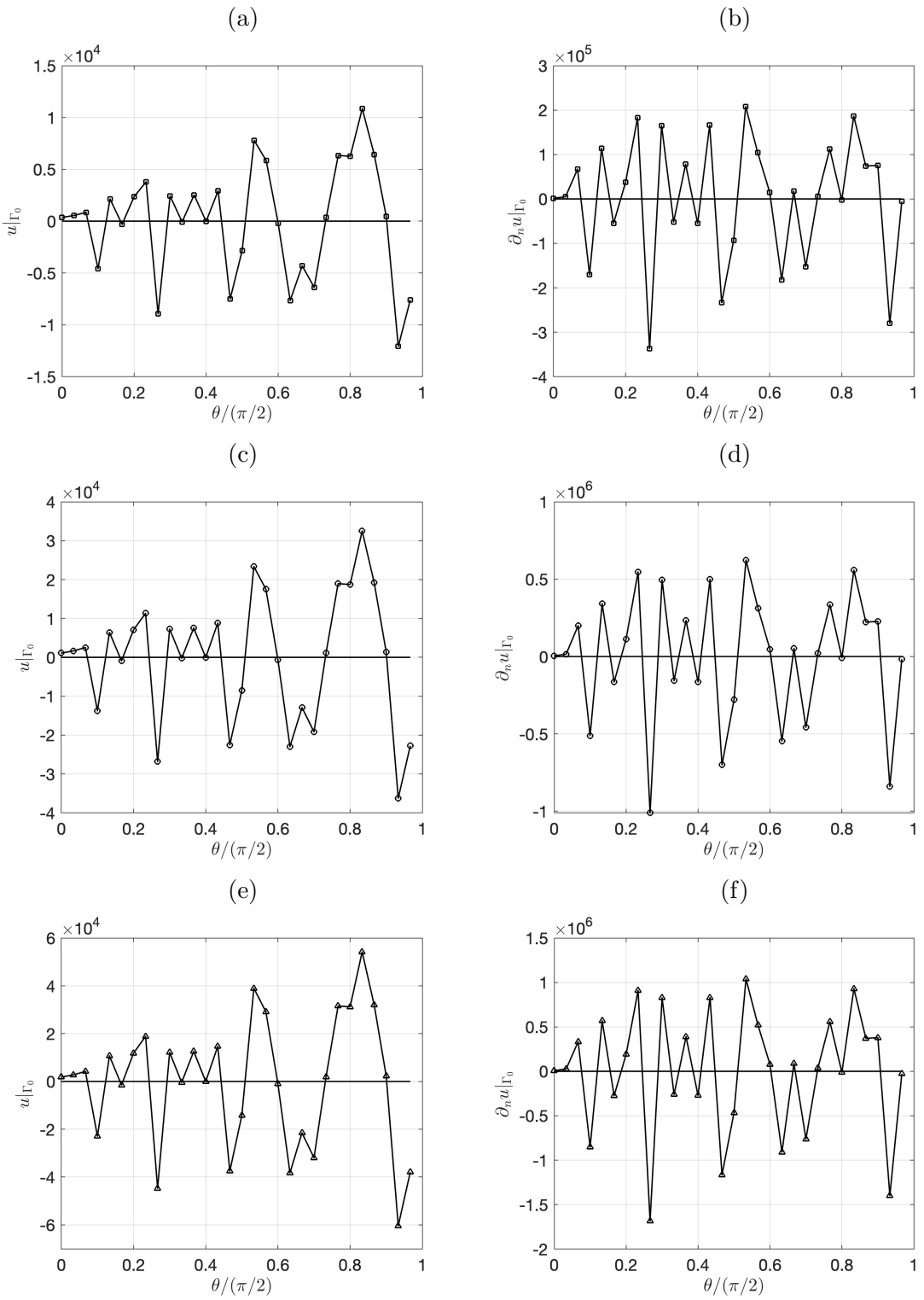

Figure 4.9: The BEM least-squares solution (4.18) for (a) u on Γ_0 and (b) $\partial_n u$ on Γ_0 of the inverse problem, shown with $(-\square-)$, $(-\circ-)$ and $(-\Delta-)$ for p = 1%, 3% and 5% noise, respectively, compared with the BEM solutions of the direct linear problem (——).

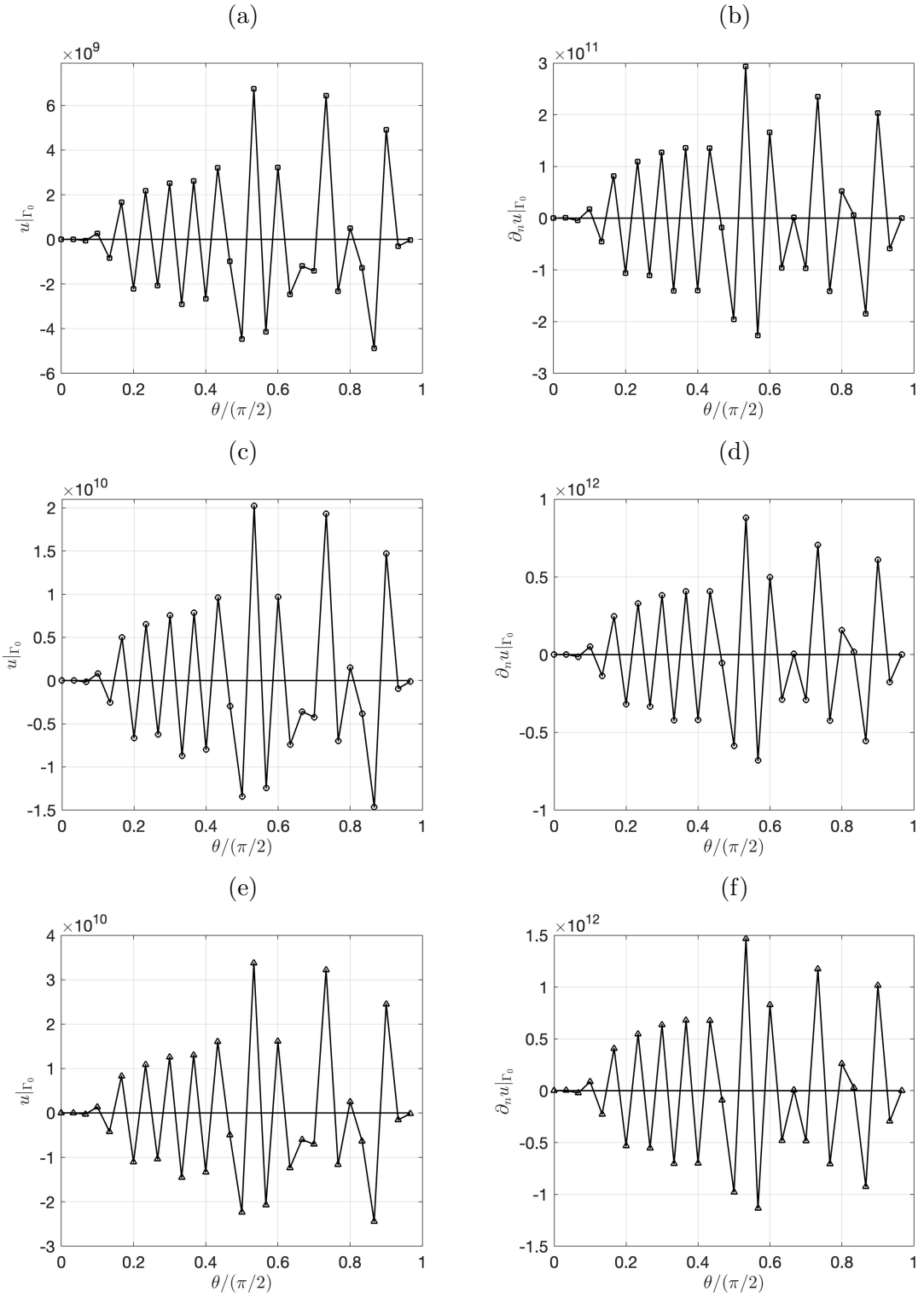

Figure 4.10: The BEM solutions for (a) u on Γ_0 and (b) $\partial_n u$ on Γ_0 of the inverse problem, shown with $(-\square-)$, $(-\circ-)$ and $(-\Delta-)$ for p = $1\%, 3\%$ and 5% noise, respectively, obtained using the untruncated SVD in comparison with the BEM solutions of the direct linear problem (——).

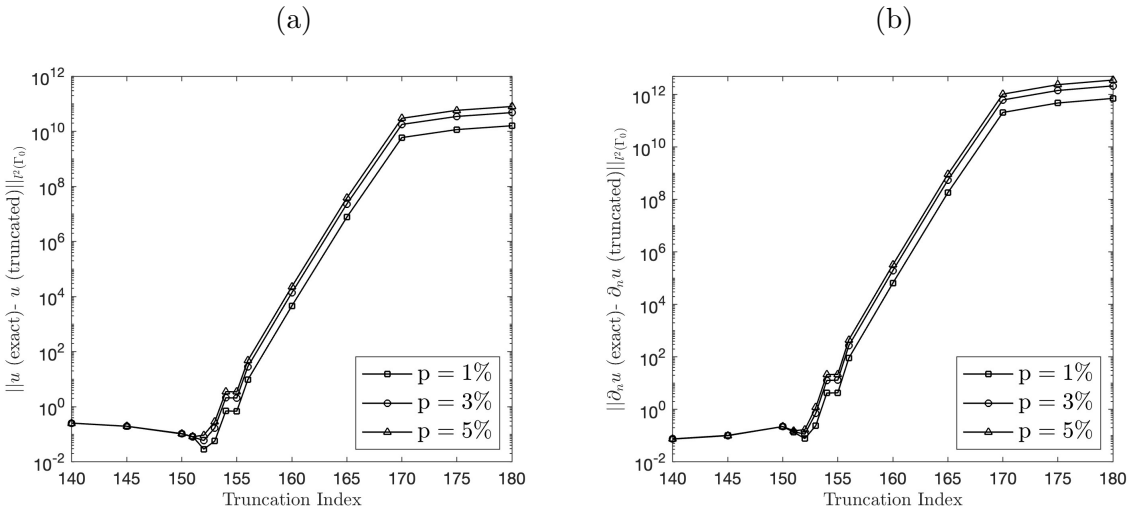

Figure 4.11: The discrete $\ell^2(\Gamma_0)$ norm of the differences between the exact (BEM solutions of the direct linear problem) and the TSVD solutions for (a) u on Γ_0 and (b) $\partial_n u$ on Γ_0, displayed at different truncation indices, for the inverse problem with various p = 1%, 3% and 5% noise.

examined. Additionally, the L-curve method corroborates these findings by identifying the truncation number 152 as its corner, as illustrated in Figure 4.12.

Table 4.3: Error norms E_u and $E_{\partial_n u}$ versus the number of retained singular values, for the inverse problem with p = 1%, 3% and 5% noisy data.

SV	p=1%		p=3%		p=5%	
	E_u	$E_{\partial_n u}$	E_u	$E_{\partial_n u}$	E_u	$E_{\partial_n u}$
155	0.6928	4.2875	2.0880	12.991	3.4833	21.695
154	0.7024	4.1989	2.1180	12.686	3.5337	21.174
153	0.0570	0.2392	0.1709	0.7130	0.2854	1.1889
152	0.0290	0.0774	0.0595	0.1140	0.0931	0.1689
151	0.0822	0.1348	0.0803	0.1413	0.0819	0.1529
150	0.1035	0.2216	0.1050	0.2218	0.1065	0.2221
145	0.1954	0.1018	0.1958	0.1007	0.1963	0.0995
140	0.2542	0.0750	0.2546	0.0744	0.2550	0.0739

Figure 4.13 offers a visual comparison of the TSVD solutions at truncation indices of 153, 152 and 151, demonstrating its ability to maintain stability across these noise levels. At the truncation index of 153, as shown in Figure 4.13(a) for $u|_{\Gamma_0}$ and 4.13(b) for $\partial_n u|_{\Gamma_0}$, the solutions exhibit significant oscillations and instability, particularly as the noise level increases, reflecting the ill-conditioned nature of the system of equations (4.14) that amplifies the noise effects. However, truncating the singular values at 152 or 151 effectively reduce these oscillations, producing smoother and more stable solutions

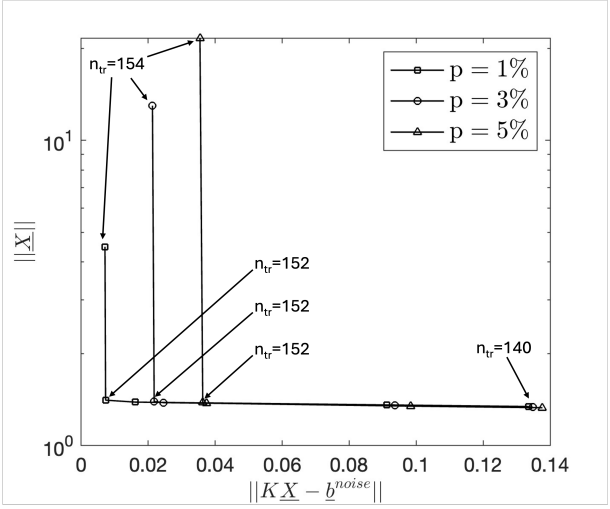

Figure 4.12: The L-curve illustrating the trade-off between the solution norm and the residual norm for the inverse problem with p = 1%, 3% and 5% noisy data.

that closely align with the direct linear problem's BEM solution. This highlights the effectiveness of the TSVD method in stabilizing the inverse problem.

The MATLAB computational program which yielded the numerical results presented and discussed in this section is included in Section A.4 of Appendix A.

Although not illustrated, it is reported that similar results have been obtained when the nonlinear boundary condition (2.5) with $\lambda = -1$ on Γ_0 given by

$$\frac{\partial u}{\partial n} + 2\sinh\left(\frac{u}{2}\right) = 0 \quad \text{on } \Gamma_0 \tag{4.22}$$

was used to fabricate the Dirichlet data (4.2) by solving the direct nonlinear problem given by equations (4.1), (4.3), (4.4) and (4.22).

4.5 Conclusions

This chapter has presented a comprehensive investigation into the application of the BEM for solving the linear Cauchy problem for the Laplace equation, through a rigorous comparison of numerical solutions with both noiseless and noisy data (fabricated, i.e. numerically simulated both from the boundary condition (2.5)). The evaluation of results demonstrates that the ill-posed problem poses significant challenges, particularly regarding stability and accuracy. The analysis indicates that the ill-posed nature of these inverse

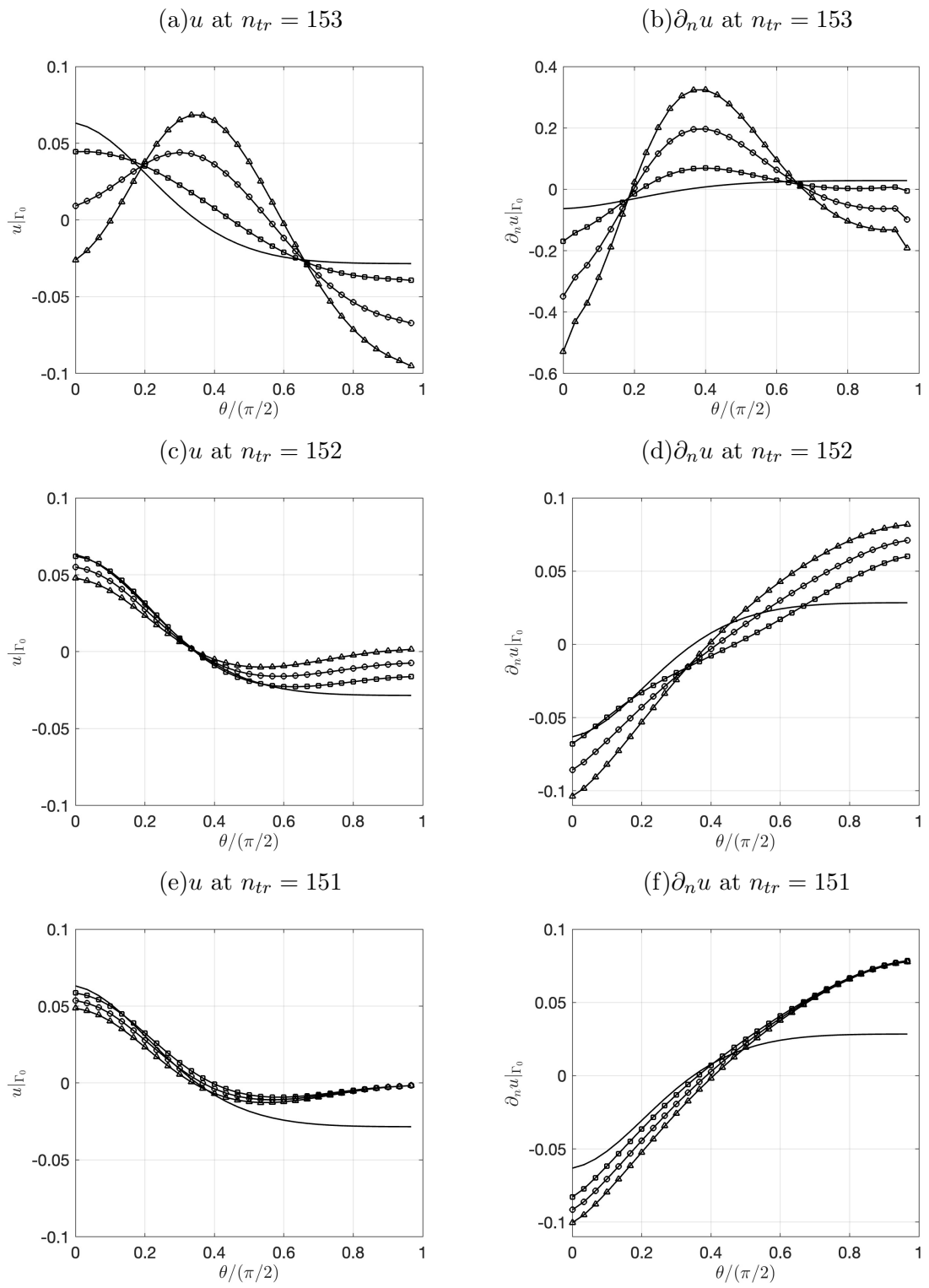

Figure 4.13: The BEM solutions for u and $\partial_n u$ on Γ_0 of the inverse problem, shown with $(-\square-)$, $(-\circ-)$ and $(-\Delta-)$ for p = 1%, 3% and 5% noise, respectively, obtained using the TSVD truncated at 153, 152 and 151 singular values, in comparison with the BEM solutions of the direct linear problem (——).

problems complicates the solution process, often resulting in substantial deviations from expected outcomes, especially when noise is introduced into the system.

The goal of this chapter was to recover the missing potential and flux values on an inaccessible boundary using over-prescribed data on an accessible boundary. Because this problem is severely ill-posed, standard least-squares and SVD methods produced physically meaningless, oscillatory results. To stabilize the system, the TSVD was implemented and tested against noise levels of 0, 1%, 3% and 5%. The primary contribution of this chapter is the validation of the TSVD as a robust regularization strategy for corrosion data recovery. By utilizing singular value spectrum analysis, this work provided a data-driven criterion for selecting the truncation parameter rather than relying on arbitrary heuristic tuning. Furthermore, the study established that while the normal derivative is significantly more sensitive to noise than the potential, the TSVD can effectively transform these instabilities into accurate solutions with error margins as low as 1%. The MATLAB computational program for this chapter is included in Section A.4 of Appendix A.

Chapter 5

Reconstruction of the space-dependent corrosion coefficient

Continuing from the previous chapter, the present chapter addresses the inverse problem of determining the space-dependent corrosion coefficient λ on the boundary Γ_0 using over-prescribed measured data on Γ_2. The BEM is employed as a direct solver in the framework of an iterative minimization procedure that is used to recover the unknown corrosion coefficient. The method's efficacy is demonstrated through numerical results, showing its potential for practical use in fields such as electrochemistry and material science, where the stable recovery of the coefficient of corrosion is paramount.

5.1 Introduction

This chapter focuses on an inverse problem involving the Laplace equation in a two-dimensional domain, see Figure 5.1, where the goal is to determine the space-dependent corrosion coefficient λ on the boundary Γ_0. To achieve this, we utilize the Cauchy data consisting of u and its normal derivative $\partial_n u$ on the accessible part Γ_2 of the boundary. The function λ is key to understanding the relationship between the flux $\partial_n u$ and the potential u on the portion Γ_0 of the boundary that is subjected to a corrosion attack. Therefore, accurately determining this function is essential for addressing practical problems related to corrosion, structural integrity and material property characterization.

Many theoretical and numerical studies, e.g., [28, 30, 37, 96], have been devoted to solve this inverse coefficient identification problem in case of linear corrosion given by equation (5.5). In [96], for instance, in the context of heat conduction, the space-dependent boundary heat transfer coefficient was estimated from Cauchy boundary data by solving the corresponding linear Cauchy problem (as described in Chapter 4) using the meshless method of fundamental solutions (MFS) and simply dividing the obtained flux by the obtained potential on Γ_0, see (5.7).

In this chapter, to discretise the inverse boundary coefficient problem, we employ the BEM, which is particularly suited for problems involving boundary conditions and is effective in reducing the dimensionality of the problem, leading to computational efficiency and ease of discretisation.

The following sections of this chapter are structured as follows. Section 5.2 outlines the mathematical formulation of the inverse corrosion coefficient problem. Section 5.3 details the computational BEM. Section 5.4 represents the nonlinear minimization method, whilst Section 5.5 provides numerical results and discussions for both linear and nonlinear boundary conditions in noise-free and noisy data scenarios. Finally, Section 5.6 concludes with final remarks.

5.2 Mathematical formulation

In this section, we present the mathematical formulation of the inverse problem for identifying the coefficient λ along with the potential u satisfying the Laplace equation in Ω with boundary conditions, see Figure 5.1, given by

$$\begin{cases} \nabla^2 u = 0 & \text{in } \Omega, \quad (5.1) \\ \dfrac{\partial u}{\partial n} = \lambda(\theta) f(u) & \text{on } \Gamma_0 = \left\{ (\cos(\theta), \sin(\theta)) |\ \theta \in \left[0, \dfrac{\pi}{2}\right] \right\}, \quad (5.2) \\ \dfrac{\partial u}{\partial n} = g & \text{on } \partial\Omega \setminus \Gamma_0, \quad (5.3) \\ u = h & \text{on } \Gamma_2, \quad (5.4) \end{cases}$$

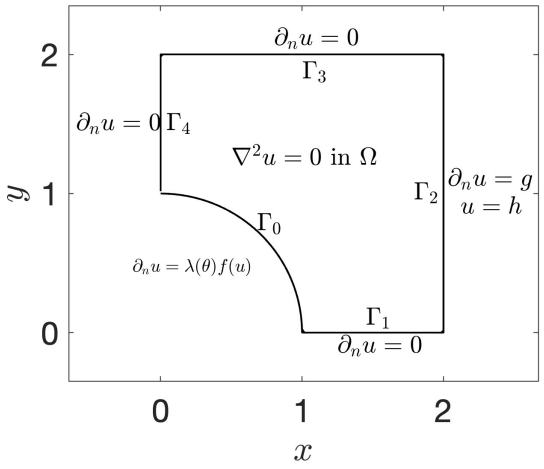

Figure 5.1: The mathematical formulation for determining the coefficient of corrosion $\lambda(\theta)$ on Γ_0.

where g and h are the known boundary data for the flux on $\partial\Omega\backslash\Gamma_0$ and potential on Γ_2, respectively. Note that according to Figure 5.1 and equation (5.3), we have that g vanishes on $\Gamma_1 \cup \Gamma_3 \cup \Gamma_4$ but we take $g|_{\Gamma_2} \not\equiv 0$. In (5.2), the law of corrosion $f(u)$ is known and given by

$$f(u) = u_a - u \qquad \text{in the linear Newton's law case of convection,} \tag{5.5}$$

where u_a is the ambient potential, which is constant and taken, for simplicity, to be zero, or

$$f(u) = -2\sinh\left(\frac{u}{2}\right) \qquad \text{in the nonlinear Butler-Volmer case.} \tag{5.6}$$

Under these laws, the unknown Robin coefficient of corrosion λ is sought as a non-negative continuous function, not identically equal to zero.

Various uniqueness results and stability estimates for inverse corrosion coefficient problems under the linear Robin boundary condition

$$\frac{\partial u}{\partial n} + \lambda u = 0 \quad \text{on } \Gamma_0. \tag{5.7}$$

are provided in [4, 26, 27, 29, 36, 37, 61, 90].

5.3 The boundary element method (BEM)

As described in the previous chapters, multiplying the Laplace equation (5.1) with the fundamental solution (2.9), integrating using the Green's formula and applying the boundary conditions (5.2)–(5.3) yields

$$\eta(\underline{p})u(\underline{p}) = \int_{\Gamma_0} G(\underline{p};\underline{p}')\ \lambda(\underline{p}')\ f(u(\underline{p}'))\ ds(\underline{p}') + \int_{\Gamma_2} G(\underline{p};\underline{p}')\ g(\underline{p}')\ ds(\underline{p}') - \int_{\partial\Omega} u(\underline{p}')\ \frac{\partial G}{\partial n(\underline{p}')}(\underline{p};\underline{p}')\ ds(\underline{p}'), \quad \underline{p} \in \overline{\Omega}. \tag{5.8}$$

By discretising the whole boundary $\partial\Omega$ into $M = 7N$ elements, the integral equation applied at the boundary element nodes $\underline{\tilde{p}}_i \in \partial\Omega$ for $i = \overline{1, M}$ is converted into the following system of equations, see also (4.13):

$$\sum_{j=1}^{N} A_{ij}\ \lambda_j\ f(u_j) + \sum_{j=1}^{M} B_{ij}\ u_j = -\sum_{j=2N+1}^{4N} A_{ij}\ g_j =: b_i, \quad i = \overline{1, M}, \tag{5.9}$$

where $\lambda_j = \lambda\left(\tilde{p}_j\right)$ for $j = \overline{1, M}$.

In case of the linear law (5.5) with $u_a = 0$ given by $f(u) = -u$, equation (5.9) simplifies as

$$\sum_{j=1}^{N} \left(B_{ij} - A_{ij}\ \lambda_j\right)\ u_j + \sum_{j=N+1}^{M} B_{ij}\ u_j = b_i, \quad i = \overline{1, M}, \tag{5.10}$$

whilst in the case of the nonlinear law (5.6) it recasts as

$$\sum_{j=1}^{N} \left[B_{ij}\ u_j - 2\ A_{ij}\ \lambda_j\ \sinh\left(\frac{u_j}{2}\right)\right] + \sum_{j=N+1}^{M} B_{ij}\ u_j = b_i, \quad i = \overline{1, M}. \tag{5.11}$$

5.4 Nonlinear constrained minimization

The corrosion coefficient $\lambda(\theta)$ on Γ_0 must be identified using the available Cauchy data pair (h, g) on Γ_2. To achieve this, the inverse problem is formulated as a constrained minimization and solved numerically in MATLAB using the toolbox routine *lsqnonlin* based

on the trust-reflective region algorithm [38, 39]. The primary objective is to minimize the discrepancy between the computed solution and the measured boundary potential data h on Γ_2. According to the analyses [26, 27, 61] at least for the linear law (5.5), the solution (u, λ) to the inverse problem (5.1), (5.3), (5.4) and (5.7) is unique. We note that for the nonlinear law of corrosion (5.6) no such uniqueness result is known yet.

Starting from an arbitrary initial guess $\lambda^{(0)}$, the goal is to iteratively reduce the difference between the computed solution and the measured data h on Γ_2. As such, the least-squares objective function to be minimized is expressed as:

$$F(\lambda) = \|u(\lambda) - h\|^2_{L^2(\Gamma_2)} = \int_{\Gamma_2} \left(u(\lambda)|_{\Gamma_2} - h\right)^2 \, d\Gamma_2. \tag{5.12}$$

In discretised form, (5.12) can be written as:

$$F(\underline{\lambda}) = \frac{2}{2N} \sum_{i=2N+1}^{4N} \left(u_i(\underline{\lambda}) - h_i\right)^2, \tag{5.13}$$

where $2N$ is the number of boundary elements discretising the vertical boundary $\Gamma_2 = \{(2, y) | y \in [0, 2]\}$, see Figure 5.1, and u_i and h_i denote the values of u and h at the boundary element midpoint nodes $\tilde{p_i}$ for $i = \overline{2N+1, 4N}$. In (5.13), $u|_{\Gamma_2}$ has been expressed in terms of λ by inverting (5.10) (or (5.11)). To ensure stability, especially when the measured data (5.4) is contaminated with noise, a regularization term $R(\lambda)$ is often added to the least-squares functional (5.12). Then, the regularized objective function becomes:

$$F_\mu(\lambda) = F(\lambda) + \mu R(\lambda), \tag{5.14}$$

where $\mu \geq 0$ is a regularization parameter to be prescribed, and $R(\lambda)$ is a regularization term, e.g.,

$$R(\lambda) = \int_0^{\pi/2} \left(\lambda(\theta) - \lambda^{(0)}(\theta)\right)^2 d\theta \quad \text{in case } \lambda \in C\left(\overline{\Gamma}_0\right), \tag{5.15}$$

or

$$R(\lambda) = \int_0^{\pi/2} \left(\lambda'(\theta)\right)^2 d\theta \quad \text{in case } \lambda \in C^1\left(\overline{\Gamma}_0\right). \tag{5.16}$$

In discrete form, (5.15) and (5.16) can be written as

$$R(\lambda) = \frac{\pi}{2N} \sum_{i=1}^{N} \left(\lambda(\tilde{\theta}_i) - \lambda^{(0)}(\tilde{\theta}_i) \right)^2, \tag{5.17}$$

and

$$R(\lambda) = \frac{\pi}{2N} \sum_{i=2}^{N} \left(\lambda(\tilde{\theta}_i) - \lambda(\tilde{\theta}_{i-1}) \right)^2, \tag{5.18}$$

respectively, where $\tilde{\theta}_i = \frac{\pi}{2}(1 - \frac{2i-1}{2N})$ for $i = \overline{1, N}$, noting that N constant boundary elements are employed to discretise the boundary Γ_0. Here, (5.17) represents zeroth-order Tikhonov regularization, while (5.18) represents first-order Tikhonov regularization.

5.5 Numerical results and discussion

We take the function g on Γ_2 given by equation (2.27) corresponding to an imposed discontinuous current flux in an electrical impedance tomography (EIT) experiment and measure the resulting potential u on Γ_2, as given by equation (5.4). In this section, this 'measured' data is numerically simulated by solving the direct problem with

$$\lambda(\theta) = \sin(\theta), \qquad \theta \in \left[0, \frac{\pi}{2}\right], \tag{5.19}$$

using the BEM with $M = 7N = 630$ boundary elements, as described in Section 2.4.3.2 (while noting the change of sign in λ).

In the inverse problem (5.1)–(5.4), we utilize only a portion of the data, specifically one-third, on Γ_2 for the BEM with $N = 30$ resulting in $M = 7N = 210$ boundary elements. This common numerical modelling approach of taking a different mesh for solving the direct and inverse problems when utilizing the same discretisation method not only prevents committing an inverse crime [65], but it also introduces some numerical noise into the data (5.4).

Simple physical bounds on the variables are prescribed as

$$0 = \lambda_{min} \leq \lambda(\tilde{\theta}_i) \leq \lambda_{max} = 1, \quad i = \overline{1, N}. \tag{5.20}$$

The initial guess is the straight line $\lambda^{(0)}(\theta) = \frac{\theta}{\pi/2}$ for $\theta \in \left[0, \frac{\pi}{2}\right]$. The results obtained from the constrained minimization problem described in Section 5.4 provide the identified coefficient of corrosion $\lambda(\theta)$ along Γ_0. The accuracy of this identification is influenced by the initial guess $\lambda^{(0)}$, the discretization of the boundary into M boundary elements, and the percentage of noise p, see (4.19), in the measured data (5.4). The tolerance parameters used by the `lsqnonlin` toolbox are setup as Function Tolerance= 10^{-20}, Step Tolerance= 10^{-20} and Optimality Tolerance= 10^{-15}.

5.5.1 Example 1: Linear boundary condition (LBC) (5.7) on Γ_0

5.5.1.1 Results for p = 0 (no noise)

Figure 5.2 presents the results obtained with the zero$^{\text{th}}$-order regularization (5.17) added to the minimizing functional (5.14). Figure 5.2(a) provides valuable insight into the convergence of (5.14) with R given by (5.17) with the number of iterations of the *lsqnonlin* constrained minimization process for different values of the regularization parameter μ. The iterative process converges quickly to a minimum small value after which it remains stationary until the imposed tolerances of the `lsqnonlin` routine are satisfied. It highlights the importance of selecting an appropriate μ to ensure that the optimization algorithm not only converges effectively but also leads to an accurate and stable solution. Figure 5.2(b) provides a quantitative assessment of the accuracy of the identified coefficient $\lambda(\theta)$ by plotting the ℓ_2–norm of the error $\|\lambda_{exact} - \lambda_{num}\|$ against the regularization parameter μ. The minimum of this error is obtained for $\mu = 3 \times 10^{-6}$, which is the optimal regularization parameter. This corresponds to the point where the regularization smooths the solution enough to mitigate noise without over-smoothing and losing important details.

The results for λ with no regularization, i.e. $\mu = 0$, in Figure 5.2(c) jumps between the upper and lower bounds in (5.20) as a manifestation of the instability of the nonlinear non-convex least-squares functional (5.12). On the other hand, the results obtained with $\mu = 10^{-5}$ in Figure 5.2(c) show a good balance between stability and accuracy; however, the obtained solution is still far from the exact solution. Better accuracy is obtained

by employing the first-order regularization (5.18) in the objective function (5.14), as illustrated in Figure 5.3.

5.5.1.2 Results for p = 1% (noise)

In this section, we add p = 1% noise to the data (5.4), as given by (4.19), to test the stability of the method to recover the coefficient $\lambda(\theta)$ under noisy conditions. Noise makes the problem more difficult, but by choosing an appropriate regularization parameter μ, we can still get reliable, i.e. stable and accurate, results. Based on the previous results obtained in case p = 0, we further employ only the first-order regularization.

Figure 5.4(a) shows how the minimization process converges with different values of μ when p = 1% noise is present. It can be seen that the objective function (5.14) attains stationary values that are higher than those in Figure 5.4(a) due to the presence of noise in the data (4.19); however, it takes fewer iterations to achieve convergence in case of noisy data than in case of error-free data.

Figure 5.4(b) shows how the ℓ_2-error between the exact and numerical corrosion coefficient changes as μ varies when p = 1% noise is present. From this figure it can be seen that the minimum of the error occurs around $\mu = 8 \times 10^{-4}$. This point represents the best balance, where the regularization smooths out the noise without losing too much detail. If μ is too large, the error starts to increase again because the solution becomes too smooth, missing important features.

Figure 5.4(c) compares the exact solution with the numerical solution for $\lambda(\theta)$ on the boundary Γ_0. The noise causes some differences between the exact and numerical solutions, but the overall pattern is still captured by the numerical method. This shows that the method is fairly stable with noise, provided that the regularization parameter μ is chosen appropriately to control the balance between noise reduction and solution's accuracy.

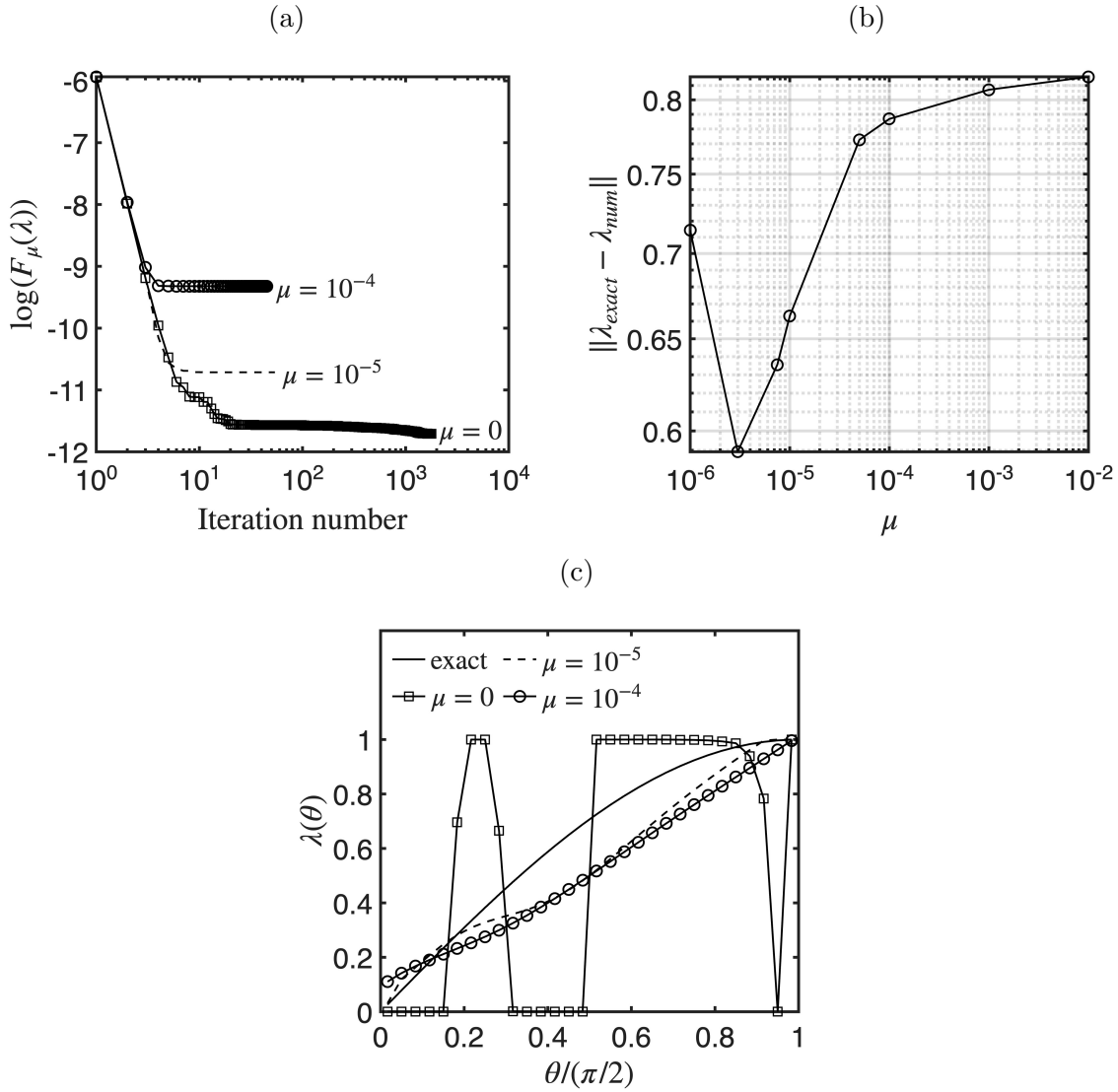

Figure 5.2: Determining the corrosion coefficient $\lambda(\theta)$ on the boundary Γ_0 using the LBC (5.7) for different values of μ with zero-order regularization (5.17), in case of no noise p = 0. (a) The convergence of the objective function with the number of iterations, (b) the norm of the difference between the exact and numerical solutions of λ for various values of μ, and (c) the comparison between the exact and numerical solutions for the coefficient λ on Γ_0.

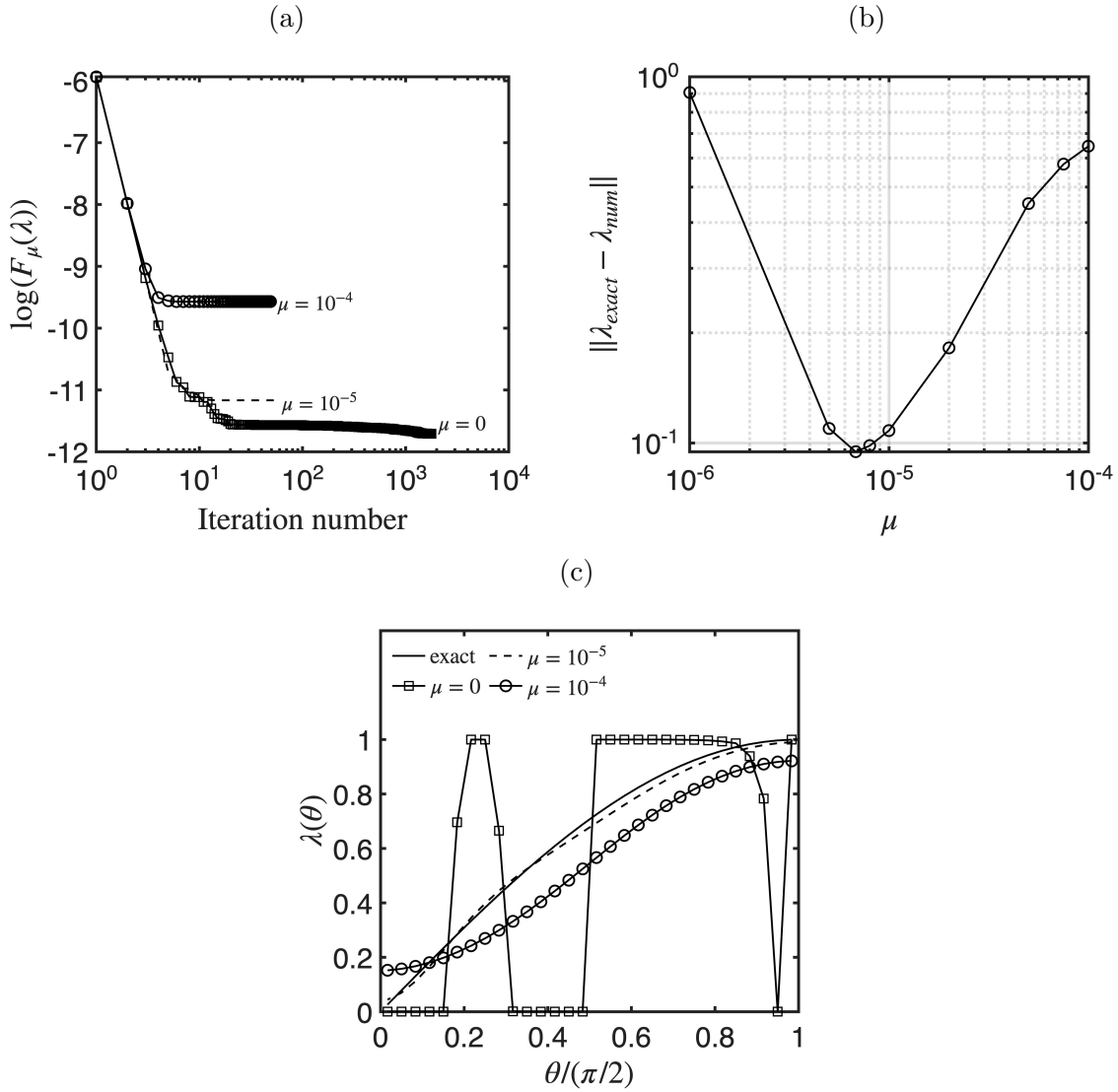

Figure 5.3: Determining the corrosion coefficient $\lambda(\theta)$ on the boundary Γ_0 using the LBC (5.7) for different values of μ with first-order regularization (5.18), in case of no noise p = 0. (a) The convergence of the objective function with the number of iterations, (b) the norm of the difference between the exact and numerical solutions of λ for various values of μ, and (c) the comparison between the exact and numerical solutions for the coefficient λ on Γ_0.

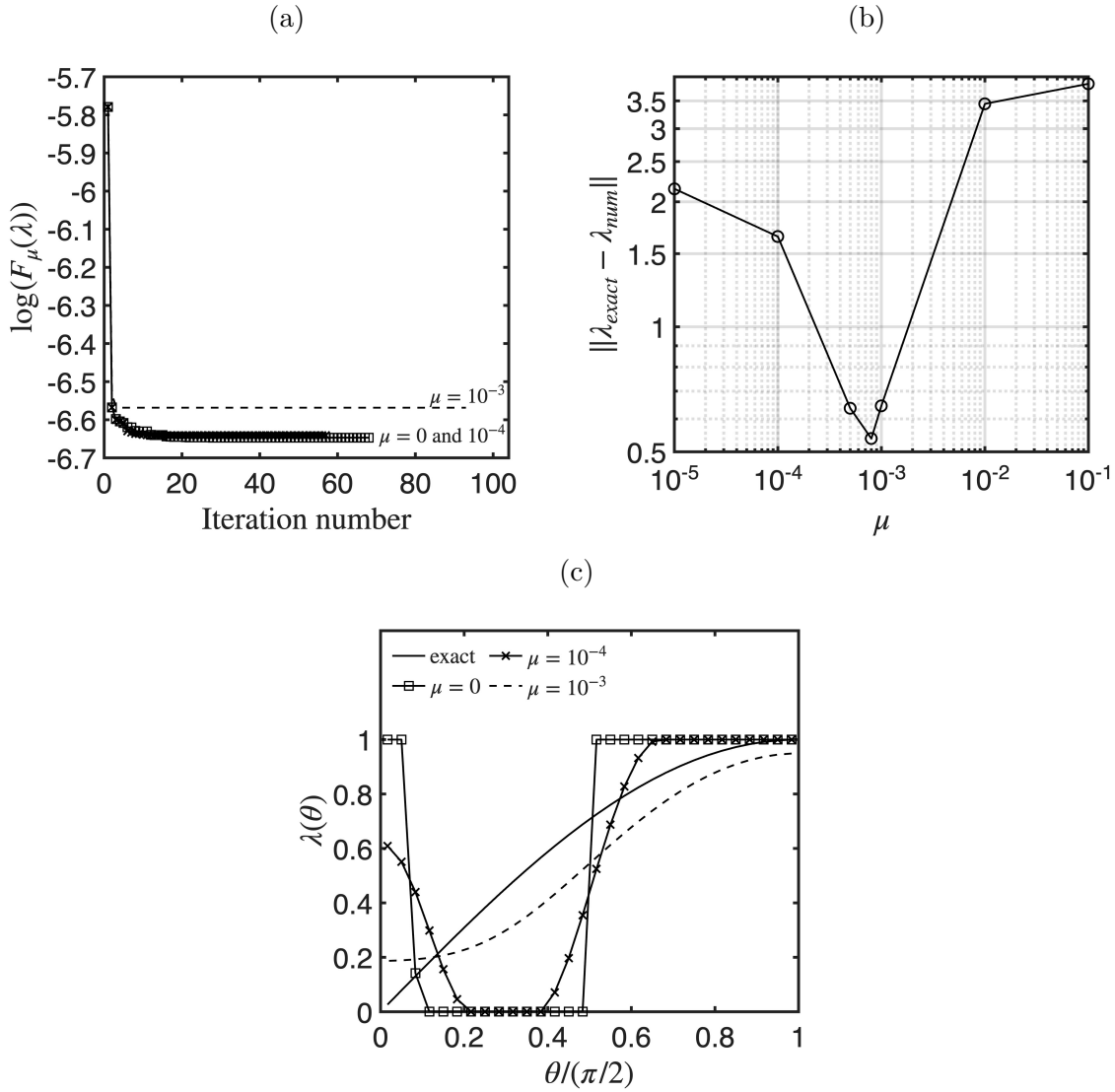

Figure 5.4: Determining the corrosion coefficient $\lambda(\theta)$ on the boundary Γ_0 using the LBC (5.7) for different values of μ with first-order regularization (5.18), in case of p = 1% noise. (a) The convergence of the objective function with respect to the number of iterations, (b) the norm of the difference between the exact and numerical solutions of λ for various values of μ, and (c) the comparison between the exact and numerical solutions for the coefficient λ on Γ_0.

5.5.2 Example 2: Nonlinear boundary condition (NLBC) on Γ_0

In case of the Butler-Volmer law of corrosion (5.6), the nonlinear boundary condition (NLBC) on Γ_0 recasts as

$$\frac{\partial u}{\partial n} + 2\lambda \sinh\left(\frac{u}{2}\right) = 0 \quad \text{on } \Gamma_0. \tag{5.21}$$

The numerical results presented in Figure 5.5 show similar features to those of the previous example; however, the less accurate results obtained in Figure 5.5(c) compared to those of Figure 5.4(c) show that the NLBC (5.21) introduces extra numerical challenges causing the functional (5.14) that is minimized becoming trapped perhaps in a local minimum. Improved results may be obtained by reducing the number of unknowns using a finite set of trigonometric polynomials for approximating the corrosion coefficient; however, this investigation is deferred to future work in Chapter 7.

5.6 Conclusions

This chapter focused on the inverse parameter identification problem, aiming to reconstruct the corrosion coefficient λ on the corroded boundary Γ_0, using data measured on the accessible boundary Γ_2. Unlike traditional studies that assume a constant coefficient, this chapter formulated λ as a spatially varying function $\lambda(\theta)$. The governing Laplace equation was discretized using the BEM. A nonlinear least-squares method was then employed to identify the unknown coefficient. To make the solution stable, a regularization term was included in the objective functional to ensure the stability of solution, especially in the presence of noise. The MATLAB computational program for the reconstruction of the space-dependent corrosion coefficient corresponding to Figure 5.5 is included in Section A.5 of Appendix A.

The originality of this chapter lies in the successful reconstruction of the corrosion coefficient as a space-dependent function using the first-order Tikhonov regularization method.

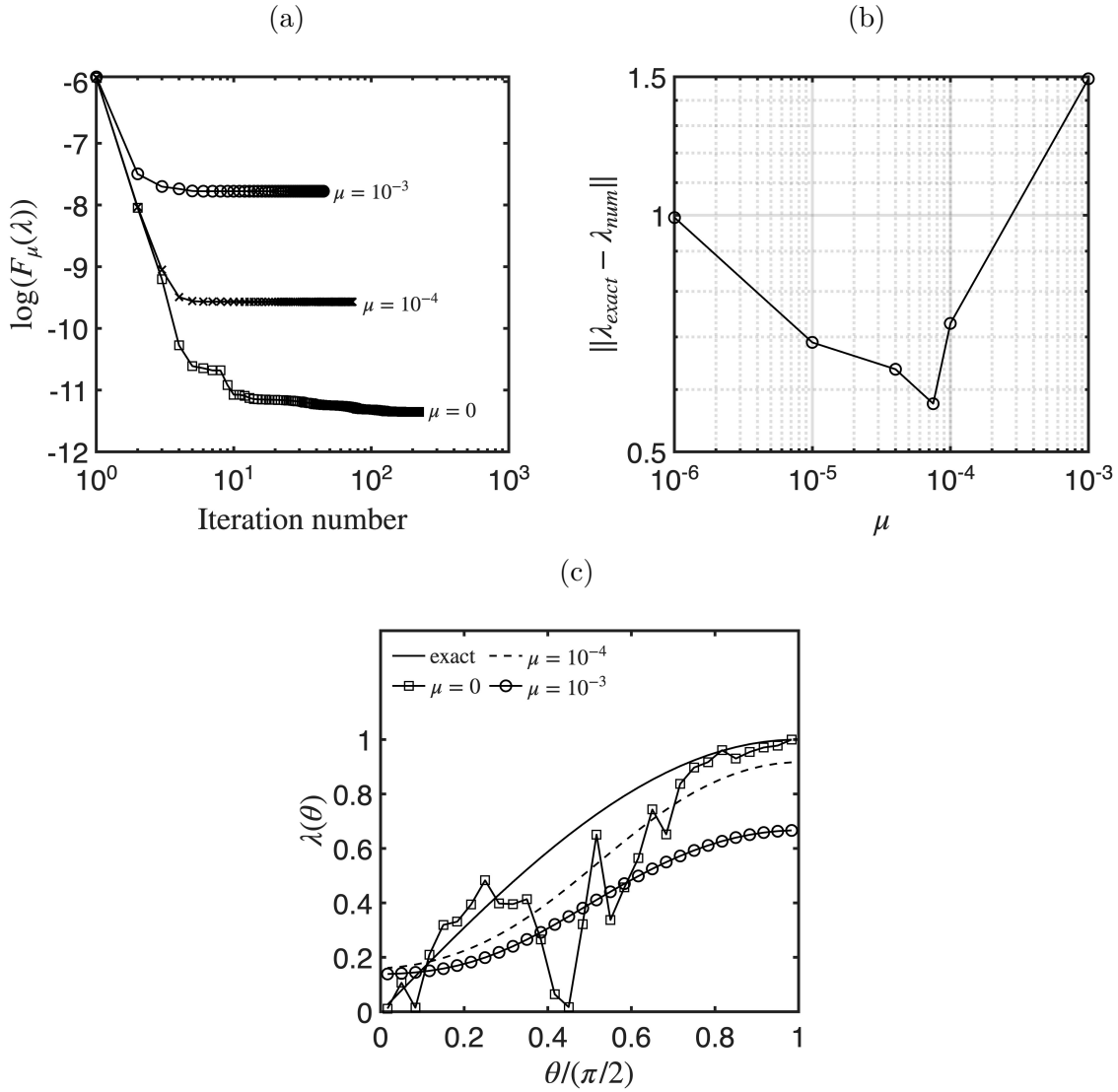

Figure 5.5: Determining the corrosion coefficient $\lambda(\theta)$ on the boundary Γ_0 using the NLBC (5.21) for different values of μ with first-order regularization (5.18), in case of no noise p = 0. (a) The convergence of the objective function with the number of iterations, (b) the norm of the difference between the exact and numerical solutions of λ for various values of μ, and (c) the comparison between the exact and numerical solutions for the coefficient λ on Γ_0.

The next Chapter 6 will investigate the inverse problem in which the corroded boundary Γ_0 is unknown, but the coefficient of corrosion λ is known, whilst in Chapter 7, we shall address the combined inverse problem of recovering both Γ_0 and λ.

Chapter 6

Determination of a corroded boundary

In this chapter, we consider the inverse problem depicted in Figure 6.1 that requires determining the corroded boundary Γ_0 from measured values of the potential on Γ_2. We use an iterative algorithm that adjusts the corrosion's geometry until the computed potential values on Γ_2 match the measured ones within a specified tolerance. This stopping criterion is chosen to account for the noise level in the measured data. This study has important implications for the modelling and prediction of boundary corrosion in industrial applications.

6.1 Introduction

In many engineering and scientific applications, boundary value problems arise in modeling physical phenomena such as heat conduction, fluid flow and electrostatics. A challenge in these problems is the presence of unknown boundaries, particularly in cases involving material degradation, corrosion or other environmental effects. Identifying these boundaries accurately is essential for predicting system behaviour and making informed decisions regarding maintenance and design.

This chapter focuses on an inverse nonlinear boundary value problem where part of the boundary, denoted as representing a corroded region, is unknown and must be determined.

The problem is formulated using Laplace's equation, which describes the steady-state potential or temperature distribution within the domain. The practical task is to determine the corroded boundary of a buried pipe from a set of measurements of electrical potentials or current densities at the accessible exterior boundary and to plan maintenance or repair work. Basically, the same inverse problem occurs in monitoring the corrosion of a blast furnace in the steel industry [91]. The electromagnetic field can be generated by passing an alternating current through the medium, and the resulting changes in the field caused by corrosion can be used to figure out the location and extent of corrosion within the pipe. Prior to this study, the determination of a perfectly conductive or insulated corroded part of boundary, formulated as a Cauchy problem in electrostatics governed by the Laplace's equation, was investigated using the BEM or the MFS combined with the level-set method in [82, 99, 100], or with the nonlinear Tikhonov's regularization method in [83]. For some special rectangular geometries, analytical treatment is also possible [81]. The novelty in our study is the linear or nonlinear Robin-type boundary condition that is prescribed on the corroded boundary. For example, in case of the linear Robin boundary condition (2.30) being prescribed over the corroded boundary Γ_0 it is known that two linearly independent pairs of Cauchy data (i.e., potential and flux) on Γ_2 are needed to ensure retrieving uniquely the unknown boundary Γ_0, see [24, 86]. This is different to the knowledge of only a single pair of Cauchy data on Γ_2 that is sufficient to identify uniquely a perfectly conductive, i.e., $u|_{\Gamma_0}= 0$, see [71], or insulated, i.e., $\partial_n u|_{\Gamma_0}= 0$, corroded boundary Γ_0, see [62]. For the fully nonlinear Robin boundary condition (2.33), uniqueness results in identifying Γ_0 are not available.

In the BEM application for solving inverse problems, the relationship between the boundary data and the unknown field variables is mathematically explained using integral equations, discretising only the boundary of the solution domain $\partial\Omega$ rather than the entire domain Ω, see [75]. The BEM divides the region's boundary into a number of boundary elements and calculates the boundary integrals analytically or numerically. The nonlinear system of equations that describes the relationship between the unknown field variables and the boundary data is frequently solved by iterative algorithms in nonlinear inverse

problems. The BEM is very adaptable and can be used to detect corrosion in a variety of structures, including metal plates and beams, as well as pipes of various shapes and sizes [48].

The estimation of corrosion distribution on a pipe surface based on observed boundary data is referred to as the inverse problem in pipe corrosion. Due to the ill-posedness of the problem, small errors in the measured data can result in significant errors in the estimated corrosion distribution, making this a challenging task. To overcome this difficulty, we suggest a thorough strategy that combines the BEM with the Tikhonov regularisation method, as in Chapter 5. The inverse problem is mathematically expressed by minimizing an objective function that includes a data misfit term and a regularisation term, that is included in order to promote solution's smoothness and stability. It aids in the prevention of over-fitting and it reduces the solution's sensitivity to measurement noise.

In this chapter, we present a concise approach to solving an inverse shape corrosion problem using boundary elements. Section 6.2 provides the mathematical formulation, followed by the BEM description in Section 6.3. The iterative method is outlined in Section 6.4, while Section 6.5 discusses the numerical results. Conclusions are given in Section 6.6.

6.2 Mathematical formulation of the inverse problem

The inverse corrosion problem considered in this chapter involves determining the unknown corroded boundary Γ_0, as illustrated in Figure 6.1, based on potential measurements $h = u|_{\Gamma_2}$ taken at the accessible boundary Γ_2, as given by the following set of equations:

$$\Delta u = 0 \qquad \text{in} \quad \Omega, \tag{6.1}$$

subject to the boundary conditions

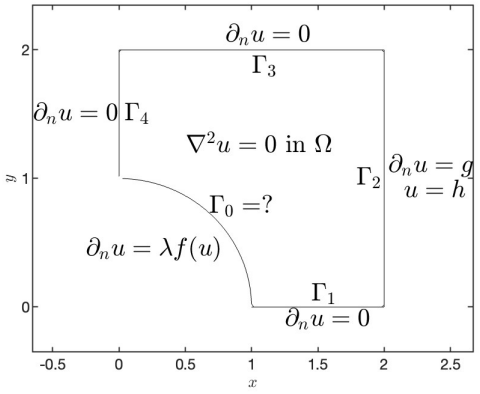

Figure 6.1: The formulation of the identification problem for the corroded boundary Γ_0.

$$\frac{\partial u}{\partial n} = 0 \quad \text{on } \Gamma_1 \cup \Gamma_3 \cup \Gamma_4, \tag{6.2a}$$

$$\frac{\partial u}{\partial n} = \lambda f(u) \quad \text{on } \Gamma_0, \tag{6.2b}$$

$$\frac{\partial u}{\partial n} = g \quad \text{on } \Gamma_2, \tag{6.2c}$$

$$u = h \quad \text{on } \Gamma_2. \tag{6.2d}$$

In (6.2b), λ denotes the coefficient of corrosion and the corrosion law in is given by

$$f(u) = -u \qquad \text{linear Newton's law,} \tag{6.3a}$$

$$f(u) = -2\sinh\left(\frac{u}{2}\right) \qquad \text{non-linear Butler-Volmer law.} \tag{6.3b}$$

This inverse nonlinear problem can be attempted to be solved using optimization techniques such as the nonlinear least-squares method, which minimizes the sum of squared errors between the observed and computed data on Γ_2. However, caution is required, as its non-convex nature means that the approximate solution can be trapped in a local minimum during the iterative solution, depending upon the initial guess [44].

6.3 The boundary element method (BEM)

We employ the BEM to discretise the inverse problem of reconstructing the shape of the corroded boundary Γ_0. Upon inducing a current flux (6.2c) and utilising the resulting boundary potential (6.2d), the objective is to infer the geometry of Γ_0, which

represents the region impacted by corrosion. By leveraging the known boundary data, we can effectively reconstruct the characteristics of the unknown corroded surface.

Once the boundary conditions outlined in (6.2a)–(6.2d) are applied, we obtain a system of equations that can be solved to determine the unknown boundary parameters and reconstruct the shape of Γ_0. To proceed with the reconstruction, we employ the BEM system of equations (4.11) presented in Chapter 4, reformulated as follows:

$$\sum_{j=1}^{2N} B_{ij}u_j + \sum_{j=4N+1}^{M} B_{ij}u_j + \sum_{j=1}^{N} \lambda_j A_{ij} f(u_j) = -\sum_{j=2N+1}^{4N} A_{ij}g_j - \sum_{j=2N+1}^{4N} B_{ij}h_j, \quad i = \overline{1, M}, \quad (6.4)$$

where h_j represents the value of u_j for $j = \overline{2N+1, 4N}$. In simulations, the Dirichlet data (6.2d) on Γ_2 is obtained numerically by solving first the direct problem (6.1), (6.2a)–(6.2c) with known Γ_0. In real applications, this data is raw, as it comes from measurements, and it is contaminated with noise.

6.4 Iterative minimization

This section explores the numerical reconstruction of the corroded boundary Γ_0 when using one or two pairs of Cauchy data (6.2c) and (6.2d). These involve applying functionally independent fluxes g_1 and g_2, defined in expressions (2.27) and (2.28), respectively, in (6.2c) and measuring the corresponding Dirichlet data $u^{(i)}|_{\Gamma_2} = h^{(i)}$ for $i = 1, 2$. The corresponding Dirichlet data (6.2d) resulted from solving the direct problem is denoted by $u^{(i)}|_{\Gamma_2} =: h^{(i)}$ for $i = 1, 2$. These fabricated data have already been presented in Figure 2.2 of Chapter 2. Furthermore, as the data is supposed to come from practical measurement it is therefore contaminated with noise, which we numerically simulate by replacing the so-called exact data $h^{(i)}$ by its noisy counterpart

$$h^{(i)}(\mathrm{p}) = (1 + \rho \mathrm{p})h^{(i)}, \quad i = 1, 2, \quad (6.5)$$

where p represents the percentage of noise and ρ are random variables drawn from a uniform distribution in $[-1, 1]$.

We model Γ_0 as part of a star-shaped domain relative to the origin, parameterized as:

$$\Gamma_0 = \{r(\theta)(\cos(\theta), \sin(\theta)) \mid \theta \in [0, \pi/2]\}. \tag{6.6}$$

A truncated Fourier series could further simplify $r(\theta)$ for enhanced stability and uniqueness [31], and this will be further adopted in Chapter 7. Alternatively, Γ_0 could be sought as the graph of unknown function defined by $\Gamma_0 = \{(x, y(x)) \mid x \in (0, 1)\}$, as in [76].

Define the points $\underline{p}_j = (r(\theta_j)\cos(\theta_j), r(\theta_j)\sin(\theta_j))$ for $j = \overline{0, N}$, with $\theta_j = \frac{\pi(N-j)}{2N}$, where $r(\theta_0) = r(\frac{\pi}{2}) = 1$ and $r(\theta_N) = r(0) = 1$. Let $r_j := r(\theta_j)$ for $j = \overline{1, N-1}$. The inverse problem then seeks the vector $\underline{r} = (r_j)_{j=\overline{1,N-1}}$ that minimizes:

$$C(\underline{r}) = \alpha\|u^{(1)}(\underline{r}; \cdot) - h^{(1)}\|^2_{L^2(\Gamma_2)} + \beta\|u^{(2)}(\underline{r}; \cdot) - h^{(2)}\|^2_{L^2(\Gamma_2)}, \tag{6.7}$$

where $u^{(i)}(\underline{r}; \cdot)|_{\Gamma_2}$ is the computed solution for a given $\underline{r}$, and (α, β) are set to $(1, 0), (0, 1)$ or $(1, 1)$, depending on the considered measurements. In discrete form:

$$C(\underline{r}) \approx \frac{2}{2N}\left\{\alpha \sum_{j=2N+1}^{4N}\left(u_j^{(1)}(\underline{r}) - h_j^{(1)}\right)^2 + \beta \sum_{j=2N+1}^{4N}\left(u_j^{(2)}(\underline{r}) - h_j^{(2)}\right)^2\right\}. \tag{6.8}$$

Further, since the inverse problem is, in general, unstable with respect to noise in the measured data (6.2d), the functional (6.7) has to be penalised by adding a regularization term to it resulting in the regularized functional

$$C_\mu(\underline{r}) = C(\underline{r}) + \mu\|\underline{r} - \underline{r}^0\|^2, \tag{6.9}$$

where $\underline{r}^0$ is a chosen initial guess and $\mu \geq 0$ is a regularization parameter to be prescribed. In discretised form, equation (6.9) becomes

$$C_\mu(\underline{r}) = C(\underline{r}) + \mu \sum_{j=1}^{N-1}\left(r_j - r_j^0\right)^2. \tag{6.10}$$

Bounds such as $r_{min} \leq r_j \leq r_{max}$ for $j = \overline{1, N-1}$ are imposed, and the constrained minimization is performed iteratively in MATLAB using the `lsqnonlin` routine, starting from an initial guess $\underline{r}^0$, with gradients computed via finite differences [59]. The `lsqnonlin` routine excels in nonlinear, potentially ill-posed parameter estimation tasks. It iteratively refines the parameter vector $\underline{r}$ by approximating the objective function quadratically and solving for steps within specified bounds. For non-quadratic cases, `fmincon` is an alternative [38]. Integrated within the MATLAB optimization toolbox, `lsqnonlin` supports various algorithms such as trust-region-reflective (default), Levenberg-Marquardt and interior-point [59, 39]. Here, the trust-region-reflective method is used to ensure the quadratic model's accuracy. The trust region adjusts based on residual reduction, balancing stability and convergence [39]. Termination criteria, like Function Tolerance (gradient magnitude threshold), can be tuned to halt the iterations when changes become negligible [59].

6.5 Numerical results and discussion

We consider two examples of corroded boundary Γ_0 given by

$$r(\theta) = 1, \qquad \theta \in [0, \pi/2] \qquad \text{(Example 1)}, \tag{6.11}$$

and

$$r(\theta) = \frac{1}{\sqrt{2}\ \sin(\theta + \pi/4)}, \qquad \theta \in [0, \pi/2] \qquad \text{(Example 2)}, \tag{6.12}$$

see Figure 6.2.

The data $h = u|_{\Gamma_2}$ is obtained by solving the direct problem (2.1)–(2.3) with $f(u)$ given by (6.3a) or (6.3b) using the BEM with $N = 90$, i.e. $M = 7N = 630$. For the inverse problem, to circumvent an inverse crime [65], the BEM direct solver employs $N = 30$ (i.e., $M = 7N = 210$) during the iterative solution process using the `lsqnonlin` algorithm (with Tolerance $= 10^{-20}$ and Function Tolerance $= 10^{-20}$).

(a)Example 1 (b)Example 2

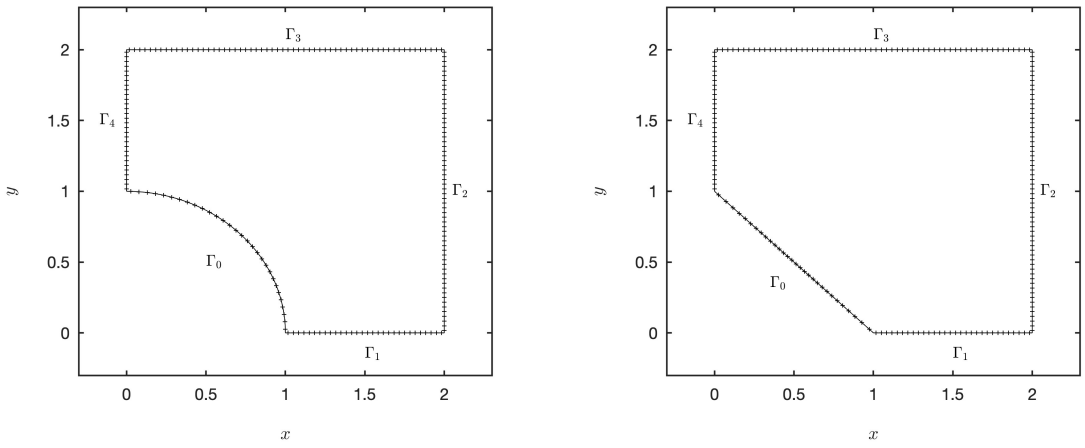

Figure 6.2: The geometry of Examples 1 and 2.

The lower and upper bounds on the variables $(r_i)_{i=\overline{1,N-1}}$ are taken as:

$$\underline{r}_{min} = \begin{cases} 0.5 & \text{for Example 1,} \\ 0.7 & \text{for Example 2,} \end{cases} \quad \text{and} \quad \underline{r}_{max} = \begin{cases} 1.5 & \text{for Example 1,} \\ 1.1 & \text{for Example 2.} \end{cases}$$

Several initial guesses for $(\underline{r}^0)_{i=\overline{1,N-1}}$ are investigated.

6.5.1 Example 1: Linear boundary condition on Γ_0

In the case of the law $f(u) = -u$ given by (6.3a), the Robin boundary condition (6.2b) is linear and given by (taking $\lambda = 1$)

$$\frac{\partial u}{\partial n} + u = 0 \quad \text{on } \Gamma_0. \tag{6.13}$$

6.5.1.1 Noiseless data (convergence)

We analyse the reconstruction of the polar radius $r(\theta)$, which yields Γ_0 through the star-shape representation (6.6), from one or two pairs of data (6.2d) (illustrated in Figure 2.2, Chapter 2) induced by the flux (2.27) and/or (2.28). Inverting individually each of these boundary data leads to different results, as depicted in Figure 6.3, for two different initial

guesses

$$r^0(\theta) = 1.1, \quad \theta \in \left[0, \frac{\pi}{2}\right], \tag{6.14}$$

and

$$r^0(\theta) = \frac{1}{\sqrt{2}\sin(\theta + \frac{\pi}{4})}, \quad \theta \in \left[0, \frac{\pi}{2}\right], \tag{6.15}$$

which is a straight line between the points of coordinates (0,1) and (1,0). Figure 6.3(a) shows the logarithm of the objective function (6.8), plotted against the iteration number. It can be seen that the objective function (6.8) decreases significantly in the first 500 iterations and stabilises around 10^{-12} for the discontinuous flux (2.27), for both initial guesses (6.14) and (6.15). This behaviour is also seen when inverting the single data $h^{(2)}$ induced by the smooth flux (2.28) with the initial guess (6.14); however, with the initial guess (6.15), which is further away from the exact solution (6.11) than the initial guess (6.14), the objective function (6.8) converges much slower taking about 3000 iterations to stabilise around 10^{-12}. Also, overall the convergence of (6.8) with the number of iterations is faster when inverting the data $h^{(1)}$ than $h^{(2)}$. The numerical recoveries of Γ_0 are illustrated in Figure 6.3(b). From this figure it can be seen that the reconstructions deviate from the exact solution, particularly when inverting the discontinuous flux g_1, showing noticeable errors, especially for the initial guess (6.15). On the other hand, inverting the smooth flux g_2 performs better for the initial guess (6.14), but still exhibits inaccuracies for the second initial guess (6.15).

The inaccuracies in Figure 6.3 highlight the limitations of using a single flux for reconstructing Γ_0, impacting the reliability of corrosion detection in buried pipe. The discontinuous flux g_1 introduces significant errors, while the smooth flux g_2, though better, is still not sufficiently accurate. The reconstruction errors with the individual fluxes highlight the idea of using data fusion to combine the fluxes g_1 and g_2, aiming to improve accuracy and reliability far better pipeline corrosion detection and maintenance.

Figure 6.4 illustrates the results of the combined data approach based on minimizing the functional (6.8) with $\alpha = \beta = 1$, i.e. based on inverting both the Dirichlet data $h^{(1)}$ and $h^{(2)}$ produced by the fluxes (2.27) and (2.28), respectively, for each of the two

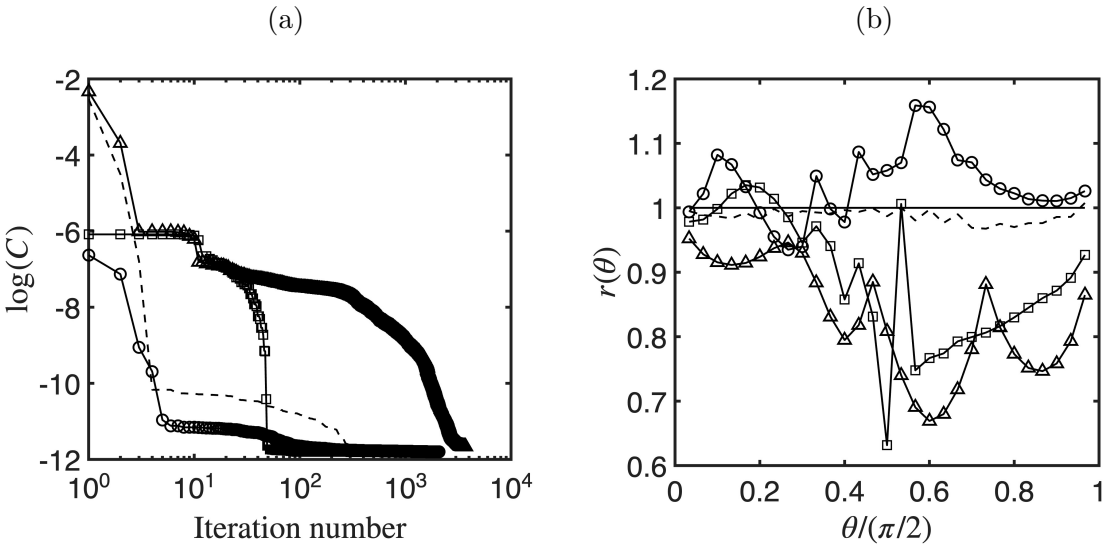

Figure 6.3: No noise, no regularization (Example 1 with linear Robin boundary condition (6.13)). (a) The convergence of the objective function (6.8) for $(\alpha, \beta) = (1, 0)$ corresponding to inverting the single data $h^{(1)}$ with the initial guess (6.14) shown with $(-\circ-)$ and with the initial guess (6.15) shown with $(-\square-)$, and for $(\alpha, \beta) = (0, 1)$ corresponding to inverting the single data $h^{(2)}$ with the initial guess (6.14) shown with $(---)$ and with the initial guess (6.15) shown with $(-\Delta-)$, along with the corresponding (b) numerical reconstruction of $r(\theta)$ in comparison with the exact solution (6.11) shown with (——).

initial guesses (6.14) and (6.15). As previously obtained, the initial guess (6.15) which is farther way from the exact solution (6.11) than the initial guess (6.14), requires a larger number of iterations to achieve a small convergence threshold of 10^{-10}, as illustrated in Figure 6.4(a). The numerical results for $r(\theta)$ illustrated in Figure 6.4(b) still shows some level of dependency on the initial guess; however, the reconstructions are for better and more robust than those obtained in Figure 6.3(b). This is to be expected because for Figure 6.4 we invert double the amount of data than for Figure 6.3, ensuring the uniqueness of solution [10]. These improvements confirm that data fusion capitalizes on the complementary information encapsulated in the two fluxes g_1 and g_2, yielding more stable and accurate results for practical corrosion detection tasks.

6.5.1.2 Noisy data with regularisation

To model real-world conditions, noise is introduced into the data $h^{(i)}$ by equation (6.5). Figure 6.5 shows the results obtained from inverting the combined data contaminated with p = 1% noise, only for the initial guess (6.14). The Tikhonov regularization (6.10)

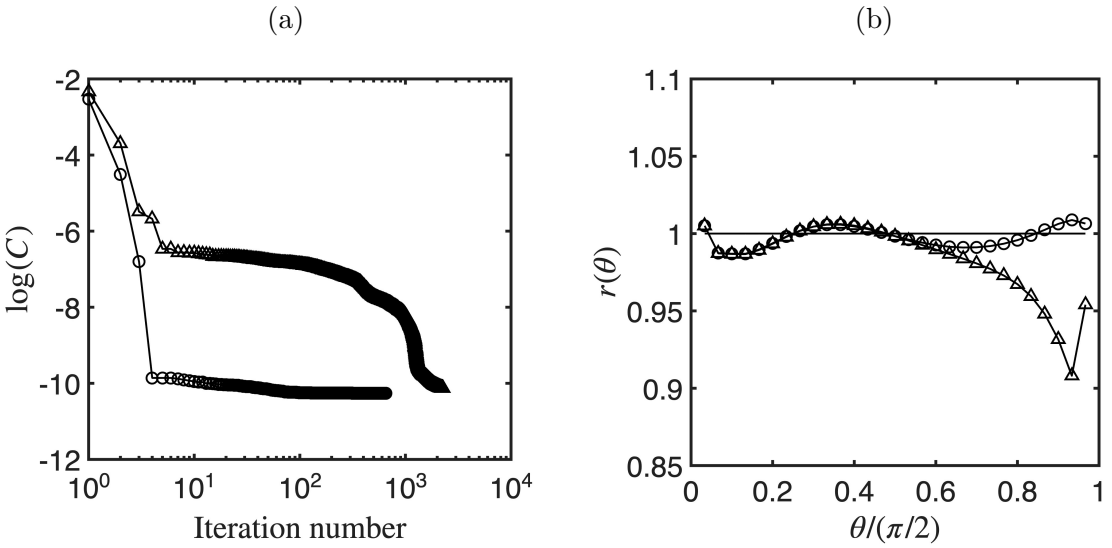

Figure 6.4: No noise, no regularization (Example 1). (a) The convergence of the objective function (6.8) for $(\alpha, \beta) = (1, 1)$ corresponding to inverting both the data $h^{(1)}$ and $h^{(2)}$ with the initial guess (6.14) shown with $(- \circ -)$ and with the initial guess (6.15) shown with $(-\Delta-)$, along with the corresponding (b) numerical reconstruction of $r(\theta)$ in comparison with the exact solution (6.11) shown with (——).

is applied for various regularization parameters μ to assess its impact on reconstruction stability and accuracy. Figure 6.5(a) shows the logarithm of the objective function plotted against the iteration number. From this figure it can be seen that in the absence of regularization $\mu = 0$, the algorithm converges slowly, requiring approximately 1200 iterations to stabilise around 10^{-5}. In contrast, for the regularized case with $\mu = 10^{-2}$, the algorithm converges more rapidly, stabilizing around 10^{-4} in approximately 100 iterations.

The numerical results for the reconstructed polar radius are illustrated in Figure 6.5(b) for different parameters μ. For $\mu = 0$, the solution exhibits noticeable oscillations and deviates significantly from the true shape, as expected since the inverse problem under investigation is ill-posed. On contrary, the reconstruction with $\mu = 10^{-2}$ shows improved agreement with the exact solution and reduced oscillatory distortions. This demonstrates the role of regularization in mitigating instabilities caused by the noisy input. For the optimal choice of the regularization parameter, the obtained L-curve in Figure 6.5(c) shows the residual (6.7) versus the norm of $(r - r^0)$. The resulting L-shaped curve reveals that $\mu = 10^{-2}$ offers the best compromise between data fidelity and regularization, suggesting

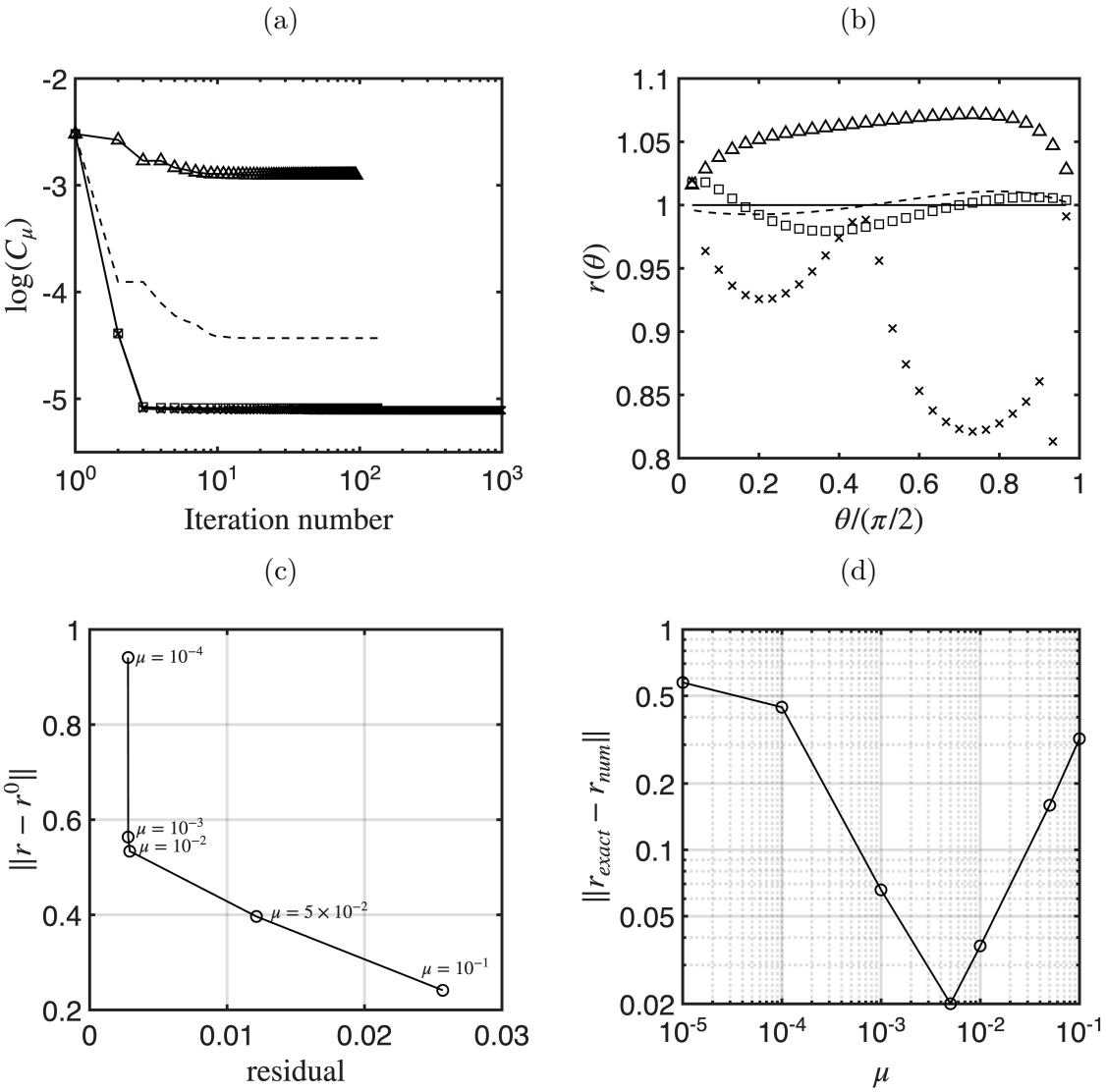

Figure 6.5: Noise p = 1%, with regularization (Example 1). (a) The convergence of the objective function (6.10) for $(\alpha, \beta) = (1, 1)$ corresponding to inverting both the data $h^{(1)}$ and $h^{(2)}$ with the initial guess (6.14), shown for different values of the regularization parameter: $(- \times -)$ for $\mu = 0$, $(-\square-)$ for $\mu = 10^{-3}$, $(- - -)$ for $\mu = 10^{-2}$ and $(-\Delta-)$ for $\mu = 10^{-1}$, along with the corresponding (b) numerical reconstruction of $r(\theta)$ in comparison with the exact solution (6.11) shown with (——), (c) the L-curve, plotting the residual (6.8) versus the norm of $(r - r^0)$, and (d) the norm of the error, as a function of the regularization parameter.

it as the most suitable parameter under the given noise conditions. Figure 6.5(d) illustrates the error norm for different values of μ, corroborating earlier observations. The highest error occurs at $\mu = 0$, whereas the lowest is attained at $\mu = 10^{-2}$. This further validates the effectiveness of the Tikhonov regularization in improving the stability and accuracy of the corrosion shape reconstruction from noisy data.

However, Figure 6.6 exposes a fundamental limitation in the method's robustness, when initialized with (6.15), the reconstruction quality deteriorates dramatically despite employing regularization. The reconstructed boundary in Figure 6.6(b) exhibits substantial deviations from the exact solution with persistent oscillatory behaviour, even when using the regularization parameters. This contrasts sharply with the accurate reconstruction achieved in Figure 6.5 using the initial guess (6.14), demonstrating that the success of the method depends critically on the choice of initial approximation. Moreover, the poor performance persists across different regularization parameter values, indicating that regularization cannot remedy the fundamental issue of inappropriate initialization. This reveals a significant practical limitation: the optimization algorithm fails to recover the true solution when started from geometrically dissimilar initial conditions, suggesting that the method lacks the robustness required for real-world applications where good initial estimates may not be available.

6.5.2 Example 1: Nonlinear boundary condition on Γ_0

In case of the law $f(u) = -2\sinh\left(\frac{u}{2}\right)$ given by 6.3(b), the Robin boundary condition 6.2(b) is nonlinear and given by (taking $\lambda = 1$)

$$\frac{\partial u}{\partial n} + 2\sinh\left(\frac{u}{2}\right) = 0 \quad \text{on } \Gamma_0. \tag{6.16}$$

6.5.2.1 Noiseless data (convergence)

We also investigate the minimization of $r(\theta)$ over Γ_0 under the nonlinear boundary condition (6.16), applying a function tolerance option of 10^{-25}. Figure 6.7 illustrates the reconstruction of the corroded boundary Γ_0 using two distinct flux datasets on Γ_2, where

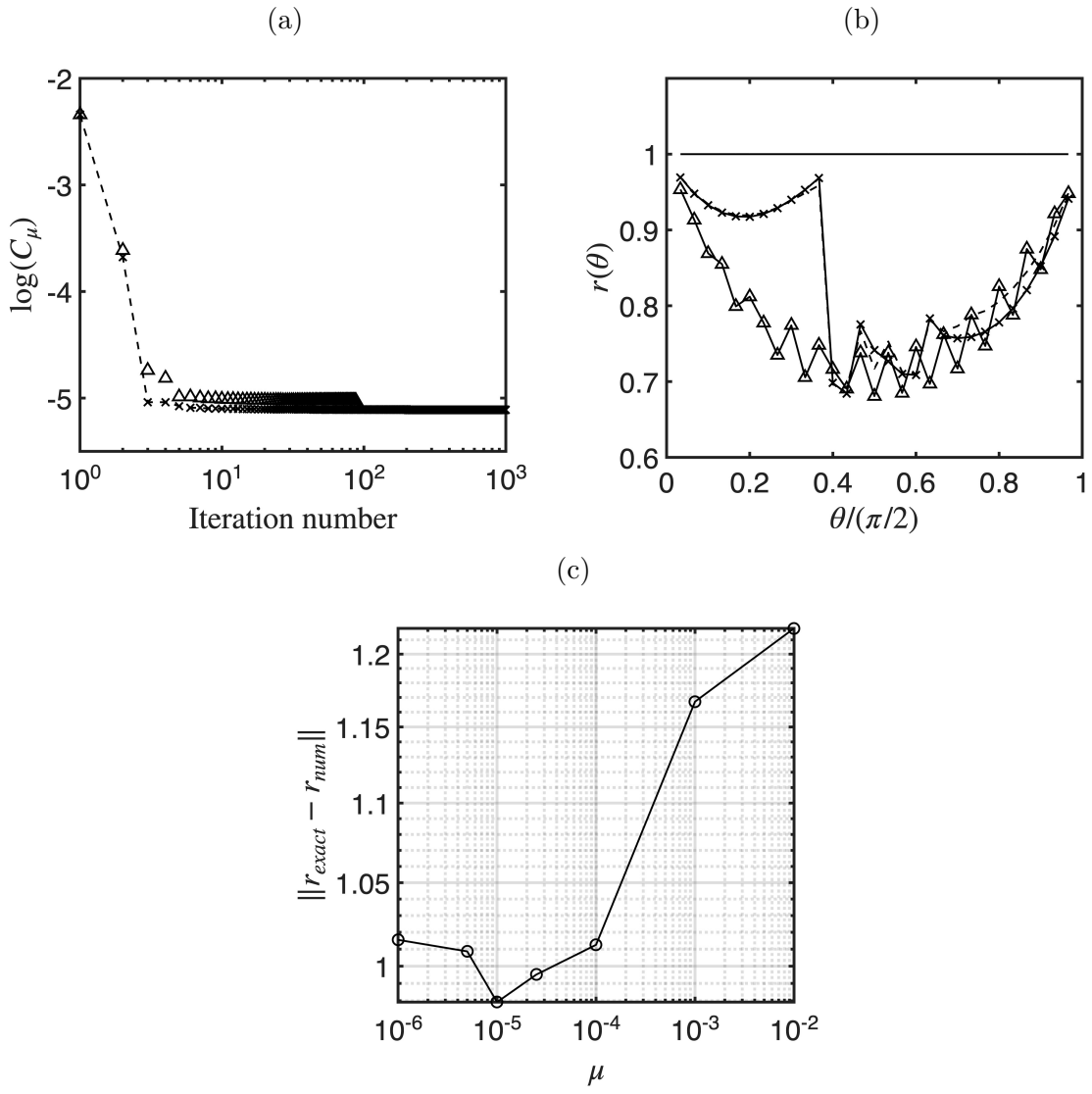

Figure 6.6: Noise p = 1%, with regularization (Example 1). (a) The convergence of the objective function (6.10) for $(\alpha, \beta) = (1, 1)$ corresponding to inverting both the data $h^{(1)}$ and $h^{(2)}$ with the initial guess (6.15), shown for different values of the regularization parameter: $(- \times -)$ for $\mu = 0$, $(- - -)$ for $\mu = 10^{-5}$ and $(-\Delta-)$ for $\mu = 10^{-2}$, along with the corresponding (b) numerical reconstruction of $r(\theta)$ in comparison with the exact solution (6.11) shown with (——), (c) the L-curve, plotting the residual (6.8) versus the norm of $(r - r^0)$, and (d) the norm of the error, as a function of the regularization parameter.

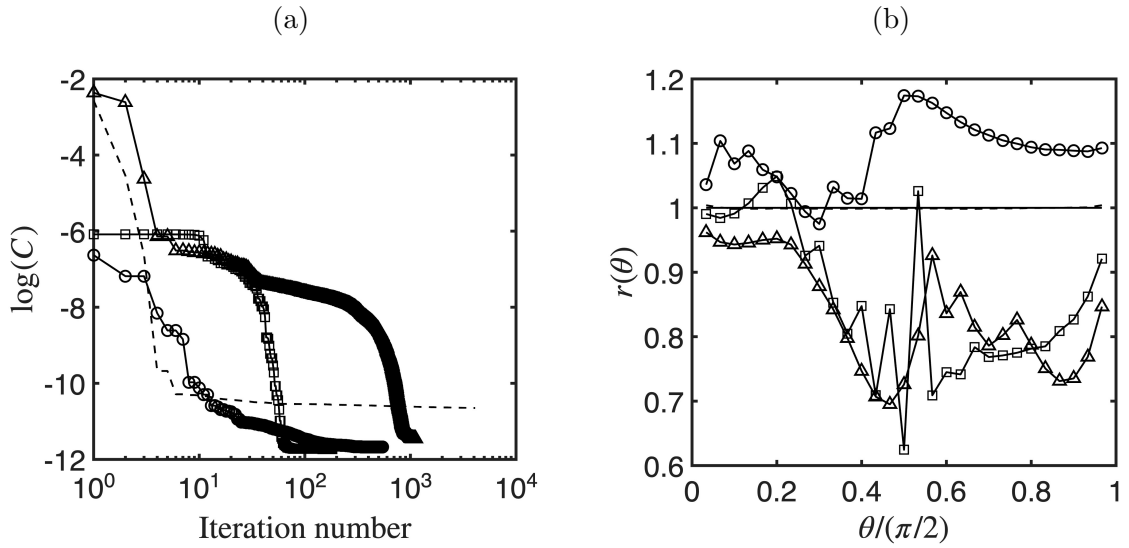

Figure 6.7: No noise, no regularization (Example 1) with NLBC. (a) The convergence of the objective function (6.8) for $(\alpha, \beta) = (1, 0)$ corresponding to inverting the single data $h^{(1)}$ with the initial guess (6.14) shown with $(- \circ -)$ and with the initial guess (6.15) shown with $(-\square-)$, and for $(\alpha, \beta) = (0, 1)$ corresponding to inverting the single data $h^{(2)}$ with the initial guess (6.14) shown with $(- - -)$ and with the initial guess (6.15) shown with $(-\Delta-)$, along with the corresponding (b) numerical reconstruction of $r(\theta)$ in comparison with the exact solution (6.11) shown with (——).

the solution of $r(\theta)$ is computed with the two initial guesses (6.14) and (6.15). In Figure 6.7(a) the solution obtained with the discontinuous flux introduces noticeable fluctuations near regions with sharp transitions. In contrast, the solution obtained with the smooth flux exhibits a more stable convergence pattern without large oscillations.

A similar process can be performed for the reconstructed corroded boundary Γ_0 with nonlinear boundary conditions as shown earlier with the linear one, see Figure 6.5, where the benefit of using two measurements to reconstruct the Robin boundary condition is highlighted in Figure 6.8.

6.5.2.2 Noisy data with regularisation

When 1% noise is introduced, the reconstruction quality remains heavily dependent on the initial guess. Figure 6.9 shows that starting with the initial guess (6.14) produces accurate results with the regularization parameter $\mu = 10^{-3}$. In contrast, Figure 6.10 demonstrates that using the initial guess (6.15) leads to poor reconstruction quality.

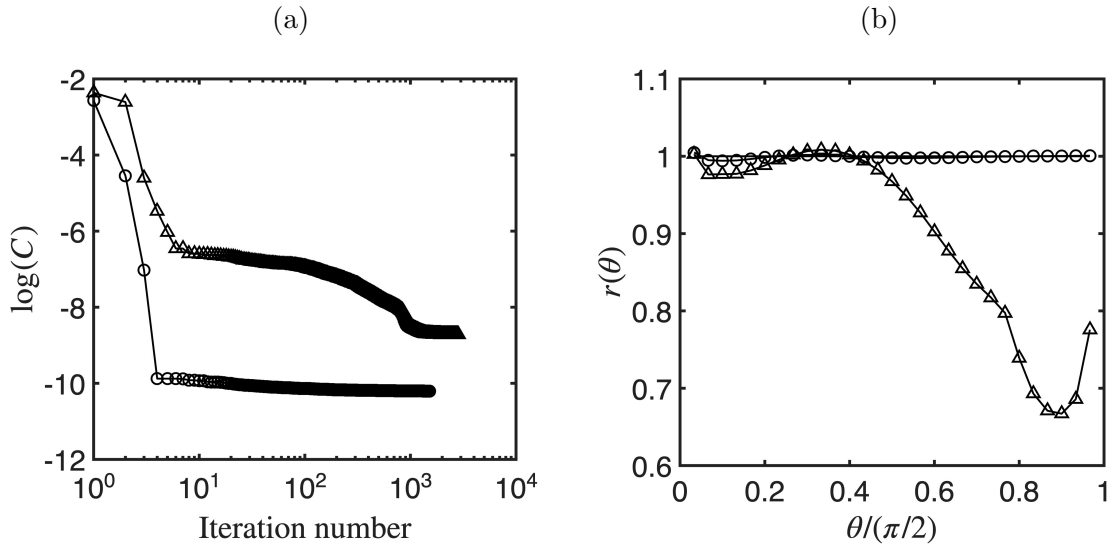

Figure 6.8: No noise, no regularization (Example 1) with NLBC. (a) The convergence of the objective function (6.8) for $(\alpha, \beta) = (1, 1)$ corresponding to inverting both the data $h^{(1)}$ and $h^{(2)}$ with the initial guess (6.14) shown with $(- \circ -)$ and with the initial guess (6.15) shown with $(-\Delta-)$, along with the corresponding (b) numerical reconstruction of $r(\theta)$ in comparison with the exact solution (6.11) shown with (——).

6.5.3 Example 2: Linear boundary condition on Γ_0

6.5.3.1 Noiseless data with regularisation

Figure 6.11 illustrates the reconstruction of the corroded boundary Γ_0, obtained by inverting the Dirichlet data $h^{(1)}$ and $h^{(2)}$ in the absence of noise, generated respectively by the fluxes (2.27) and (2.28), with the initial guess $r^0(\theta) = 1$. Figure 6.11(a) demonstrates the convergence behaviour for different regularization parameters. Notably, the unregularized case ($\mu = 0$) shows the slowest convergence, requiring approximately 1000 iterations to reach 10^{-9}. On the other hand, for $\mu = 10^{-4}$, the algorithm converges more quickly, stabilizing around 10^{-8} in approximately 500 iterations. The numerical results in Figure 6.11(b) reveal that regularization improves the solution quality. The unregularized solution ($\mu = 0$) exhibits some inaccuracies, but the regularized solution with $\mu = 10^{-4}$ or $\mu = 10^{-3}$ provide smoother and more accurate approximations to the exact solution (6.12).

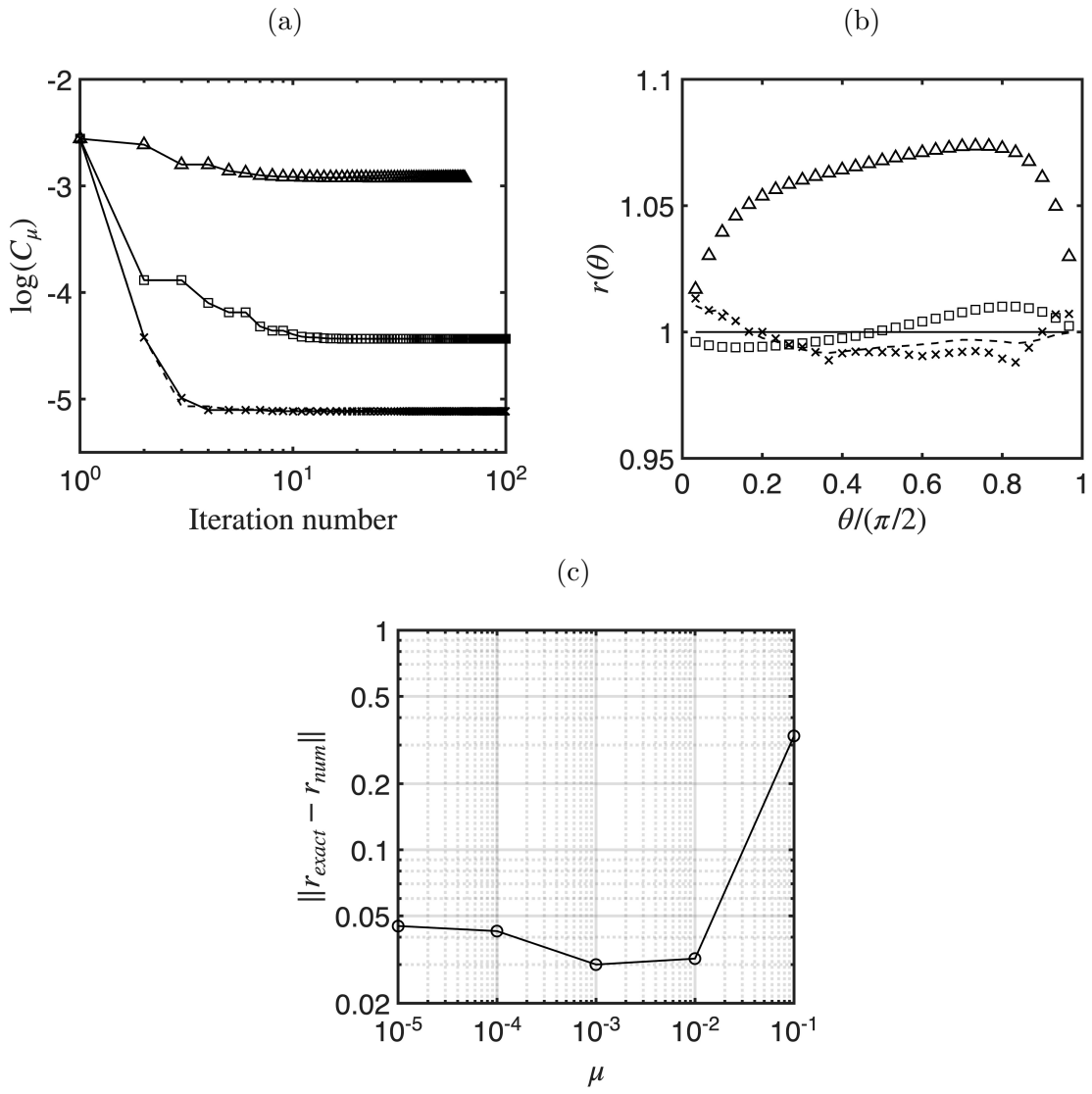

Figure 6.9: Noise p = 1%, with regularization (Example 1) with NLBC. (a) The convergence of the objective function (6.10) for $(\alpha, \beta) = (1, 1)$, corresponding to inverting both the data $h^{(1)}$ and $h^{(2)}$ with the initial guess (6.14), shown for different values of the regularization parameter: $(- \times -)$ for $\mu = 0$, $(- - -)$ for $\mu = 10^{-3}$, $(-\square-)$ for $\mu = 10^{-2}$ and $(-\Delta-)$ for $\mu = 10^{-1}$, along with the corresponding (b) numerical reconstruction of $r(\theta)$ in comparison with the exact solution (6.11) shown with (——), and (c) the norm of the error, as a function of the regularization parameter.

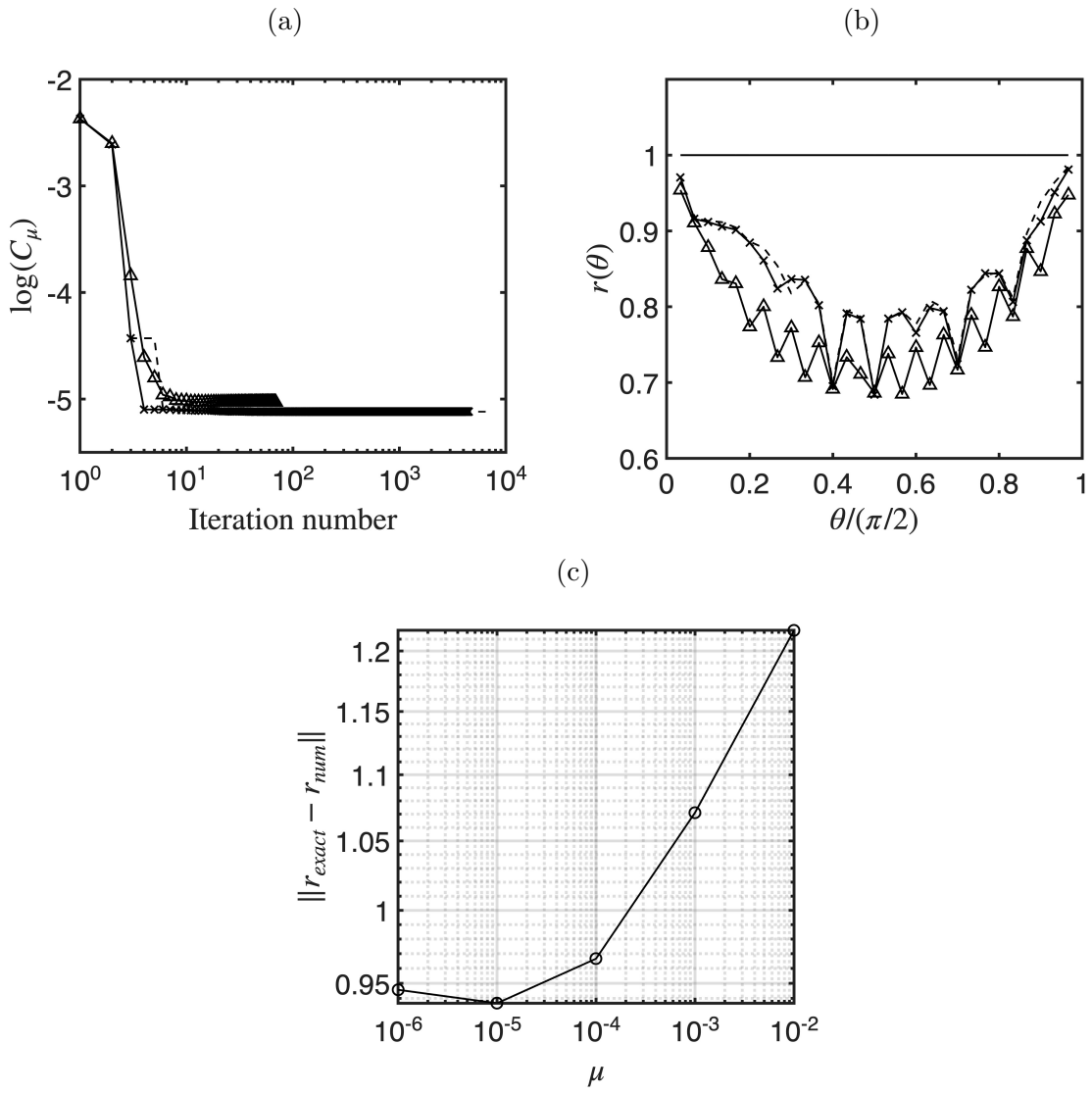

Figure 6.10: Noise p = 1%, with regularization (Example 1) with NLBC. (a) The convergence of the objective function (6.10) for $(\alpha, \beta) = (1, 1)$, corresponding to inverting both the data $h^{(1)}$ and $h^{(2)}$ with the initial guess (6.15), shown for different values of the regularization parameter: $(- \times -)$ for $\mu = 0$, $(- - -)$ for $\mu = 10^{-2}$ and $(-\Delta-)$ for $\mu = 10^{-1}$, along with the corresponding (b) numerical reconstruction of $r(\theta)$ in comparison with the exact solution (6.11) shown with (——), and (c) the norm of the error, as a function of the regularization parameter.

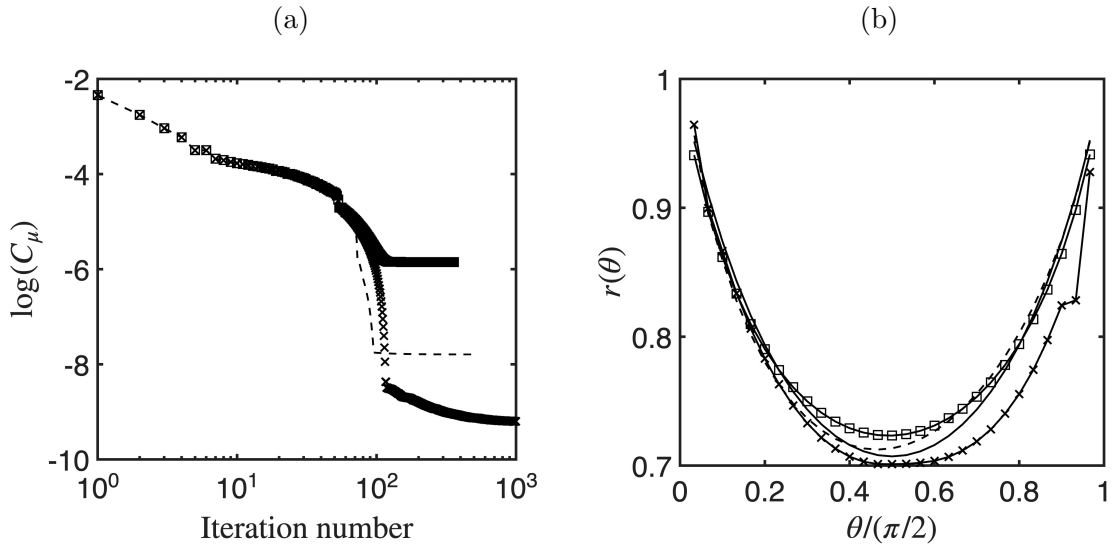

Figure 6.11: No noise, with regularization (Example 2) with LBC. (a) The convergence of the objective function (6.8) for $(\alpha, \beta) = (1, 1)$, corresponding to inverting both the data $h^{(1)}$ and $h^{(2)}$ with the initial guess $r^0(\theta) = 1$, shown for different values of the regularization parameter: $(- \times -)$ for $\mu = 0$, $(- - -)$ for $\mu = 10^{-4}$ and $(-\square-)$ for $\mu = 10^{-3}$, along with the corresponding (b) numerical reconstruction of $r(\theta)$ in comparison with the exact solution (6.12) shown with (——).

6.5.3.2 Noisy data with regularisation

To assess the stability of the reconstruction method under realistic conditions, noise is introduced into the data $h^{(i)}$ for $i = 1, 2$, by equation (6.5) with noise level p = 1%.

Figure 6.12 presents the reconstruction results for the corroded boundary Γ_0 from the combined noisy data with the initial guess $r^0(\theta) = 1$. Figure 6.12(a) shows the convergence behaviour of the objective function for different regularization parameters. Without regularization ($\mu = 0$), the algorithm converges to approximately 10^{-5} after about $10,000$ iterations. However, the introduction of appropriate regularization significantly improves the convergence characteristics. For $\mu = 5 \times 10^{-3}$, the algorithm achieves convergence to a constant value below 10^{-4} within approximately 500 iterations, demonstrating enhanced computational efficiency.

The numerical reconstruction results are presented in Figure 6.12(b) for different values of μ. The unregularized solution ($\mu = 0$) exhibits noticeable oscillations and deviates from the true boundary shape. This behaviour is typical of ill-posed inverse problems when noisy contaminated data is processed without stabilization. Moreover, regularization with

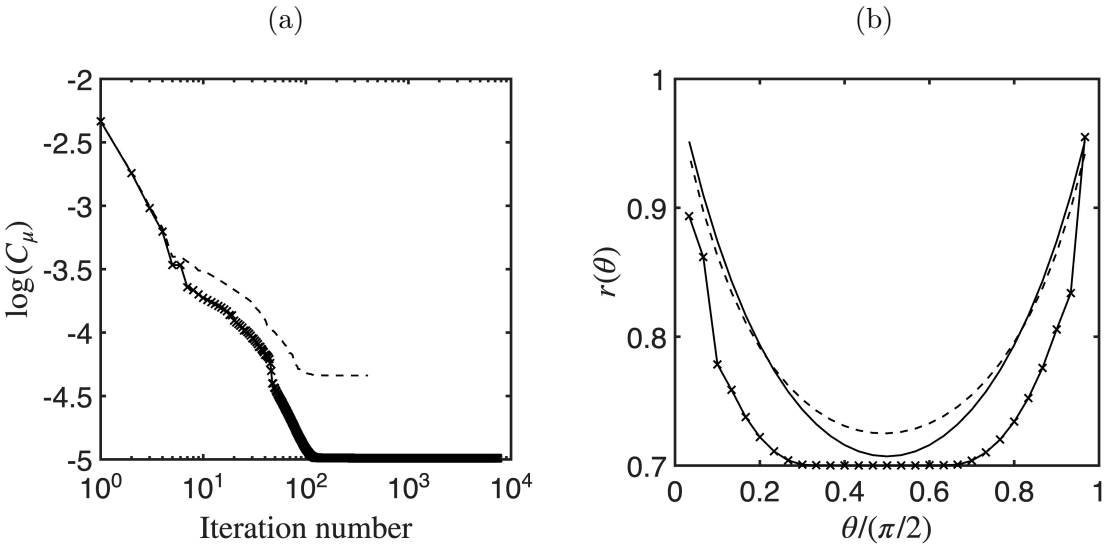

Figure 6.12: Noise p = 1% (Example 2) with LBC. (a) The convergence of the objective function (6.8) for $(\alpha, \beta) = (1, 1)$, corresponding to inverting both the data $h^{(1)}$ and $h^{(2)}$ with the initial guess $r^0(\theta) = 1$, shown for different values of the regularization parameter: $(-\times-)$ for $\mu = 0$ and $(---)$ for $\mu = 5\times10^{-3}$, along with the corresponding (b) numerical reconstruction of $r(\theta)$ in comparison with the exact solution (6.12) shown with (——).

$\mu = 5\times10^{-3}$ provides appropriate balance between data fidelity and smoothness, yielding a reconstruction that closely approximates the exact solution (6.11), while effectively suppressing noise-induced oscillations.

6.5.4 Example 2: Nonlinear boundary condition on Γ_0

6.5.4.1 Noiseless data (convergence)

Figure 6.13 demonstrates the reconstruction performance under the nonlinear boundary condition (6.16) for the noiseless case. Unlike the linear boundary condition (6.13) case, we obtain different convergence characteristics and regularization requirements.

From Figure 6.13(a), the convergence behaviour shows that all regularization parameters achieve similar final convergence levels around 10^{-8}. However, the convergence patterns differ notably: the unregularized case ($\mu = 0$) demonstrates rapid initial convergence but stabilizes at a slightly higher error level. The regularized cases with $\mu = 10^{-4}$ and $\mu = 10^{-5}$ show more consistent convergence profiles, reaching comparable final accuracy within approximately 200 iterations. The reconstruction results in Figure 6.13(b) reveal that all three regularization parameters $\mu \in \{0, 10^{-5}, 10^{-4}\}$ yield nearly identical

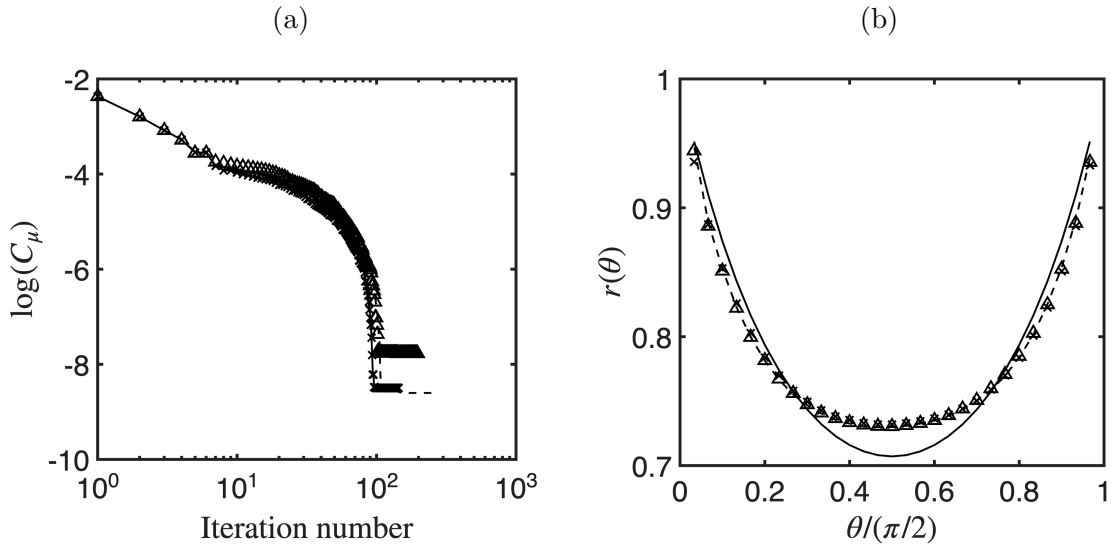

Figure 6.13: No noise, with regularization (Example 2) with NLBC. (a) The convergence of the objective function (6.8) for $(\alpha, \beta) = (1, 1)$ corresponding to inverting both the data $h^{(1)}$ and $h^{(2)}$ with the initial guess $r^0(\theta) = 1$, shown for different values of the regularization parameter: $(- \times -)$ for $\mu = 0$, $(- - -)$ for $\mu = 10^{-5}$ and $(-\Delta-)$ for $\mu = 10^{-4}$, along with the corresponding (b) numerical reconstruction of $r(\theta)$ in comparison with the exact solution (6.12) shown with (——).

reconstructions, which deviate slightly from the exact solution (6.12). The insensitivity of the solution to the regularization parameter suggests that the error is primarily due to discretization or the quality of the initial guess rather than the regularization itself.

This behaviour contrasts with the LBC case, suggesting that the NLBC absorbs the instability effects through its stiff structure, leading to more stable reconstructions even in the absence of regularization.

6.5.4.2 Noisy data with regularization

The stability assessment under noisy conditions (p = 1%) is illustrated in Figure 6.14.

Figure 6.14(a) shows that the convergence of the objective function is quickly achieved in about 100-200 iteration for all regularization parameters $\mu \in \{0, 10^{-5}, 10^{-4}\}$.

The reconstruction of $r(\theta)$, shown in Figure 6.14(b), shows good performance even with no regularization included, consistent with the previous results obtained for Example 1 with the NLBC.

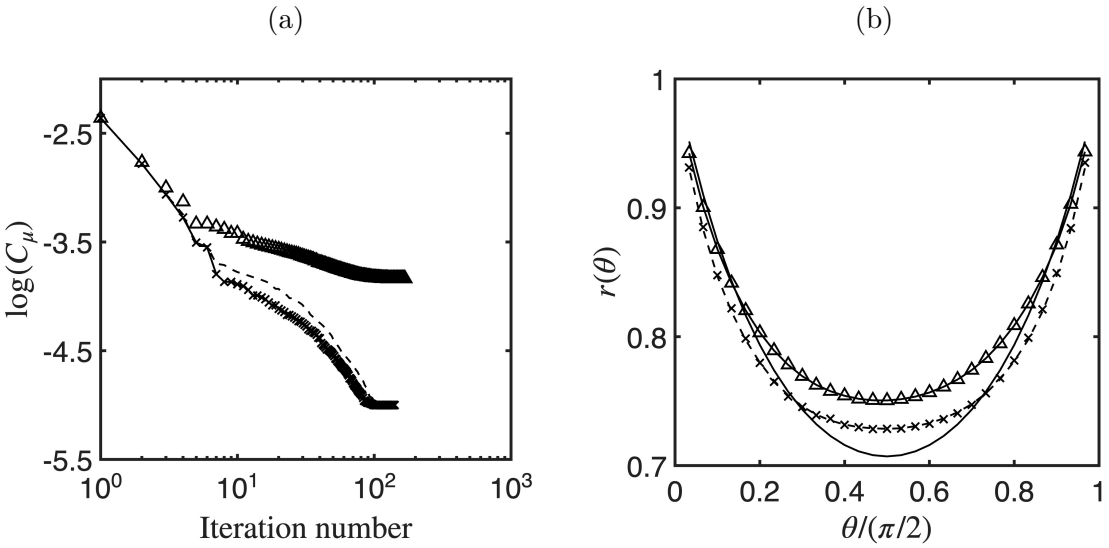

Figure 6.14: Noise p = 1% (Example 2) with NLBC. (a) The convergence of the objective function (6.8) for $(\alpha, \beta) = (1, 1)$, corresponding to inverting both the data $h^{(1)}$ and $h^{(2)}$ with the initial guess $r^0(\theta) = 1$, shown for different values of the regularization parameter: $(- \times -)$ for $\mu = 0$, $(- - -)$ for $\mu = 10^{-4}$ and $(-\Delta-)$ for $\mu = 10^{-2}$, along with the corresponding (b) numerical reconstruction of $r(\theta)$ in comparison with the exact solution (6.12) shown with (——).

6.6 Conclusions

This chapter presented a comprehensive reconstruction analysis of the corroded boundary Γ_0 from two distinct pairs of Cauchy data on Γ_2 generated from the discontinuous flux (2.27) and the smooth flux (2.28). Based on two distinct geometric configurations under both linear and nonlinear Robin boundary conditions, in noiseless and noisy data scenarios, we have established fundamental relationships between geometric complexity, boundary condition formulation and reconstruction stability of the solutions obtained.

The core contribution of this chapter is the implementation of a data fusion technique to ensure the uniqueness and stability of reconstruction of the corroded boundary. By simultaneously integrating measurements from two functionally independent flux configurations (discontinuous and smooth), the framework successfully constrained the search space to retrieve the exact boundary shape.

The investigation revealed that data fusion, utilizing measurements from both discontinuous and smooth flux inputs simultaneously, significantly enhances reconstruction

accuracy compared to single-flux approaches. The combined approach demonstrated superior convergence characteristics and reduced sensitivity to initialization parameters, establishing data fusion as a robust strategy for practical corrosion detection applications.

The geometric complexity of the corroded boundary significantly influences the reconstruction accuracy and convergence behaviour. Simple geometries, such as the quarter-circle configuration in Example 1, generally yield more reliable reconstructions with faster convergence rates. More complex shapes, as demonstrated in Example 2, present additional challenges requiring more sophisticated regularization strategies and potentially larger datasets for accurate reconstruction.

The comparative analysis between single-flux and dual-flux reconstruction approaches conclusively established the superiority of the data fusion methodology enhancing the uniqueness and stability of solution.

The MATLAB computational program for the reconstruction of the corroded boundary Γ_0 is included in Section A.6 of Appendix A.

Chapter 7

Simultaneous reconstruction of the corrosion coefficient and corroded boundary

Building upon the methodologies introduced in Chapters 5 and 6, this chapter tackles the complex inverse problem of simultaneously reconstructing the space-dependent corrosion coefficient λ on the boundary Γ_0 and the geometry of the corroded boundary Γ_0, utilizing over-prescribed measured potential data on Γ_2. The proposed algorithm iteratively updates both the corrosion coefficient and the boundary geometry until the computed potential values on Γ_2 matches the measured data in the least-squares sense. Numerical experiments validate the method's accuracy and stability, demonstrating its potential for practical applications in electrochemistry, material science and industrial corrosion modelling, while extending the individual frameworks of the preceding chapters into a unified approach.

7.1 Introduction

The field of inverse problems governed by the Laplace equation presents fundamental challenges in determining inaccessible physical properties and geometric configurations from boundary measurements, with applications spanning numerous scientific and engineering

disciplines [24, 62, 71]. The preceding two chapters of this thesis have systematically addressed distinct but interconnected classes of such inverse problems in the context of corrosion monitoring for buried pipelines. Chapter 5 focused on the identification of a space-dependent Robin boundary coefficient $\lambda(\theta)$ on a corroded boundary segment Γ_0, under the assumption that the geometry of Γ_0 was known. Conversely, Chapter 6 considered the complementary scenario where the corroded boundary geometry Γ_0 itself was unknown and required reconstruction, while a Robin-type boundary condition was prescribed on it. In both cases, the problems were reformulated as Cauchy problems on the accessible boundary Γ_2 and solved numerically using the BEM combined with appropriate regularisation strategies [8, 100].

However, real corrosion problems are much more complex than simple cases. When pipes corrode, two things happen simultaneously: the geometric shape changes and the electrochemical properties change. These effects occur together in the same corroded region. Mathematically, we use the spatially varying Robin-type corrosion coefficient $\lambda(\theta)$ to represent the changing electrochemical response at different locations. This requires solving a comprehensive problem: simultaneously determining both the unknown corroded boundary Γ_0 and the space-dependent coefficient $\lambda(\theta)$ from measurement data on the accessible boundary Γ_2. This class of problems, known as Robin inverse problems, represents one of the most challenging areas in inverse theory due to the intrinsic coupling between geometric and material parameters [45, 85].

The joint identification problem addressed in this chapter is inherently more ill-posed than either of the individual sub-problems examined in the preceding chapters. It combines the instability associated with recovering unknown geometries with the sensitivity inherent in reconstructing spatially varying Robin parameters, while introducing additional nonlinearity due to the coupling between the two unknowns [45]. Furthermore, a single set of boundary measurements is frequently insufficient to uniquely identify both the boundary shape and the Robin coefficient [24, 45, 62, 86], making the problem significantly more sensitive to measurement noise and requiring sophisticated mathematical treatments. Achieving uniqueness often necessitates multiple independent measurements

[1, 2, 11, 86]. Furthermore, small perturbations in the measured Cauchy data, such as electrical potentials or current densities recorded on Γ_2, may lead to significant amplification of errors in the simultaneous recovery of both Γ_0 and $\lambda(\theta)$. In this chapter, we extend the methodologies from previous chapters to solve this coupled boundary-and-coefficient identification problem. We employ the BEM, which is well-suited for this application due to its boundary-only discretization and flexibility in handling complex boundary conditions [18, 19]. The BEM's proven effectiveness for inverse boundary problems in heat conduction and elasticity [72, 63] makes it an ideal choice for corrosion detection applications.

The mathematical framework developed herein utilizes shape derivatives of objective functions with respect to both the boundary geometry and the Robin coefficient, implemented through gradient-based iterative schemes. The approach leverages boundary integral equation methods for their computational efficiency in handling boundary changes and reducing the computational domain complexity. Through this unified methodology, we aim to demonstrate improved robustness and accuracy in the simultaneous identification of complex corroded boundaries and their associated electrochemical parameters, particularly under realistic measurement conditions that include noise and limited data availability.

The chapter is organised as follows. Section 7.2 presents the mathematical formulation of the coupled inverse boundary and coefficient identification problem. Section 7.3 describes the BEM discretisation approach. Section 7.4 introduces the unified iterative minimization algorithm for corrosion reconstruction. Numerical results and discussion are provided in Section 7.5, illustrating the effectiveness of the approach under both noise-free and noisy data conditions. Finally, Section 7.6 summarises the main findings.

7.2 Mathematical formulation

The inverse problems studied in Sections 5.2 and 6.2 addressed, respectively, the identification of the unknown corrosion coefficient $\lambda(\theta)$ on a known corroded boundary Γ_0 and the reconstruction of the unknown corroded boundary Γ_0 with known $\lambda(\theta)$. Here

we consider the more general and challenging problem where both the Robin coefficient $\lambda(\theta)$ and the boundary geometry Γ_0 are unknowns and must be simultaneously identified from boundary measurements. The mathematical model, see again Figure 6.1, remains governed by the Laplace equation in the domain Ω,

$$\Delta u^{(i)} = 0 \qquad \text{in} \quad \Omega, \tag{7.1}$$

subject to the boundary conditions

$$\frac{\partial u^{(i)}}{\partial n} = 0 \quad \text{on } \Gamma_1 \cup \Gamma_3 \cup \Gamma_4, \tag{7.2a}$$

$$\frac{\partial u^{(i)}}{\partial n} = \lambda(\theta) f(u) \quad \text{on } \Gamma_0, \tag{7.2b}$$

$$\frac{\partial u^{(i)}}{\partial n} = g^{(i)} \quad \text{on } \Gamma_2, \tag{7.2c}$$

$$u^{(i)} = h^{(i)} \quad \text{on } \Gamma_2, \tag{7.2d}$$

for $i = 1, 2$, where $g^{(1)}$ and $g^{(2)}$ are two functionally independent fluxes, at least one of which is positive. Here, $f(u)$ is given either by Newton's law or the Butler–Volmer law, as previously introduced in (5.5)–(5.6) and (6.3a)–(6.3b).

The goal is to determine simultaneously the pair $(\Gamma_0, \lambda(\theta))$ within admissible classes of geometries and coefficients such that the equations (7.1), (7.2a)–(7.2d) are satisfied, where:

- $\lambda : [0, \pi/2] \to \mathbb{R}_+$ represents the space-dependent corrosion coefficient
- $r : [0, \pi/2] \to \mathbb{R}_+$ represents the radial function defining the corroded boundary geometry in (6.6)

Compared with the formulations in Sections 5.2 and 6.2, this problem exhibits higher ill-posedness and stronger nonlinearity, as both geometric and functional unknowns interact in the Robin boundary condition (7.2b).

7.3 The boundary element method (BEM)

Building on the formulations of Sections 5.3 and 6.3, we now extend the BEM to the simultaneous identification of both the corrosion coefficient $\lambda(\theta)$ and the geometry of the corroded boundary Γ_0. The strategy combines the coefficient reconstruction framework of Section 5.3 with the geometric parametrization approach of Section 6.3.

Applying the boundary integral representation of the Laplace equation (7.1), together with the boundary conditions (7.2a)–(7.2c), yields an integral equation of the form

$$\eta(\underline{p})u^{(i)}(\underline{p}) = \int_{\Gamma_0} G(\underline{p};\underline{p}')\,\lambda(\underline{p}')\,f(u^{(i)}(\underline{p}'))\,ds(\underline{p}') + \int_{\Gamma_2} G(\underline{p};\underline{p}')\,g(\underline{p}')\,ds(\underline{p}') - \int_{\partial\Omega} u^{(i)}(\underline{p}')\,\frac{\partial G}{\partial n(\underline{p}')}(\underline{p};\underline{p}')\,ds(\underline{p}'), \quad \underline{p}\in\overline{\Omega},\ i=1,2. \tag{7.3}$$

Discretising the boundary $\partial\Omega$ into $M = 7N$ elements and applying (7.2a) and (7.2c) at the boundary element nodes we obtain the algebraic BEM system of equations

$$\sum_{j=1}^{N} A_{kj}\,\lambda_j\,f(u_j^{(i)}) + \sum_{j=1}^{M} B_{kj}\,u_j^{(i)} = -\sum_{j=2N+1}^{4N} A_{kj}\,g_j =: b_k, \qquad k=\overline{1,M},\ i=1,2. \tag{7.4}$$

To allow for the reconstruction of Γ_0, we employ the star-shaped parameterization introduced in (6.6). Thus, the BEM system of equations (7.4) now couples two sets of unknowns: the Robin coefficients λ_j, which appear multiplicatively with $f(u_j^{(i)})$, and the geometric parameters $\underline{r}$, which enter through the parametrization of Γ_0 and thereby modify both A_{kj} and B_{kj}. This coupling makes the inverse problem highly nonlinear and ill-posed, requiring simultaneous updates of $\lambda(\theta)$ and Γ_0 within the optimization framework described in the next section.

7.4 The iterative method

In this chapter, we merge the approaches presented in Chapters 5 and 6 to formulate a comprehensive iterative minimization framework for the reconstruction of both the corrosion coefficient $\lambda(\theta)$ on Γ_0 and the shape of the corroded boundary.

Given the Cauchy data pairs $(h^{(i)}, g^{(i)})$ on Γ_2, for $i = 1, 2$, the inverse problem aims to determine the unknown parameters $\lambda(\theta)$ and the boundary Γ_0. Following Chapters 5 and 6, we define the combined least-squares functional as

$$F(\underline{r}, \underline{\lambda}) = \sum_{i=1}^{2} \|u^{(i)}(\underline{r}, \underline{\lambda})|_{\Gamma_2} - h^{(i)}\|^2_{L^2(\Gamma_2)}, \tag{7.5}$$

where $u^{(i)}(\underline{r}, \underline{\lambda})$ is the solution of the direct problem corresponding to flux $g^{(i)}$ and the current approximation of the boundary Γ_0 and corrosion coefficient λ.

To ensure stability against noise in the measured data, we incorporate Tikhonov regularization. Combining the data misfit and regularization of Chapters 5 and 6, the fully regularized functional to be minimized is

$$F_\mu(\underline{r}, \lambda) = F(\underline{r}, \lambda) + R_1(\underline{r}) + R_2(\lambda), \tag{7.6}$$

where

$$R_1(\underline{r}) = \mu_0 \sum_{j=1}^{N-1} \left(r_j - r_j^0\right)^2, \tag{7.7}$$

with $r_j = r(\theta_j)$ are the discretized radial boundary values and r_j^0 are the initial guesses for $j = \overline{1, N-1}$.

and

$$R_2(\lambda) = \mu_1 \sum_{j=2}^{N} (\lambda_j - \lambda_{j-1})^2, \tag{7.8}$$

with $\lambda_j = \lambda(\tilde{\theta}_j)$ are the discretised values of λ for $j = \overline{1, N}$.

In discretized form, using $2N$ boundary elements on Γ_2 and N discretization points on Γ_0, equation (7.6) becomes

$$\begin{aligned} F_\mu(\underline{r}, \lambda) \approx \frac{2}{2N} \sum_{j=2N+1}^{4N} \left[\left(u_j^{(1)}(\underline{r}, \lambda) - h_j^{(1)}\right)^2 + \left(u_j^{(2)}(\underline{r}, \lambda) - h_j^{(2)}\right)^2 \right] \\ + \mu_0 \sum_{j=2}^{N-1} \left(r_j - r_j^0\right)^2 + \mu_1 \sum_{i=2}^{N} (\lambda_j - \lambda_{j-1})^2. \end{aligned} \tag{7.9}$$

The constrained minimization of (7.9) is performed iteratively in MATLAB using

`lsqnonlin`, starting from initial guesses $r_j^0 = 1.1$ for $j = \overline{1, N-1}$ and $\lambda_j^{(0)} = \frac{\tilde{\theta}_j}{\pi/2}$ for $j = \overline{1, N}$. The bounds $0.5 = r_{\min} \leq r_j \leq r_{\max} = 1.5$ and $0 = \lambda_{\min} \leq \lambda_i \leq \lambda_{\max} = 1$ are imposed to ensure physically meaningful reconstructions. The exact solutions to be retrieved are

$$\lambda(\theta) = \sin(\theta) \quad \text{and} \quad r(\theta) = 1 \quad \text{for } \theta \in \left[0, \frac{\pi}{2}\right]. \tag{7.10}$$

7.5 Numerical results and discussion

Two independent Cauchy data sets were employed for the reconstruction process. The results are presented and compared for both the LBC (5.7) and the NLBC (5.21) at noise levels $\mathrm{p} \in \{0, 0.1, 0.5, 1\}\%$ in Figures 7.1 and 7.2, and Table 7.1. The convergence behaviour from Figures 7.1(a) and 7.2(a) reveals that both cases achieve convergence. For $\mathrm{p} = 0$, the objective function converges to approximately 10^{-12} after 400 iterations. As noise increases, the objective function rises to 10^{-7} for $\mathrm{p} = 0.1\%$ and to 10^{-5} for $\mathrm{p} = 0.5\%$.

The identification of the coefficient $\lambda(\theta)$ in Figures 7.1(b) and 7.2(b) further highlights the performance gap between the two types of boundary conditions. For low noise levels ($\mathrm{p} \leq 0.1\%$), λ is retrieved reasonably in both cases (5.7) and (5.21). However, for $\mathrm{p} = 0.5\%$, the reconstruction in the case of the LBC exhibits significant spurious oscillations that deviate substantially from the exact solution, whereas in the case of the NLBC the solution maintains a smooth and accurate profile.

The reconstruction of $r(\theta)$ in Figures 7.1(c) and 7.2(c) shows that in both LBC and NLBC cases the solution is recovered accurately for $\mathrm{p} \in \{0, 0.1\}\%$. However, for $\mathrm{p} = 0.5\%$, the solution exhibits pronounced oscillations in case of the LBC, while in case of the NLBC the numerical solution maintains a smoother profile with significantly reduced perturbations.

Table 7.1 shows the quantitative performance comparison between the numerical solutions obtained in the cases of LBC and NLBC across different noise levels. The error

Table 7.1: The error norms $E_r = \|\underline{r}_{exact} - \underline{r}_{num}\|_{\ell^2(\Gamma_0)}$ and $E_\lambda = \|\underline{\lambda}_{exact} - \underline{\lambda}_{num}\|_{\ell^2(\Gamma_0)}$ at different levels of noise.

	p	E_r	E_λ
initial	0	0.5385	0.8264
LBC	0	0.1532	0.3816
	0.1	0.1914	0.5875
	0.5	0.8297	1.8869
	1	1.0101	2.6569
NLBC	0	0.1653	0.4570
	0.1	0.1876	0.6374
	0.5	0.2192	0.7667
	1	0.2881	0.7230

norms used to assess numerical accuracy are the discrete ℓ^2 norms [57] defined as

$$E_r = \|\underline{r}_{exact} - \underline{r}_{num}\|_{\ell^2(\Gamma_0)} = \sqrt{\sum_{i=1}^{N}(r_{exact}(\theta_i) - r_{num}(\theta_i))^2},$$

and similarly for E_λ. These discrete norms are consistent with the objective function minimized by the optimization algorithm. For the initial guesses $r_j^0 = 1.1$ for $j = \overline{1, N-1}$ and $\lambda_j^{(0)} = \frac{\tilde{\theta}_j}{\pi/2}$ for $j = \overline{1, N}$ we obtain $E_{r^{(0)}} = 0.5385$ and $E_{\lambda^{(0)}} = 0.8264$, respectively.

For p = 0, the numerical solutions successfully reduced both errors below the initial values: the LBC gives $E_r = 0.1532$ and $E_\lambda = 0.3816$, while the NLBC gives $E_r = 0.1653$ and $E_\lambda = 0.4570$. However, as noise increases, the gap widens significantly. For p = 0.5%, the LBC gives $E_r = 0.8297$ and $E_\lambda = 1.8869$, indicating that the final solution is worse than the initial guess. In contrast, the NLBC gives $E_r = 0.2192$ and $E_\lambda = 0.7667$, showing modest improvement over $E_{\lambda^{(0)}}$.For p = 1% (high noise), the LBC yields completely deteriorated result of $E_r = 1.0101$ and $E_\lambda = 2.6569$ both substantially worse than the initial guess, whereas the NLBC yields results that remain stable at $E_r = 0.2881$ and $E_\lambda = 0.7230$.

The accuracy of the numerical results can be improved by approximating the polar radius $r(\theta)$ and the corrosion coefficient $\lambda(\theta)$ using finite-dimensional trigonometric polynomials as

$$r(\theta) = a_0 + \sum_{k=1}^{K} a_k \cos(k\theta) + \sum_{k=1}^{K} b_k \sin(k\theta), \tag{7.11}$$

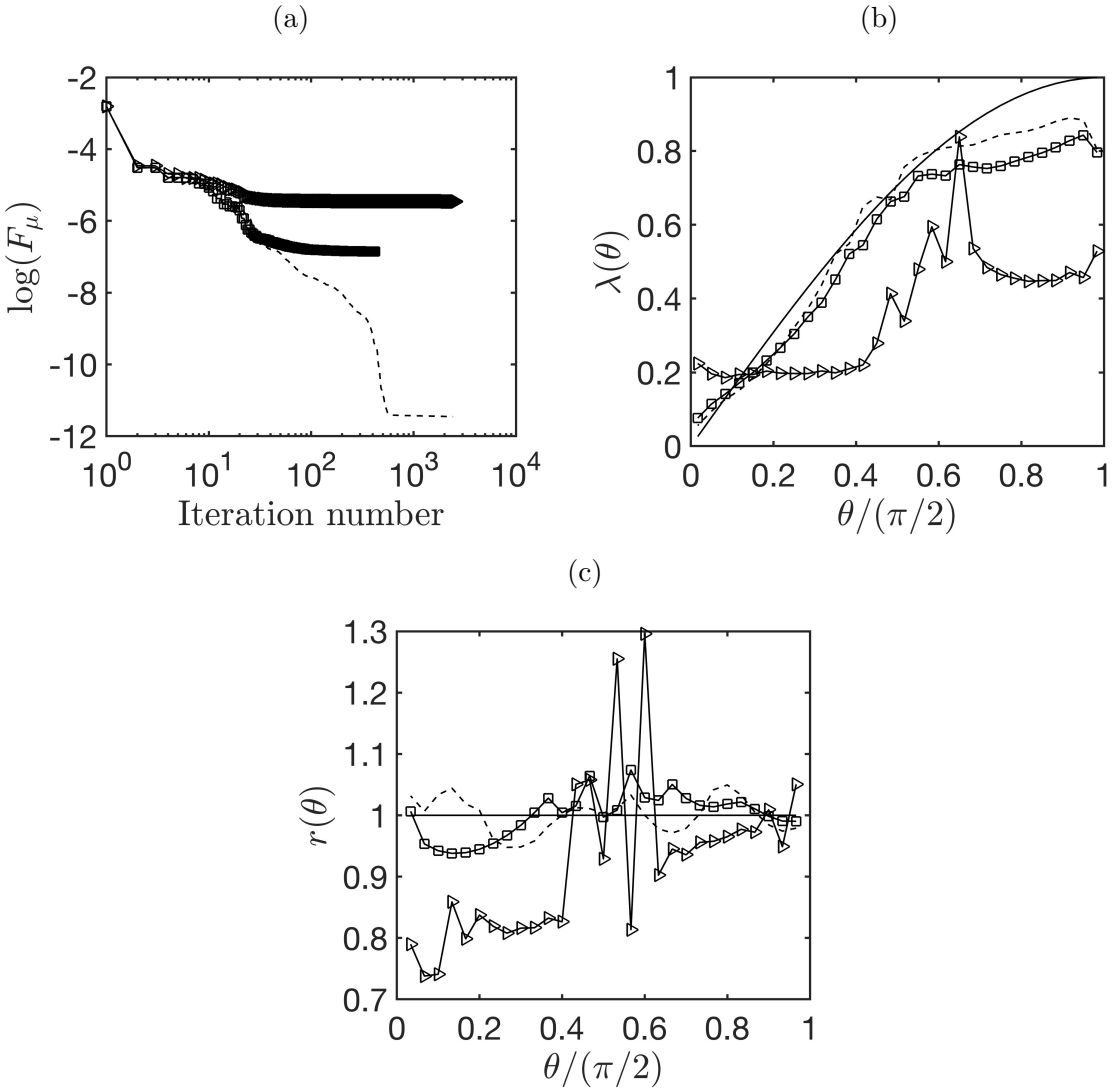

Figure 7.1: Numerical results obtained for the linear boundary condition (5.7) for different noise levels p: $(- - -)$ for p = 0, $(-\square-)$ for p = 0.1%, and $(-\triangle-)$ for p = 0.5%. (a) The convergence of the objective function (7.9), (b) numerical identification of $\lambda(\theta)$ in comparison with the exact solution shown with (——), and (c) numerical reconstruction of $r(\theta)$ in comparison with the exact solution shown with (——).

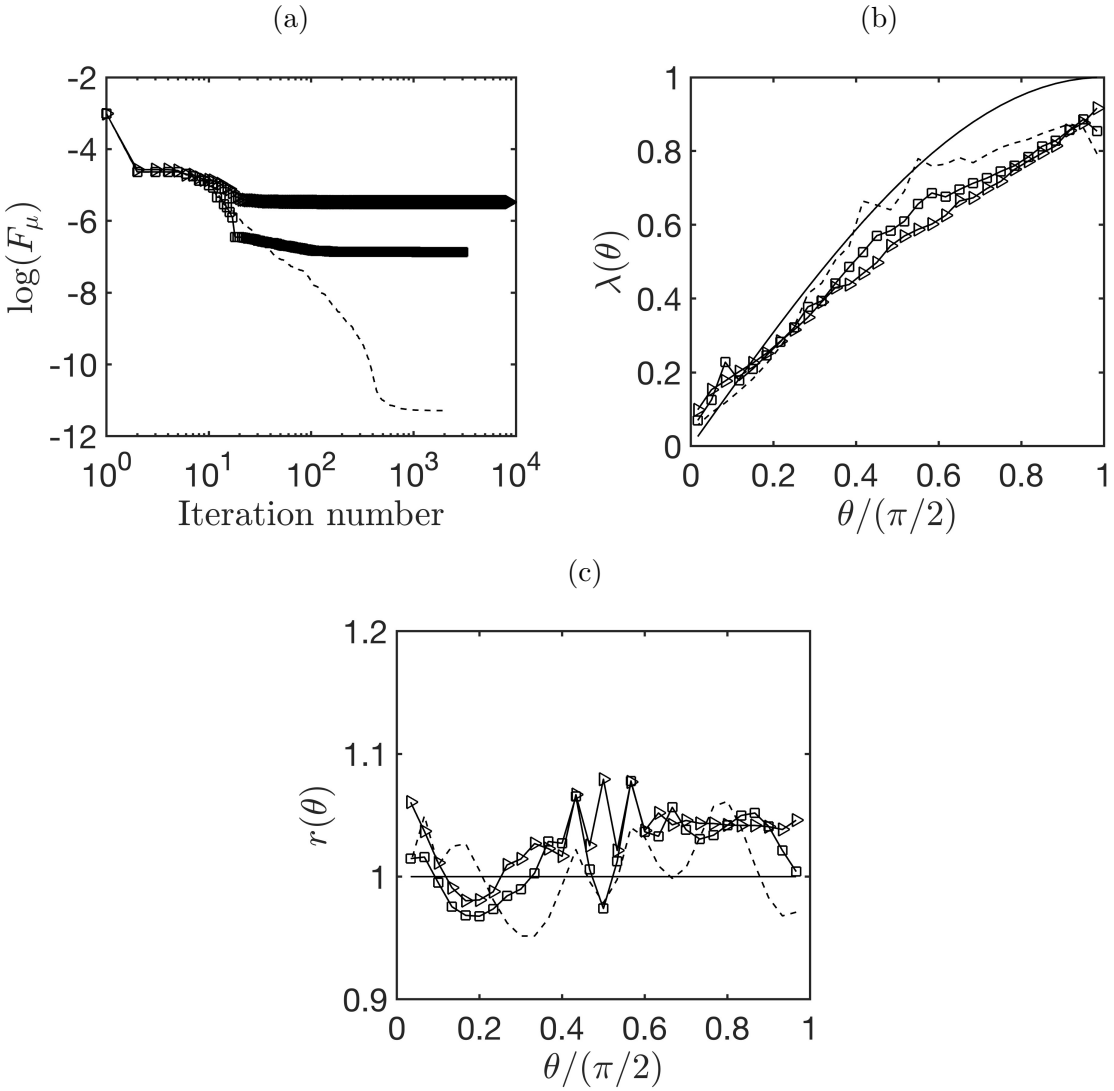

Figure 7.2: Numerical results obtained for the nonlinear boundary condition (5.21) for different noise levels p: $(---)$ for p = 0, $(-\square-)$ for p = 0.1%, and $(-\triangle-)$ for p = 0.5%. (a) The convergence of the objective function (7.9), (b) numerical identification of $\lambda(\theta)$ in comparison with the exact solution shown with (——), and (c) numerical reconstruction of $r(\theta)$ in comparison with the exact solution shown with (——).

$$\lambda(\theta) = c_0 + \sum_{k=1}^{K} c_k \cos(k\theta) + \sum_{k=1}^{K} d_k \sin(k\theta), \tag{7.12}$$

where $\underline{a} = (a_k)_{k=\overline{0,K}}$, $\underline{b} = (b_k)_{k=\overline{1,K}}$, $\underline{c} = (c_k)_{k=\overline{0,K}}$ and $\underline{d} = (d_k)_{k=\overline{1,K}}$ are real coefficients to be determined. The lower and upper bounds on these variables are taken as:

$$\underline{l} = (0.5, -0.5, -0.5, 0, -1, -1) \leq (a_0, a_1, b_1, c_0, c_1, d_1) \leq (1.5, 0.5, 0.5, 1, 1, 1.5) = \underline{u},$$

and the initial guesses

$$(a_0^{(0)}, a_1^{(0)}, b_1^{(0)}, c_0^{(0)}, c_1^{(0)}, d_1^{(0)}) = (1.1, 0, 0, 0, 0, 0.5). \tag{7.13}$$

Applying the reduced model, see Equations (7.11) and (7.12), improves numerical stability and computational efficiency. The convergence results shown in Figures 7.3(a) and 7.4(a) confirm that both the LBC and NLBC yield results that converge rapidly and remain stable across all noise levels. For the LBC, the objective function reaches approximately 10^{-10} in the noise-free case and maintains constant values near 10^{-6} and 10^{-5} for p = 0.5% and 1%, respectively. For the NLBC, results exhibit comparable convergence characteristics, indicating that the reduced representation effectively mitigates sensitivity to noise.

The reconstructions of $\lambda(\theta)$, presented in Figures 7.3(b) and 7.4(b), confirm that for both boundary conditions we retain high accuracy and stability of numerical results even under increased noise. Unlike the full model, the reduced representation prevents the appearance of spurious oscillations and ensures smooth, physically meaningful estimates.

The reconstructions of $r(\theta)$ in Figures 7.3(c) and 7.4(c) show remarkable accuracy at all noise levels. For both boundary condition cases, we recover the solution with high fidelity when data is exact, i.e., p = 0. For p = 0.5% and even for p = 1% noisy data, the model reduction acts as an effective implicit regularization, eliminating spurious oscillations.

Table 7.2 validates the quantitative advantage of the reduced formulation. The errors are calculated at the same points θ_i for $i = \overline{1, N-1}$ and $\tilde{\theta}_i$ for $i = \overline{1, N}$. For p = 0, the LBC case yields $E_r = 0.1046$ and $E_\lambda = 0.2771$, whereas the NLBC case yields $E_r = 0.0259$

Table 7.2: The error norms $E_r = \|\underline{r}_{exact} - \underline{r}_{num}\|_{\ell^2(\Gamma_0)}$ and $E_\lambda = \|\underline{\lambda}_{exact} - \underline{\lambda}_{num}\|_{\ell^2(\Gamma_0)}$ at different levels of noise in case of model reduction.

	p	E_r	E_λ
LBC	0	0.1046	0.2771
	0.5	0.1552	0.3758
	1	0.2740	0.6750
NLBC	0	0.0259	0.1222
	0.5	0.1498	0.3611
	1	0.2941	0.7311

Table 7.3: The numerically retrieved Fourier coefficients for the LBC and NLBC and various percentages of noise p $\in \{0, 0.5, 1\}\%$.

	p	a_0	a_1	b_1	c_0	c_1	d_1
LBC	0	1.0775	-0.0388	-0.0938	0.3488	-0.3146	0.7974
	0.5	1.1013	-0.1098	-0.0269	0.0125	0.0322	0.8725
	1	1.1545	-0.1756	-0.0303	0.0433	0.0405	0.7586
NLBC	0	1.0776	-0.0497	-0.0753	0.2229	-0.1988	0.8648
	0.5	1.0675	-0.0842	0.0030	0.0486	0.0035	0.8390
	1	1.1170	-0.1483	0.0076	0.0290	0.0570	0.7506

and $E_\lambda = 0.1222$. For p $= 0.5\%$ noise, the LBC case yields $E_r = 0.1552$ and $E_\lambda = 0.3758$, whilst the NLBC case gives $E_r = 0.1498$ and $E_\lambda = 0.3611$. Even for p $= 1\%$ noise, both boundary conditions yield results maintain stable accuracy, with the LBC yielding $E_r = 0.2740$ and $E_\lambda = 0.6750$, and the NLBC producing $E_r = 0.2941$ and $E_\lambda = 0.7311$.

Table 7.3 presents the Fourier coefficients for remain stable at different noise levels. The results for the a_0 coefficient around 1.05–1.15, correctly recovering the circular geometry (6.11). The oscillatory terms a_1 and b_1 remain negligible (≈ -0.01 to -0.17), as expected. The sine coefficient d_1 ranges from 0.76 to 0.91, reasonably approximating the dominant component of $\sin(\theta)$ in the expression (5.19).

7.6 Conclusion

This chapter presented the ultimate challenge in corrosion modeling: the joint identification of both the space-dependent corrosion coefficient $\lambda(\theta)$ and the polar radius $r(\theta)$ of the corroded boundary Γ_0 from over-prescribed potential measurements on the accessible boundary Γ_2. This problem is highly nonlinear and severely ill-posed due to the

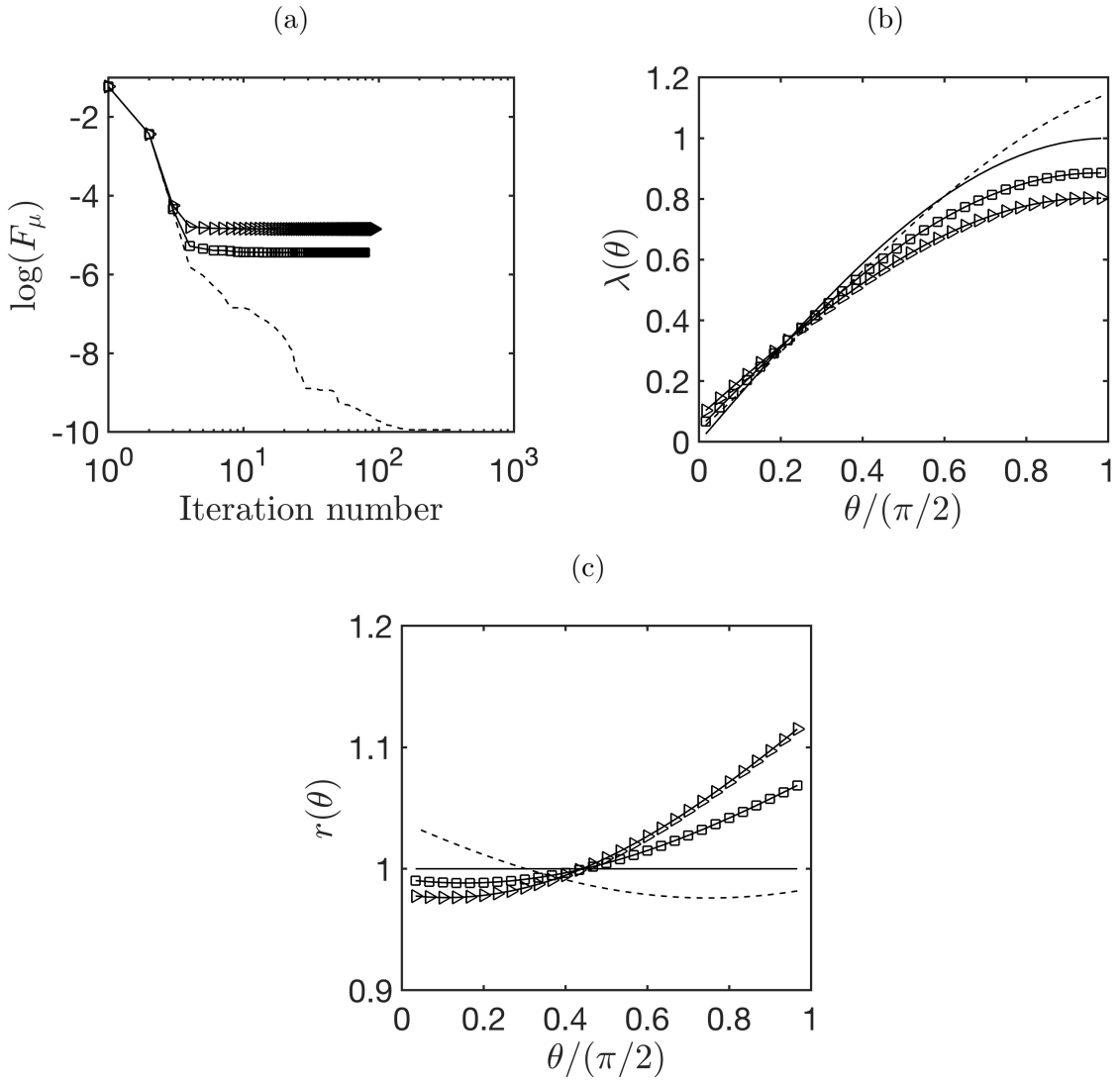

Figure 7.3: Numerical results obtained for the linear boundary condition (5.7) in case of model reduction. (a) The convergence of the objective function (7.9) for different noise levels p: $(---)$ for p = 0, $(-\square-)$ for p = 0.5% and $(-\triangle-)$ for p = 1%, along with the corresponding: (b) numerical identification of $\lambda(\theta)$ in comparison with the exact solution shown with (——), and (c) numerical reconstruction of $r(\theta)$ in comparison with the exact solution shown with (——).

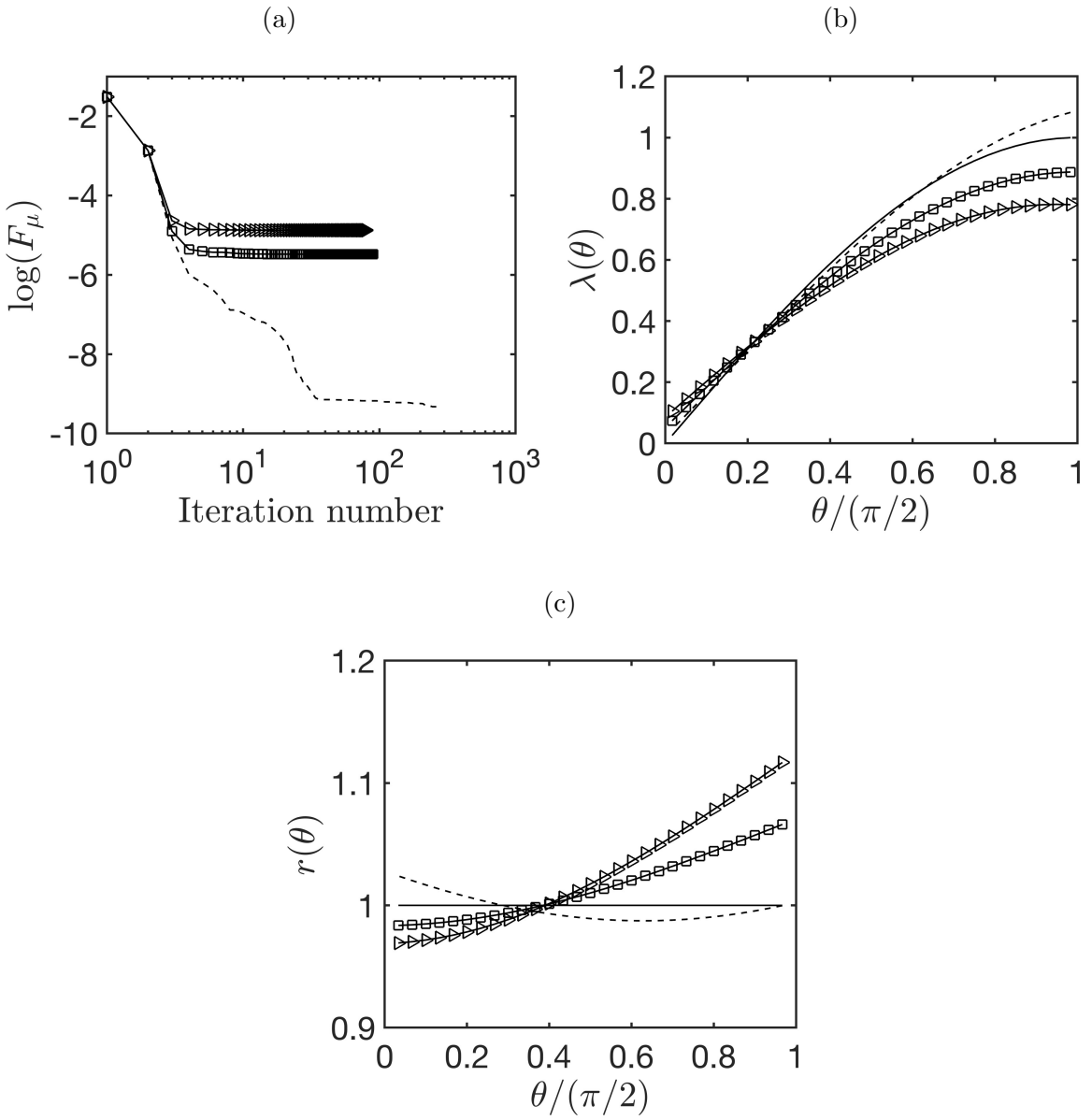

Figure 7.4: Numerical results obtained for the nonlinear boundary condition (5.21) in case of model reduction. (a) The convergence of the objective function (7.9) for different noise levels p: $(- - -)$ for p = 0, $(-\square-)$ for p = 0.5% and $(-\triangle-)$ for p = 1%, along with the corresponding: (b) numerical identification of $\lambda(\theta)$ in comparison with the exact solution shown with (——), and (c) numerical reconstruction of $r(\theta)$ in comparison with the exact solution shown with (——).

coupling between geometry and material unknowns. By combining and extending the methodologies developed in Chapters 5 and 6, we have established a unified approach that addresses the realistic scenario where both parameters are unknown in practical corrosion assessment applications.

The comparative study between the linear and nonlinear boundary condition formulations demonstrated that the nonlinear model exhibits superior robustness and stability under noisy data, particularly at higher noise levels. The nonlinear formulation effectively mitigates oscillations and provides smoother reconstructions of both $r(\theta)$ and $\lambda(\theta)$ compared with the linear model. The MATLAB computational code is included in Section A.7 of Appendix A.

This chapter represents the culmination and main original contribution of the thesis. It introduced a Model Reduction strategy using trigonometric polynomials to overcome the severe ill-posedness of the joint inverse problem. By reducing the number of unknowns from 59 to 6, the framework acted as an implicit regularizer, suppressing non-physical oscillations and ensuring stable convergence even under high noise levels. This established a unified and validated numerical framework capable of addressing the most realistic and complex scenarios in non-destructive engineering assessments.

Chapter 8

Conclusions

8.1 Initial investigation

The initial investigation has contributed to the foundational knowledge of the field of corrosion modeling by establishing the computational limits and reliability of boundary-integral techniques under realistic electrochemical conditions. A primary contribution of this initial research, detailed in Chapter 2, was the resolution of the direct problem in scenarios where no exact analytical solution was available to describe the potential distribution in the solution domain. By solving the direct problem using two distinct numerical approaches, specifically the Boundary Element Method (BEM) and the meshless Method of Fundamental Solutions (MFS), this work provided a rigorous benchmark numerical verification of the BEM. Next, the work of Chapter 3 investigated a perfectly conducting inclusion within an annular domain by solving both direct and inverse problems. A critical knowledge contribution was the systematic validation of the BEM's accuracy by comparing numerical results to known exact solutions, followed by successfully applying the method to complex cases where analytical solutions were unavailable. These studies established a quadratic order of convergence and effectively mapped how variations in current flux impact potential distribution, providing a vital baseline for interpreting indirect manifestations of corrosion in inaccessible regions. This work was presented at the Thirteenth UK conference on Boundary Integral Methods and published in its conference

proceedings [6].

Building upon the validated BEM framework from the direct problem analysis of Chapter 2, Chapter 3 marks a transition toward inverse problem formulations by introducing general inverse methodologies for parameter identification and boundary reconstruction. This chapter serves as a bridge between the direct problem foundation and the detailed inverse problem investigations that follow. The subsequent Chapters (4–7) return to the quarter-pipe geometry established in Chapter 2, systematically exploring different inverse problem formulations using the computationally efficient BEM approach validated in the initial studies. While Chapter 3 problems remain linear, they established the computational framework and validated the BEM's accuracy for the potential field equation governing corrosion processes. This provided confidence for tackling the nonlinear inverse problems in subsequent chapters.

While the direct linear solver generates a high resolution for $N = 90$, the inverse reconstruction with $N = 30$ is adopted for several critical considerations: first, this choice avoids committing an inverse crime [65] by ensuring that the discretization used for data generation differs from that employed in the reconstruction process. Second, comparative analysis reveals that the reconstruction accuracy for $N = 30$ is virtually identical to that obtained with $N = 90$, as demonstrated in Chapter 2 in Table 2.3. Third, even at $N = 30$, the computational cost of the iterative inverse solver can exceed 12 hours for complex problems, suggesting that higher discretizations would result in computationally intractable execution times. Therefore, $N = 30$ represents a suitable compromise between accuracy and computational efficiency.

8.2 Quarter-pipe geometry

The core contribution lies in Chapters 4–7, which systematically address increasingly complex inverse problems using a unified quarter-pipe geometry. This geometry exploits symmetry to reduce computational cost while maintaining physical relevance to real corrosion scenarios, see Figure 8.1. The progression of complexity across these chapters is summarized in Table 8.1.

Table 8.1: Summary of problem features and regularization approaches in Chapters 4–7.

Chapter	λ known?	Γ_0 known?	Regularization method	Heat flux on Γ_2
4	Yes	Yes	TSVD	discontinuous
5	No	Yes	Tikhonov 1^{st} order	discontinuous
6	Yes	No	Tikhonov 0^{th} order	discontinuous & continuous
7	No	No	Tikhonov 0^{th} for r & 1^{st} for λ order	discontinuous & continuous

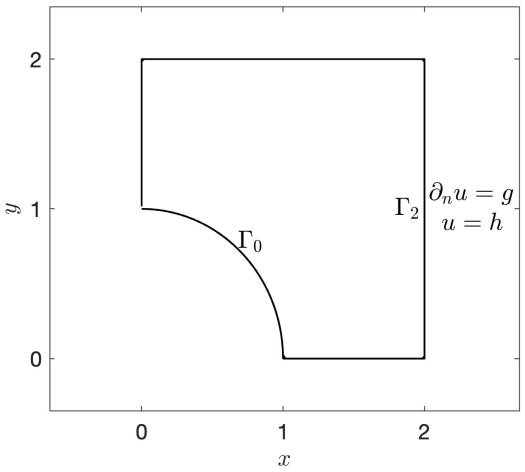

Figure 8.1: The geometry of the domain.

Chapter 4 demonstrated the effectiveness of TSVD regularization in solving Cauchy problems for Laplace's equation. This regularization technique addressed the fundamental challenge of numerical instability that characterizes inverse problems. The comparison between regularized and unregularized solutions clearly illustrated TSVD's impact. Without regularization, the numerical solution became completely unstable, with massive oscillations that rendered the results physically meaningless. TSVD regularization transformed this instability into stable, accurate solutions with errors below 1% for the boundary potential u and approximately 5% for its normal derivative $\partial_n u$. Under noisy conditions, TSVD maintained solution stability while adapting to increasing noise levels. The normal derivative $\partial_n u$ consistently exhibited higher sensitivity to noise compared to the boundary potential u, though this sensitivity difference decreased as noise levels increased. The method demonstrated robust performance across practical measurement uncertainty ranges, making it suitable for real-world electromagnetic corrosion detection applications. The key contribution lay in TSVD's ability to provide systematic criteria for truncation parameter selection through singular value spectrum analysis. This data-driven approach enabled informed parameter selection based on noise characteristics, offering a more principled alternative to arbitrary parameter tuning while providing reliable solutions for industrial non-destructive testing applications.

Chapter 5 advanced to coefficient identification problems where spatially-varying corrosion parameters $\lambda(\theta)$ must be reconstructed from boundary measurements. This scenario represents practical situations where corrosion severity varies along the boundary, requiring reconstruction of a function rather than simple parametric constants. The mathematical formulation leads to a nonlinear least-squares problem requiring iterative solution with BEM as the forward solver. First-order Tikhonov regularization was employed to ensure stable solutions while preserving the spatial variation characteristics of the corrosion coefficient. Two boundary condition formulations on Γ_0 were investigated: a LBC that demonstrated robust performance with minimal regularization requirements, and the NLBC (Butler-Volmer law), which introduced additional computational challenges and exhibited slower convergence toward accurate solutions. The method achieved

accurate reconstruction of $\lambda(\theta)$ under clean data conditions with an optimal regularization parameter of $\mu = 10^{-5}$, while maintaining stability when subjected to 1% noise in the measured data by adjusting the regularization parameter to $\mu = 10^{-3}$. This extends previous work on electrochemical parameter identification, which typically assumed constant coefficients, to handle realistic spatial variations in material properties.

Chapter 6 tackled the geometric inverse problem — reconstructing the corroded boundary Γ_0 from external measurements while assuming known material properties. This represents one of the most challenging applications due to the highly nonlinear relationship between the boundary shape and potential measurements. A data fusion technique was employed that integrated measured potential data on the accessible boundary Γ_2 to constrain the shape reconstruction process, combining two measurement sets to enhance the robustness and uniqueness of the inverse solution. Two boundary condition formulations were systematically investigated: the LBC and the NLBC (Butler–Volmer equation). Zeroth-order Tikhonov regularization was employed to maintain stability while preserving geometric features of the reconstructed boundary. The numerical results conclusively demonstrated that the accuracy and stability of the geometric inverse problem were primarily determined by the selection of the initial guess, with the choice of boundary condition formulation. For the quarter-circle unknown configuration under LBC, the well-chosen initial guess $r^{(0)} = 1.1$ achieved consistent reconstruction error below 4% across both clean and noisy data, whereas the poorly-chosen straight-line initialization failed, producing error norms of 15.4% (clean) and 97.9% (1% noise) — a performance degradation exceeding 80%. This illustrated that formulation choice became nearly inconsequential when confronted with an inadequate initial approximation. Similarly, under the NLBC formulation for the quarter-circle geometry, the superior initial guess $r^{(0)} = 1.1$ achieved optimal results (1.3% for clean data and 3.0% for 1% noisy data), while the alternative initialization yielded 83.0% (for clean data) and 93.7% (for 1% noisy data) errors. For the straight-line unknown boundary Γ_0 with the initial guess $r^{(0)} = 1$, both LBC and NLBC formulations produced comparable errors (5–9%), confirming that a geometrically appropriate initial guess mitigated formulation-dependent effects.

Chapter 7 addressed the ultimate challenge: simultaneous determination of both unknown corrosion coefficients $\lambda(\theta)$ and corroded boundary geometry $r(\theta)$. This represents the most realistic scenario for practical corrosion assessment, where both material properties and geometric damage are unknown a priori. The solution approach employed alternating minimization, iterating between coefficient and boundary updates while maintaining stability through combined zeroth and first-order Tikhonov regularization terms. The results confirm that the proposed unified inverse framework, combining the boundary modeling with reduced Fourier coefficient parametrisation, provides a stable, efficient and accurate methodology for corrosion shape and parameter identification. This approach not only enhances numerical robustness but also offers a practical foundation for real-world corrosion assessment, where measurement noise and geometric uncertainties are inevitable. Future work may extend this framework to three-dimensional geometries, time-dependent corrosion processes and coupled multi-physics systems to further improve predictive capabilities and practical applicability.

8.2.1 Numerical methods and optimization

The primary contribution to numerical methods in this thesis was the development of a stable and computationally efficient framework for solving complex, multi-variable optimization problems inherent in non-destructive corrosion assessment. This research successfully addressed the challenge of simultaneously reconstructing both unknown geometric boundary shapes and spatially-varying corrosion coefficients by implementing a sophisticated alternating minimization scheme. To overcome the severe ill-posedness and high dimensionality of this coupled inverse problem, a significant methodological contribution was the introduction of model reduction via trigonometric polynomials parameterization. This approach reduced the number of optimization variables (from 59 to 6 in tested cases), acting as a powerful implicit regularizer that suppressed non-physical oscillations and enhanced computational efficiency without sacrificing reconstruction fidelity.

Additionally, the work contributed significantly to the methodology of data fusion in

numerical optimization, proving that the simultaneous integration of multiple independent Cauchy datasets from different flux configurations is essential for ensuring solution uniqueness and stability under noisy conditions. Finally, the analysis provided a systematic quantification of the role of initialization in non-convex optimization, establishing that the selection of a geometrically appropriate initial guess is often more critical for the success of the inverse solver than the specific boundary condition formulation—linear or nonlinear—employed.

8.3 Future work

The established framework opens several important research avenues with significant potential for advancing both theoretical understanding and practical applications:

Geometric Complexity: Our current method uses a simplified approach called quarter-pipe symmetry to make calculations easier. A key future step is to extend the analysis beyond the quarter-circle case to more realistic geometries exhibiting either left–right symmetry, up–down symmetry or no symmetry at all, as illustrated in Figure 8.2. This extension would better represent practical corrosion scenarios and capture more general boundary behaviours. Although the underlying mathematical equations remain consistent, such an expansion would significantly increase computational demands due to larger system sizes. These challenges could be effectively addressed through the use of advanced iterative solvers and parallel computing techniques.

Three-Dimensional Extension: Real-world corrosion occurs in complex 3D geometries. Extending the boundary reconstruction framework to three-dimensional pipe geometries represents a natural progression. However, computational costs may become prohibitive without advanced fast BEM algorithms. Recent developments in $\mathcal{H}$-matrix (Hierarchical Matrix) techniques and adaptive cross approximation (ACA) methods, see e.g. Bebendorf [12], offer promising solutions to this challenge.

Multiple Defect Detection: Simultaneous identification of multiple corroded regions within a bounded surrounding domain would address practical scenarios where corrosion occurs at multiple locations. This extends beyond simple superposition due to

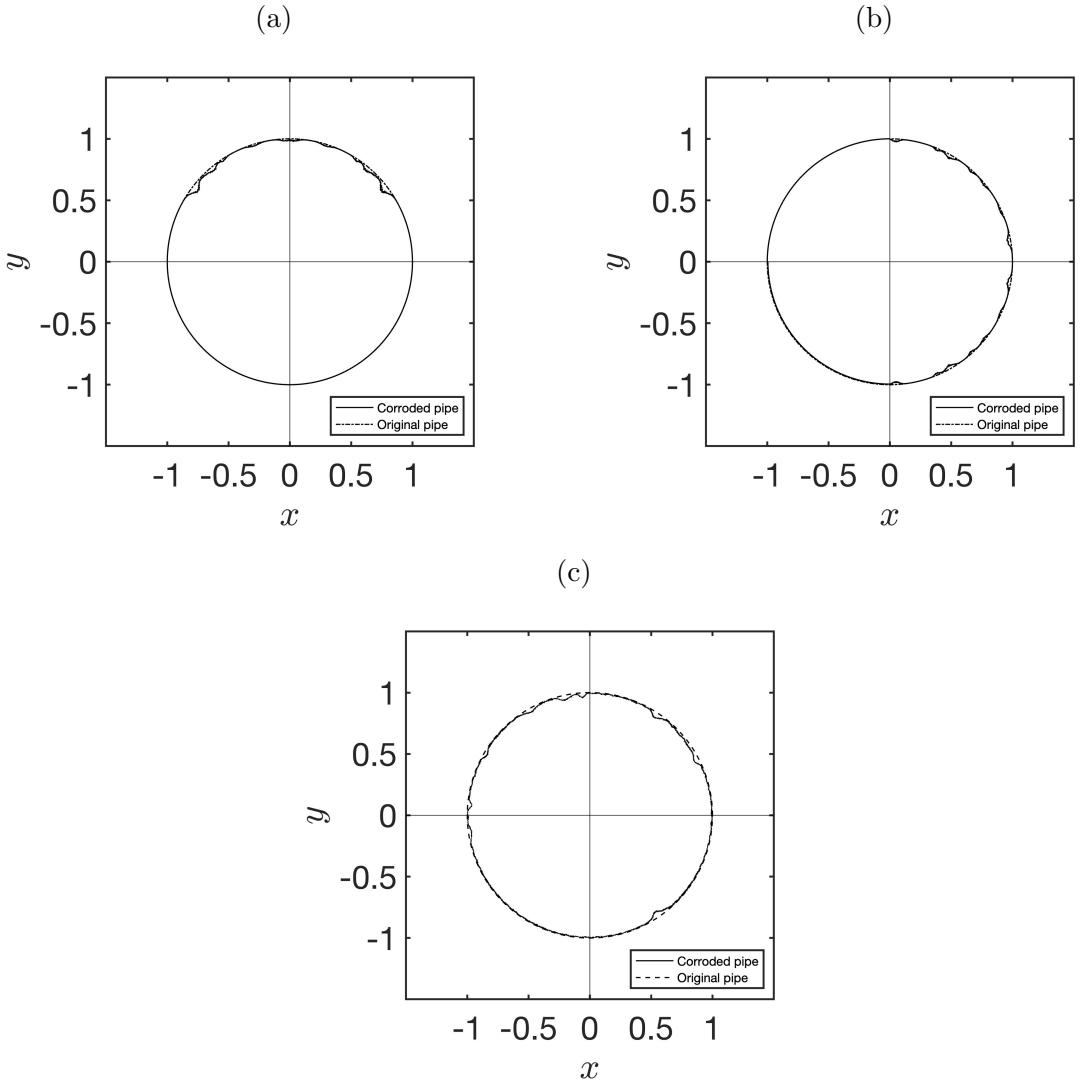

Figure 8.2: Corrosion distribution patterns in corroded pipes: (a) right–left symmetry, (b) up–down symmetry and (c) no symmetry.

electromagnetic field interactions between corroded regions. The mathematical formulation requires significant modification to handle multiple unknown boundaries simultaneously.

Computational Method Comparison: An important avenue for future research involves systematic comparison between BEM and the FEM approach for the Vogelius-Xu corrosion model. While this thesis demonstrates BEM's efficiency advantages in terms of problem setup and computational overhead, comprehensive benchmarking studies comparing solution accuracy, convergence rates and computational time between BEM and the sophisticated multigrid FEM implementation would provide valuable insights for practitioners choosing between numerical methods for specific problem scales and geometries.

Advanced Regularization: Current parameter selection relies on trial-and-error approaches in some chapters. Implementing systematic methods such as the L-curve would enhance robustness and enable automated deployment. For multi regularization parameter choices, the L-hypersurface method [16] provides promising alternatives, though computational costs remain high.

Experimental Validation: The most critical future direction involves validating theoretical predictions against real corrosion experimental data. This requires establishing partnerships with industry to obtain potential measurements from corroded structures, moving beyond synthetic test problems to demonstrate engineering practical utility in field conditions.

Real-Time Implementation: Developing efficient algorithms suitable for real-time corrosion monitoring would enable continuous structural health assessment. This requires further optimization of the iterative algorithms and potentially machine learning approaches to accelerate convergence.

These extensions would establish BEM as a comprehensive tool for non-destructive corrosion evaluation, addressing critical needs in infrastructure maintenance, industrial safety assessment and structural health monitoring. The mathematical framework developed in this thesis provides a solid foundation for these future developments, while the quantitative results demonstrate the practical viability of the approach.

Bibliography

[1] L. Afraites and J. F. T. Rabago. Boundary shape reconstruction with Robin condition: existence result, stability analysis, and inversion via multiple measurements. *Computational and Applied Mathematics*, 43:270, 2024.

[2] L. Afraites and J. F. T. Rabago. Shape optimization methods for detecting an unknown boundary with the Robin condition by a single boundary measurement. *Discrete and Continuous Dynamical Systems*, 18:43–76, 2025.

[3] M. T. Ahmed, J. Lavers, and P. Burke. An evaluation of the direct boundary element method and the method of fundamental solutions. *IEEE Transactions on Magnetics*, 25(4):3001–3006, 1989.

[4] G. Alessandrini, L. Del Piero, and L. Rondi. Stable determination of corrosion by a single electrostatic boundary measurement. *Inverse Problems*, 19:973–984, 2003.

[5] O. M. Alifanov. *Inverse Heat Transfer Problems*. Springer Science & Business Media, 2012.

[6] N. R. Altalhi, D. Lesnic, and S. D. Griffiths. The boundary element method for solving rigid inclusion problems. In *Proceedings of the Thirteenth UK Conference on Boundary Integral Methods (UKBIM13)*, (ed. D. Menshykov), Uni Print, University of Aberdeen, Chapter 5, pp. 41–50, 2023.

[7] N. R. Altalhi, D. Lesnic, and S. D. Griffiths. Solution of corrosion problems using the boundary element method. In *WIT Transactions on Engineering Sciences*, Vol. 136, pp. 15–26, 2024.

[8] N. R. Altalhi, D. Lesnic, and S. D. Griffiths. Determination of a corroded boundary using the boundary elements method. In *Proceedings of the Fourteenth UK Con-*

ference on Boundary Integral Methods (UKBIM14), (ed. E. Chadwick), University of Salford, Chapter 5, pp. 37–48, 2025.

[9] S. Andrieux and A. B. Abda. Identification of planar cracks by complete overdetermined data: inversion formulae. *Inverse Problems*, 12:553–563, 1996.

[10] V. Bacchelli. Uniqueness for the determination of unknown boundary and impedance with the homogeneous Robin condition. *Inverse Problems*, 25(1), Article ID 015004, 2008.

[11] V. Bacchelli. Uniqueness for the determination of unknown boundary and impedance with the homogeneous Robin condition. *Inverse Problems*, 25:Article ID 015004, (4 pages), 2009.

[12] M. Bebendorf. *Hierarchical Matrices*, pages 49–98. Springer, Berlin, 2008.

[13] J. V. Beck and K. J. Arnold. *Parameter Estimation in Engineering and Science*. Publishing house, New York, 1977.

[14] J. V. Beck and K. A. Woodbury. Inverse problems and parameter estimation: integration of measurements and analysis. *Measurement Science and Technology*, 9 (6):839–847, 1998.

[15] F. B. Belgacem. Why is the Cauchy problem severely ill-posed? *Inverse Problems*, 23(2):823–836, 2007.

[16] M. Belge, M. E. Kilmer, and E. L. Miller. Efficient determination of multiple regularization parameters in a generalized L-curve framework. *Inverse Problems*, 18:1161–1183, 2002.

[17] M. Bertero, P. Boccacci, and C. De Mol. *Introduction to Inverse Problems in Imaging*. CRC Press, 2021.

[18] D. E. Beskos. Boundary element methods in dynamic analysis. *Applied Mechanics Reviews*, 30:1–23, 1987.

[19] R. Bialecki, E. Divo, and A. Kassab. Reconstruction of time-dependent boundary heat flux by a BEM-based inverse algorithm. *Engineering Analysis with Boundary Elements*, 30(9):767–773, 2006.

[20] A. Bogomolny. Fundamental solutions method for elliptic boundary value problems.

SIAM Journal on Numerical Analysis, 22(4):644–669, 1985.

[21] D. Borman, D. B. Ingham, B. T. Johansson, and D. Lesnic. The method of fundamental solutions for detection of cavities in EIT. *Journal of Integral Equations and Applications*, 21(3):381–404, 2009.

[22] C. A. Brebbia, J. C. F. Telles, and L. C. Wrobel. *Boundary Element Techniques: Theory and Applications in Engineering*. Springer Science & Business Media, 2012.

[23] G. Burgess and E. Mahajerin. A comparison of the boundary element and superposition methods. *Computers & Structures*, 19(5-6):697–705, 1984.

[24] F. Cakoni and R. Kress. Integral equations for inverse problems in corrosion detection from partial Cauchy data. *Inverse Problems and Imaging*, 1:229–245, 2007.

[25] D. Calvetti, S. Morigi, L. Reichel, and F. Sgallari. Tikhonov regularization and the L-curve for large discrete ill-posed problems. *Journal of Computational and Applied Mathematics*, 123:423–446, 2000.

[26] S. Chaabane and M. Jaoua. Identification of Robin coefficients by the means of boundary measurements. *Inverse Problems*, 15:1425–1438, 1999.

[27] S. Chaabane and M. Jaoua. Recovering Robin's coefficients from boundary measurements. In *Proceedings of 3ICIPE'99, 1999 ASME Design Engineering Technical Conferences*, June 13–18, Port Ludlow, Washington, USA, (11 pages), 1999.

[28] S. Chaabane, C. Elhechmi, and M. Jaoua. A stable recovery method for the Robin inverse problem. *Mathematics and Computers in Simulation*, 66:367–383, 2004.

[29] S. Chaabane, I. Fellah, M. Jaoua, and J. Leblond. Logarithmic stability estimates for a Robin coefficient in two-dimensional Laplace inverse problems. *Inverse Problems*, 20:47–59, 2004.

[30] S. Chaabane, I. Feki, and N. Mars. Numerical reconstruction of a piecewise constant Robin parameter in the two- or three-dimensional case. *Inverse Problems*, 28, Article ID 065016, (19 pp), 2012.

[31] K. Chaji and M. El Bagdouri. Identification of an internal material boundary. *Inverse Problems in Science and Engineering*, 16(4):511–522, 2008.

[32] C. Chen, H. A. Cho, and M. Golberg. Some comments on the ill-conditioning of the

method of fundamental solutions. *Engineering Analysis with Boundary Elements*, 30(5):405–410, 2006.

[33] C. S. Chen, A. Karageorghis, and Y. Li. On choosing the location of the sources in the MFS. *Numerical Algorithms*, 72:107–130, 2015.

[34] C. S. Chen, A. Noorizadegan, D. L. Young, and C.-S. Chen. On the determination of locating the source points of the MFS using effective condition number. *Journal of Computational and Applied Mathematics*, 423, Article ID 114955, 2023.

[35] A. H.-D. Cheng, C.-S. Chen, and A. Karageorghis. *An Introduction to the Method of Fundamental Solutions*. World Scientific Publishing, Singapore, 2025.

[36] J. Cheng, M. Choulli, and J. Lin. Stable determination of a boundary coefficient in an elliptic equation. *Mathematical Models and Methods in Applied Sciences*, 18 (1):107–123, 2008.

[37] M. Choulli. An inverse problem in corrosion detection: Stability estimates. *Journal of Inverse and Ill-Posed Problems*, 12(4):349–367, 2004.

[38] T. F. Coleman and Y. Li. On the convergence of interior-reflective Newton methods for nonlinear minimization subject to bounds. *Mathematical Programming*, 67(1): 189–224, 1994.

[39] T. F. Coleman and Y. Li. An interior trust region approach for nonlinear minimization subject to bounds. *SIAM Journal on Optimization*, 6(2):418–445, 1996.

[40] D. L. Colton and R. Kress. *Inverse Acoustic and Electromagnetic Scattering Theory*. Springer, Berlin, 1998.

[41] J. Deconinck. *Current Distributions and Electrode Shape Changes in Electrochemical Systems*. Springer Science & Business Media, 2012.

[42] E. J. Dickinson and A. J. Wain. The Butler-Volmer equation in electrochemical theory: Origins, value, and practical application. *Journal of Electroanalytical Chemistry*, 872, Article ID 114145, 2020.

[43] A. A. El-Feki. *Mathematical Modelling of Corrosion Measurements*. PhD thesis, University of Wollongong, 1999.

[44] H. W. Engl, M. Hanke, and A. Neubauer. *Regularization of Inverse Problems*.

Springer Science & Business Media, 2000.

[45] M. Essahraoui, E. Cherrat, L. Afraites, and J. F. T. Rabago. Simultaneous recovery of corroded boundaries and admittance using the Kohn–Vogelius method. In *International Conference on Mathematics Data Science*, pages 73–89. Springer, 2024.

[46] M. G. Fontana and N. D. Greene. *Corrosion Engineering*. McGraw-Hill Book Company, 1978.

[47] E. Fontes and B. Nistad. Modeling corrosion and corrosion protection. *Comsol White Paper Series*, 4, 2019.

[48] Q. Fulian and A. Fisher. Computational electrochemistry: the boundary element method. *The Journal of Physical Chemistry B*, 102(48):9647–9652, 1998.

[49] S. V. Gavrilov. Numerical method for solving an inverse problem for Laplace's equation in a domain with an unknown inner boundary. *Computational Mathematics and Mathematical Physics*, 59(1):59–65, 2019.

[50] M. Ghassemi, M. Kamvar, and R. Steinberger-Wilckens. *Fundamentals of Heat and Fluid Flow in High Temperature Fuel Cells*. Academic Press, 2020.

[51] M. Grau-Sánchez, M. Noguera, and J. Gutiérrez. On some computational orders of convergence. *Applied Mathematics Letters*, 23(4):472–478, 2010.

[52] E. Gutierrez-Miravete and M. Turner. Modeling galvanic corrosion. *COMSOL, Boston*, 2:1–4, 2013.

[53] K. Hagiwara. Regularization learning, early stopping and biased estimator. *Neurocomputing*, 48:937–955, 2002.

[54] L. Hanka. Deconvolution of gamma ray spectra using singular value decomposition of matrices. *Land Forces Academy Review*, 2:136–145, 2020.

[55] P. C. Hansen. The truncated SVD as a method for regularization. *BIT Numerical Mathematics*, 27:534–553, 1987.

[56] P. C. Hansen. *Rank Deficient and Discrete Ill-Posed Problems: Numerical Aspects of Linear Inversion*. SIAM, Philadelphia, 1998.

[57] R. A. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge University Press,

Cambridge, England, 1990.

[58] C.-H. Huang and B.-H. Chao. An inverse geometry problem in identifying irregular boundary configurations. *International Journal of Heat and Mass Transfer*, 40(9): 2045–2053, 1997.

[59] T. M. Inc. *Statistics and Machine Learning Toolbox*, 2022. URL `https://www.mathworks.com/help/stats/index.html`.

[60] D. B. Ingham and Y. Yuan. *Boundary Element Method for Solving Improperly Posed Problems*. Computational Mechanics Publications, Southampton, 1994.

[61] G. Inglese. An inverse problem in corrosion detection. *Inverse Problems*, 13:977–994, 1997.

[62] O. Ivanyshyn and R. Kress. Nonlinear integral equations for solving inverse boundary value problems for inclusions and cracks. *Journal of Integral Equations and Applications*, 18:13–38, 2006.

[63] M. Jaoua, S. Chaabane, C. Elhechmi, J. Leblond, M. Mahjoub, and J. R. Partington. On some robust algorithms for the Robin inverse problem. *Revue Arima*, 9: 287–307, 2008.

[64] R. Johnston and G. Fairweather. The method of fundamental solutions for problems in potential flow. *Applied Mathematical Modelling*, 8(4):265–270, 1984.

[65] J. Kaipio and E. Somersalo. Statistical inverse problems: discretization, model reduction and inverse crimes. *Journal of Computational and Applied Mathematics*, 198:493–504, 2007.

[66] A. Karageorghis and D. Lesnic. Reconstruction of an elliptical inclusion in the inverse conductivity problem. *International Journal of Mechanical Sciences*, 142: 603–609, 2018.

[67] A. Karageorghis, D. Lesnic, and L. Marin. An efficient moving pseudo-boundary MFS for void detection. *Engineering Analysis with Boundary Elements*, 147:90–111, 2023.

[68] J. T. Katsikadelis. *Boundary Elements: Theory and Applications*. Elsevier, 2002.

[69] B. I. Kharisov, editor. *Direct Synthesis of Metal Complexes*. Elsevier, 2018.

[70] V. Kolehmainen, M. Lassas, P. Ola, and S. Siltanen. Recovering boundary shape and conductivity in electrical impedance tomography. *Inverse Problems and Imaging*, 7(1):217–242, 2013.

[71] R. Kress. Inverse Dirichlet problem and conformal mapping. *Mathematics and Computers in Simulation*, 66(4-5):255–265, 2004.

[72] K. Kurpisz and A. Nowak. BEM approach to inverse heat conduction problems. *Engineering Analysis with Boundary Elements*, 10(4):291–297, 1992.

[73] D. Landolt. Electrochemical and materials science aspects of alloy deposition. *Electrochimica Acta*, 39(8-9):1075–1090, 1994.

[74] D. Lesnic. A numerical investigation of the inverse potential conductivity problem in a circular inclusion. *Inverse Problems in Engineering*, 9(1):1–17, 2001.

[75] D. Lesnic. *Inverse Problems with Applications in Science and Engineering*. Chapman and Hall/CRC, 2021.

[76] D. Lesnic, J. Berger, and P. Martin. A boundary element investigation of the potential boundary determination. *Third UK Conference on Boundary Integral Methods*, (ed. P. J. Harris), University of Brighton Press, Brighton, pp. 47–56, 2001.

[77] D. Lesnic, L. Elliott, and D. Ingham. The boundary element solution of the Laplace and biharmonic equations subjected to noisy boundary data. *International Journal for Numerical Methods in Engineering*, 43(3):479–492, 1998.

[78] D. Lesnic, L. Elliott, D. Ingham, B. Clennell, and R. Knipe. The identification of the piecewise homogeneous thermal conductivity of conductors subjected to a heat flow test. *International Journal of Heat and Mass Transfer*, 42:143–152, 1999.

[79] D. Lesnic, J. Berger, and P. Martin. A boundary element regularization method for the boundary determination in potential corrosion damage. *Inverse Problems in Engineering*, 10(2):163–182, 2002.

[80] D. Lesnic, T. T. M. Onyango, and D. B. Ingham. The boundary element method for the determination of nonlinear boundary conditions in heat conduction. *WIT Transactions on Modelling and Simulation*, 49:45–55, 2009.

[81] C.-S. Liu, C.-W. Chang, and C.-Y. Chiang. A regularized integral equation method for the inverse geometry heat conduction problem. *International Journal of Heat and Mass Transfer*, 51(21-22):5380–5388, 2008.

[82] J.-C. Liu and T. Wei. Determination of a part of boundary for the Cauchy problem for the Laplace equation. *Inverse Problems in Science and Engineering*, 18(4): 535–548, 2010.

[83] L. Marin, A. Karageorghis, and D. Lesnic. The MFS for numerical boundary identification in two-dimensional harmonic problems. *Engineering Analysis with Boundary Elements*, 35(3):342–354, 2011.

[84] J. Newman and N. P. Balsara. *Electrochemical Systems*. John Wiley & Sons, 2021.

[85] R. Nicholson and M. Niskanen. Joint estimation of Robin coefficient and domain boundary for the Poisson problem. *Inverse Problems*, 38(1):015008, 2021.

[86] C. D. Pagani and D. Pierotti. Identifiability problems of defects with the Robin condition. *Inverse Problems*, 25(5):055007, 2009.

[87] P. Ponthiaux, F. Wenger, and J. Galland. Study of the anodic current-voltage curve of an iron-nickel alloy in normal sulfuric acid. *Journal of the Electrochemical Society*, 142, Article ID 2204, 1995.

[88] H. Rismani-Yazdi, S. M. Carver, A. D. Christy, and O. H. Tuovinen. Cathodic limitations in microbial fuel cells: an overview. *Journal of Power Sources*, 180: 683–694, 2008.

[89] N. Rumigny, P. Papadopoulos, and E. Polak. On the use of consistent approximations in boundary element-based shape optimization in the presence of uncertainty. *Computer Methods in Applied Mechanics and Engineering*, 196:3999–4010, 2007.

[90] E. Sincich. Lipschitz stability for the inverse Robin problem. *Inverse Problems*, 23: 1311–1326, 2007.

[91] Y. Tan. Monitoring the corrosion of the blast furnace by perturbation method. In *Industrial and Applied Mathematics in China*, pages 112–128. World Scientific, 2009.

[92] M. Tanaka and G. S. Dulikravich. *Inverse Problems in Engineering Mechanics*.

Elsevier, 1998.

[93] M. Tanaka and Y. Masuda. Boundary element method applied to some inverse problems. *Engineering Analysis*, 3(3):138–143, 1986.

[94] R. Tibshirani. Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 58(1):267–288, 1996.

[95] B. Ullah and J. Trevelyan. A boundary element and level set based topology optimisation using sensitivity analysis. *Engineering Analysis with Boundary Elements*, 70:80–98, 2016.

[96] M. F. Valle, M. J. Colaço, and F. S. Neto. Estimation of the heat transfer coefficient by means of the method of fundamental solutions. *Inverse Problems in Science and Engineering*, 16(6):777–795, 2008.

[97] M. Vogelius and J.-M. Xu. A nonlinear elliptic boundary value problem related to corrosion modeling. *Quarterly of Applied Mathematics*, 56(3):479–505, 1998.

[98] F. Yaman, V. G. Yakhno, and R. Potthast. A survey on inverse problems for applied sciences. *Mathematical Problems in Engineering*, Article ID 976837, 2013.

[99] F. Yang, L. Yan, and T. Wei. Reconstruction of the corrosion boundary for the Laplace equation by using a boundary collocation method. *Mathematics and Computers in Simulation*, 79(7):2148–2156, 2009.

[100] F. Yang, L. Yan, and T. Wei. Reconstruction of part of a boundary for the Laplace equation by using a regularized method of fundamental solutions. *Inverse Problems in Science and Engineering*, 17(8):1113–1128, 2009.

[101] X. Yang, M. Choulli, and J. Cheng. An iterative BEM for the inverse problem of detecting corrosion in a pipe. *Frontiers and Prospects of Contemporary Applied Mathematics*, 14:1–17, 2005.

[102] N. Zamani, J. Porter, and A. Mufti. A survey of computational efforts in the field of corrosion engineering. *International Journal for Numerical Methods in Engineering*, 23(7):1295–1311, 1986.

[103] H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 67:301–

320, 2005.

Appendix A

Codes and computational routines

This appendix provides supplementary material supporting the numerical methods and computational implementations discussed throughout the thesis. The routines presented here are implemented in MATLAB and constitute the computational framework of the direct and inverse problems studied in Chapters 2– 7.

The codes are organized to reflect their use in different parts of the thesis. In particular, some general-purpose routines are employed across all chapters, while specialized well-defined functions are developed to address the direct and inverse problems equipped with either linear or nonlinear boundary conditions on Γ_0. The following sections document the relevant codes and provide explanations of their mathematical roles within the numerical framework.

A.1 General routines

The general MATLAB routines included here serve as the building blocks of the BEM framework applied throughout this thesis. The subroutine `SetGeo.m` initializes the geometry and boundary discretization, while `DAC.m` provides the inverse cosine function. The routine `RES.m` computes solutions inside the computational domain once the boundary values are obtained. These general functions are repeatedly invoked in later codes for solving both direct and inverse problems.

```
% Set the geometry
```

```
function [X, Y, Z1, Z2] =SetGeo(N)
    M=7*N;
    X=zeros(M+1,1);
    Y=zeros(M+1,1);
    Z1=zeros(M,1);
    Z2=zeros(M,1);

    % Write the coordinates of the endpoints of the boundary elements
    %quarter circle side
    pi=4.0*atan(1.0);
    R=1.0;
    for i=1:N
        theta=(pi/2)*(1-(i-1)/N);
        X(i)=R*cos(theta);
        Y(i)=R*sin(theta);
    end

    XMAX=2.0;
    YMAX=2.0;
    %bottom side
    dx=(XMAX-R)/N;
    for i=N+1:2*N
       X(i) = R + (i-N-1)*dx;
       Y(i) = 0.0;
    end
    %right side
    dy=YMAX/(2*N);
    for i=2*N+1:4*N
       X(i) = XMAX;
       Y(i) = (i-2*N-1)*dy;
    end
    %top side
    dx=XMAX/(2*N);
    for i=4*N+1:6*N
       X(i) = XMAX-(i-4*N-1)*dx;
       Y(i) = YMAX;
    end
    %left side
    dy=(YMAX-R)/N;
    for i=6*N+1:7*N+1
       X(i) = 0.0;
       Y(i) = YMAX-(i-6*N-1)*dy;
    end

    % Write the coordinates of the midpoints of the boundary elements
    for i=1:M
        Z1(i)=(X(i)+X(i+1))/2.0;
        Z2(i)=(Y(i)+Y(i+1))/2.0;
    end
end
```

```
%% Function DAC just calculate the arccos function
function d=DAC(X)

    if (X >= 1.0)
        d=0.0;
    elseif (X <= -1.0)
```

```
        d=pi;
    else
        d=acos(X);
    end

end
```

```
%% Subroutine RES gives the solution in the domain when using BEM
function u0 = RES(M,X,Y,u,du,X0,Y0)
    S=0.0;
    for j=1:M
        H=sqrt((X(j+1)-X(j))^2+(Y(j+1)-Y(j))^2);
        A0=sqrt((X(j)-X0)^2+(Y(j)-Y0)^2);
        B0=sqrt((X(j+1)-X0)^2+(Y(j+1)-Y0)^2);
        CB=(A0^2+H^2-B0^2)/(2.0*A0*H);
        SB=sin(DAC(CB));
        CZ=(A0^2+B0^2-H^2)/(2.0*A0*B0);
        psi=DAC(CZ);
        A=(-A0*CB*log(A0/B0)+H-H*log(B0)-A0*psi*SB)/(2.0*pi);
        dx1=X(j)-X0;
        dy1=Y(j)-Y0;
        dx2=X(j+1)-X0;
        dy2=Y(j+1)-Y0;
        if abs(dx1*dy2-dx2*dy1) < 1e-14
            B=0.0;
        else
            if dx1*dy2-dx2*dy1 > 0
                B=psi/(2.0*pi);
            else
                B=-psi/(2.0*pi);
            end
        end
        S=S+A*du(j)+B*u(j);
    end
    u0=S;
    return;
end
```

A.2 Code for chapter 2

The direct problem under consideration is governed by the Laplace equation (2.1) with prescribed boundary conditions on $\partial\Omega$. To validate the theoretical framework and compare numerical accuracy, two distinct numerical methods are implemented: the BEM and the MFS. Both methods are applied to solve the same boundary value problem, enabling a comprehensive comparison of their computational efficiency, accuracy and convergence properties.

A.2.1 The BEM implementation

The BEM is adopted to reformulate the problem into a system of integral equations over the boundary only, thus reducing the dimensionality by one.

The computational geometry of the boundary $\partial\Omega$ is generated using the routine `SetGeo`, which provides the endpoints (X, Y) and midpoints (Z_1, Z_2) of the boundary elements. The auxiliary routines `DAC` (arccos computation) and `RES` (solution evaluation at interior points) are repeatedly invoked throughout the implementation.

For the linear boundary condition (LBC) case, the solver `laplBEM_LBC_Disc` assembles the matrices A and B corresponding to the single- and double-layer potentials, then imposes the linear relation (2.30). For the nonlinear boundary condition (NLBC) case, the solver `laplBEM_NLBC_Disc` replaces the linear condition with the nonlinear relation (2.33), which is solved iteratively using `fsolve`. Both formulations support either discontinuous flux (Equation (2.27)) or continuous flux (Equation (2.28)) on Γ_2, and allow the parameter λ to be either constant ($\lambda = \pm 1$) or variable ($\lambda(\theta) = \pm\sin(\theta)$).

In the following codes, only the discontinuous (Disc) flux case will be presented. For the continuous flux formulation, the same code structure applies with a simple modification to the flux definition on Γ_2. The discontinuous function:

```
du(i) = 0;
if abs(Z2(i)-1/3) <= epsilon
     du(i) = 1/(2*epsilon);
end
if abs(Z2(i)-2/3) <= epsilon
     du(i) = -1/(2*epsilon);
end
```

is simply replaced by the continuous polynomial:

```
du(i) = Z2(i).*(2 - Z2(i));
```

This change transforms the flux from discontinuous to continuous while keeping all other aspects of the implementation identical.

```
function [u, du] = laplBEM_LBC_Disc(N, X, Y, Z1, Z2, lambda)
    epsilon = 0.1;
    M = 7*N;
    A = zeros(M,M);
    B = zeros(M,M);
    u = zeros(M,1);
    du = zeros(M,1);

    % Set the Boundary conditions
    for i = N+1:2*N
        du(i) = 0;
    end

    for i = 2*N+1:4*N
        du(i) = 0;
        if abs(Z2(i)-1/3) <= epsilon
            du(i) = 1/(2*epsilon);
        end
        if abs(Z2(i)-2/3) <= epsilon
            du(i) = -1/(2*epsilon);
        end
    end

    for i = 4*N+1:6*N
        du(i) = 0;
    end
    for i = 6*N+1:M
        du(i) = 0;
    end

    % Write the matrices A,B
    for i = 1:M
        for j = 1:M
            H = sqrt((X(j+1)-X(j))^2 + (Y(j+1)-Y(j))^2);
            A0 = sqrt((X(j)-Z1(i))^2 + (Y(j)-Z2(i))^2);
            B0 = sqrt((X(j+1)-Z1(i))^2 + (Y(j+1)-Z2(i))^2);
            CB = (A0^2 + H^2 - B0^2)/(2.0*H*A0);
            SB = sin(DAC(CB));
            CZ = (A0^2 + B0^2 - H^2)/(2.0*A0*B0);
            psi = DAC(CZ);
            if (i == j)
                B(i,j) = -0.50;
                A(i,j) = (H - H*log(H/2.0))/(2.0*pi);
            else
                dx1 = X(j)-Z1(i);
                dy1 = Y(j)-Z2(i);
                dx2 = X(j+1)-Z1(i);
                dy2 = Y(j+1)-Z2(i);
                if abs(dx1*dy2-dx2*dy1) < 1e-14
                    B(i,j) = 0.0;
                else
                    if dx1*dy2-dx2*dy1 > 0
                        B(i,j) = psi/(2.0*pi);
                    else
                        B(i,j) = -psi/(2.0*pi);
                    end
                end
                A(i,j) = (-A0*CB*log(A0/B0) + H - H*log(B0) - A0*SB*psi
```

```
    )/(2.0*pi);
            end
        end
    end

    % Solve the LBC problem
    K = B;
    if isscalar(lambda)
        % If lambda is a scalar
        C = A(1:end,1:N) * lambda;
        K(1:end,1:N) = K(1:end,1:N) + C;
    else
        % If lambda is a vector
        C = A(1:end,1:N) * diag(lambda);
        K(1:end,1:N) = K(1:end,1:N) + C;
    end

    f = -A*du;
    u = K\f;

    if isscalar(lambda)
        du(1:N) = lambda * u(1:N);  % Scalar multiplication
    else
        du(1:N) = lambda .* u(1:N);  % Element-wise multiplication
    end
end
```

```
function [u, du] =laplBEM_NLBC_Disc(N, X, Y, Z1, Z2, lambda)
    epsilon=0.1;
    M=7*N;
    A=zeros(M,M);
    B=zeros(M,M);
    u=zeros(M,1);
    du=zeros(M,1);

    % Set the boundary conditions
    for i=N+1:2*N
        du(i)=0;
    end

    for i = 2*N+1:4*N
        du(i) = 0;
        if abs(Z2(i)-1/3) <= epsilon
            du(i) = 1/(2*epsilon);
        end
        if abs(Z2(i)-2/3) <= epsilon
            du(i) = -1/(2*epsilon);
        end
    end

    for i=4*N+1:6*N
        du(i)=0;
    end
    for i=6*N+1:M
        du(i)=0;
    end
```

```
    % Write the matrices A and B
    for i=1:M
        for j=1:M
            H=sqrt((X(j+1)-X(j))^2+(Y(j+1)-Y(j))^2);
            A0=sqrt((X(j)-Z1(i))^2+(Y(j)-Z2(i))^2);
            B0=sqrt((X(j+1)-Z1(i))^2+(Y(j+1)-Z2(i))^2);
            CB=(A0^2+H^2-B0^2)/(2.0*H*A0);
            SB=sin(DAC(CB));
            CZ=(A0^2+B0^2-H^2)/(2.0*A0*B0);
            psi=DAC(CZ);
            if (i==j)
                B(i,j)=-0.50;
                A(i,j)=(H-H*log(H/2.0))/(2.0*pi);
            else
                dx1=X(j)-Z1(i);
                dy1=Y(j)-Z2(i);
                dx2=X(j+1)-Z1(i);
                dy2=Y(j+1)-Z2(i);
                if abs(dx1*dy2-dx2*dy1) < 1e-14
                    B(i,j)=0.0;
                else
                    if dx1*dy2-dx2*dy1 > 0
                        B(i,j)=psi/(2.0*pi);
                    else
                        B(i,j)=-psi/(2.0*pi);
                    end
                end
                A(i,j)=(-A0*CB*log(A0/B0)+H-H*log(B0)-A0*SB*psi)/(2.0*
   pi);
            end
        end
    end

    % solve the NLBC problem
    f = @(u) B*u + A * [2 * lambda .* sinh(u(1:N)/2); du(N+1:M)];
    options = optimoptions('fsolve','Display','off');
    u = fsolve(f,u,options);
    du(1:N) = 2 * lambda .* sinh(u(1:N)/2);
end
```

To verify the numerical results, the following script evaluates the solution at selected test points A, B, and C inside the domain Ω by calling `RES`. Numerical experiments are performed with increasing the number of boundary discretization N, highlighting the convergence properties of the BEM under both linear and nonlinear models, and comparing results with the MFS.

```
clear;
clearvars -global;
addpath('../functions/');

% Test points
A = [1.5, 0.5];
B = [1.5, 1.5];
```

```
C = [0.5, 1.5];
N = [10, 30, 90, 270];  % Array of N values
solA = []; solB = []; solC = [];


for n = N
    m = 7 * n;

    % Initialize geometry and lambda
    theta = (pi/2) * (1 - (1:n)/n)';
    [X, Y, Z1, Z2] = SetGeo(n);
    lambda = sin(theta);  % or lambda={-sin(theta), 1, -1}

    % Compute BEM solution
    [u, du] =laplBEM_LBC_Disc(n, X, Y, Z1, Z2, lambda);

    % Compute solutions at test points
    uA = RES(m, X, Y, u, du, A(1), A(2));
    uB = RES(m, X, Y, u, du, B(1), B(2));
    uC = RES(m, X, Y, u, du, C(1), C(2));

    % Print results
    fprintf('N = %2d       X              Y              U \n', n);
    fprintf('-------------------------------------------------\n');
    fprintf('  A  %14.4f %14.4f %14.4f\n', A(1), A(2), uA);
    fprintf('  B  %14.4f %14.4f %14.4f\n', B(1), B(2), uB);
    fprintf('  C  %14.4f %14.4f %14.4f\n\n\n', C(1), C(2), uC);

    % Store solutions for plotting
    solA = [solA uA];
    solB = [solB uB];
    solC = [solC uC];
end
```

A.2.2 The MFS implementation

The MFS finds the numerical solution using non–singular fundamental solutions with sources placed outside the physical domain, specifically addressing the same geometry and boundary conditions as the BEM formulation for direct comparison. The MFS implementation performs convergence analysis by testing different numbers of source and collocation points. The method follows a modular structure where the primary function `laplaceMFS(n, m)` sets up the MFS discretization including source points, collocation points and boundary conditions. Source points are strategically placed outside the domain at distance $dG = 0.2$ to avoid singularities, while collocation points are positioned where boundary conditions are enforced.

```
function [XS, YS, X, Y, Z1, Z2, CU, lambda] = laplaceMFS(n, m)
```

```
epsilon = 0.1;
N = 7 * n;                  % total number of source points
M = 7 * m;                  % total number of collocation points
dG = 0.2;                   % distance from real boundary to sources

XO = zeros(N+1, 1);
YO = zeros(N+1, 1);
XS = zeros(N, 1);           % sources' points coordinates
YS = zeros(N, 1);           % sources' points coordinates

CU = ones(N, 1);            % unknown coefficients of sources'
functions

X = zeros(M+1, 1);
Y = zeros(M+1, 1);
Z1 = zeros(M, 1);           % collocation points coordinates
Z2 = zeros(M, 1);           % collocation points coordinates
lambda = zeros(M, 1);

% Write the coordinate of endpoints of the elements
%----quarter circle side
R = 1;
for i = 1:m
    theta = pi/2 - ((i-1)/m) * pi/2;
    X(i) = R * cos(theta);
    Y(i) = R * sin(theta);
    lambda(i) = -sin(theta);            %set lambda
end

%----bottom side S1
dx = 1.0 / m;
for i = m+1:2*m
   X(i) = 1.0 + (i-m-1) * dx;
   Y(i) = 0.0;
   lambda(i) = 0;
end
%----right side S2
dy = 1.0 / m;
for i = 2*m+1:4*m
   X(i) = 2.0;
   Y(i) = (i-2*m-1) * dy;
   lambda(i) = 0;
end
%----top side S3
dx = 1.0 / m;
for i = 4*m+1:6*m
   X(i) = 2.0 - (i-4*m-1) * dx;
   Y(i) = 2.0;
   lambda(i) = 0;
end
%----left side S4
dy = 1.0 / m;
for i = 6*m+1:7*m+1
   X(i) = 0.0;
   Y(i) = 2.0 - (i-6*m-1) * dy;
   lambda(i) = 0;
end
```

```
    % Write the coordinate of collocation points as midpoints of each
   elements
    for i = 1:M
        Z1(i) = (X(i) + X(i+1)) / 2.0;
        Z2(i) = (Y(i) + Y(i+1)) / 2.0;
    end

    % Define XS, YS coordinates of source points
    R = 1;
    for i = 1:n
        theta = pi/2 - ((i-1)/n) * pi/2;
        XO(i) = -dG + R * cos(theta);
        YO(i) = -dG + R * sin(theta);
    end
    dx = (1.0 + 2*dG) / n;
    for i = n+1:2*n
       XO(i) = 1.0 - dG + (i-n-1) * dx;
       YO(i) = -dG;
    end
    dy = (1.0 + dG) / n;
    for i = 2*n+1:4*n
       XO(i) = 2.0 + dG;
       YO(i) = -dG + (i-2*n-1) * dy;
    end
    dx = (1.0 + dG) / n;
    for i = 4*n+1:6*n
       XO(i) = 2.0 + dG - (i-4*n-1) * dx;
       YO(i) = 2.0 + dG;
    end
    dy = (1.0 + 2*dG) / n;
    for i = 6*n+1:7*n+1
       XO(i) = -dG;
       YO(i) = 2.0 + dG - (i-6*n-1) * dy;
    end
    for i = 1:N
       XS(i) = (XO(i) + XO(i+1)) / 2.0;
       YS(i) = (YO(i) + YO(i+1)) / 2.0;
    end

    [K, f] = Kf(X, Y, Z1, Z2, XS, YS, m, lambda);
    CU = linsolve(K, f);
end
```

The function `Kf(X, Y, XM, YM, XS, YS, m, lambda)` constructs the coefficient matrix K and right-hand side vector f based on the fundamental solution of Laplace's equation $G(x, y; \xi, \eta) = -\frac{1}{2\pi} \ln(r)$, where $r = \|(x, y) - (\xi, \eta)\|$. The matrix assembly includes standard MFS terms, Robin boundary modifications depending on λ, and the same piecewise boundary conditions implemented in the BEM formulation. The quarter-circle boundary employs the Robin condition with $\lambda(\theta) = -\sin(\theta)$, while the right boundary implements the discontinuous flux condition, as in the BEM implementation.

```
function [K, f] = Kf(X, Y, XM, YM, XS, YS, m, lambda)
    epsilon = 0.1;
    N = length(XM);
    K = zeros(N, N);
    f = zeros(N, 1);
    for i = 2*m+1:4*m
        if abs(YM(i) - 1/3) <= epsilon
            f(i) = 1 / (2 * epsilon);
        end
        if abs(YM(i) - 2/3) <= epsilon
            f(i) = -1 / (2 * epsilon);
        end
    end
    DX = XM - XS';
    DY = YM - YS';
    R2 = DX.^2 + DY.^2;

    dS = sqrt((X(2:N+1) - X(1:N)).^2 + (Y(2:N+1) - Y(1:N)).^2);
    nxv = (Y(2:N+1) - Y(1:N)) ./ dS;
    nyv = (X(1:N) - X(2:N+1)) ./ dS;
    K1 = -(0.5 / pi) * (DX ./ R2);
    K = -(0.5 / pi) * (DX ./ R2 ).* nxv - (0.5 / pi) * (DY ./ R2) .*
   nyv;

    % Adjust K for bottom side S1 (indices 1:m) using the lambda array
    for i = 1:m
        K(i, :) = K(i, :) + lambda(i) * (0.25 / pi )* log(R2(i, :));
    end
end
```

Solution's evaluation functions `RESMFS` and `res1` provide capabilities for computing solution values and the normal derivative. The discretization strategy employs $N = 7n$ total source points distributed across all boundary segments, with $M = 7m$ total collocation points matching the boundary discretization.

```
%% Subroutine RESMFS gives the solution in point (X0,Y0) as sum of
   sources fields
function u0 = RESMFS(X0,Y0,XS,YS,CU)
    DX=X0-XS';
    DY=Y0-YS';
    R2=DX.^2+DY.^2;
    u0=-(0.25/pi)*log(R2)*CU;
end
```

```
%% Subroutine res1 gives the solution at point (X0,Y0)
% and derivative of solution in direction (nx,ny) as sum of sources
   fields
function [u0, dudn] = res1(X0,Y0,XS,YS,CU, nx, ny)
    DX=X0-XS';
    DY=Y0-YS';
    R2=DX.^2+DY.^2;
```

```
    u0=-(0.25/pi)*log(R2)*CU;
    dux = -(0.5 / pi) * (DX ./ R2) * CU;
    duy = -(0.5 / pi) * (DY ./ R2) * CU;
    dudn=dux.*nx+duy.*ny;
end
```

Similar to the BEM approach, the MFS formulation accommodates both discontinuous and continuous flux conditions on Γ_2. All such boundary conditions are specified within the function `fdu(X,Y,XS,YS,CU,m, lambda)`.

```
function f = fdu(X, Y, XS, YS, CU, m, lambda)
    epsilon = 0.1;
    f = zeros(length(X), 1);
    DX = X - XS';
    DY = Y - YS';
    R2 = DX.^2 + DY.^2;
    u = -(0.25 / pi) * log(R2) * CU;
    dux = -(0.5 / pi) * (DX ./ R2) * CU;
    duy = -(0.5 / pi) * (DY ./ R2) * CU;

    f(1:m) = -dux(1:m) .* X(1:m) - duy(1:m) .* Y(1:m) - lambda(1:m) .*
   u(1:m);
    f(m+1:2*m) = -duy(m+1:2*m);
    f(2*m+1:4*m) = dux(2*m+1:4*m);
    for i = 2*m+1:4*m
        if abs(Y(i) - 1/3) <= epsilon
            f(i) = dux(i) - 1 / (2 * epsilon);
        end
        if abs(Y(i) - 2/3) <= epsilon
            f(i) = dux(i) + 1 / (2 * epsilon);
        end
    end
    f(4*m+1:6*m) = duy(4*m+1:6*m);
    f(6*m+1:7*m) = -dux(6*m+1:7*m);
end
```

For the discontinuous (piecewise constant) flux case, the boundary condition on the right boundary is implemented as:

```
f(2*m+1:4*m) = dux(2*m+1:4*m);
for i = 2*m+1:4*m
    if abs(Y(i) - 1/3) <= epsilon
        f(i) = dux(i) - 1 / (2 * epsilon);
    end
    if abs(Y(i) - 2/3) <= epsilon
        f(i) = dux(i) + 1 / (2 * epsilon);
```

```
    end
end
```

For the continuous flux formulation, the piecewise can be replaced by the continuous polynomial boundary condition:

```
f(2m+1:4m) = dux(2m+1:4m) - Y(2m+1:4m).(2-Y(2m+1:4*m));
```

Furthermore, for linear boundary value problems, the MFS system is solved, in `laplaceMFS(n, m)`, directly using MATLAB's optimized `linsolve(K, f)` function given by

```
[K, f] = Kf(X, Y, Z1, Z2, XS, YS, m, lambda);
CU = linsolve(K, f);
```

For nonlinear problems, the linear system solution, in `laplaceMFS(n, m)`, is replaced by an iterative approach using `fsolve`:

```
fm = @(c) fdu(Z1,Z2,XS,YS,c,m, lambda);
options = optimoptions('fsolve','MaxFunctionEvaluations',8e6,...
'MaxIterations',80000,'FunctionTolerance',1e-16, 'Algorithm',...
'levenberg-marquardt','Display','off','StepTolerance',1e-9);
CU = 0.01*ones(N,1);
CU = fsolve(fm,CU,options);
```

Both linear and nonlinear formulations support either discontinuous or continuous flux conditions on Γ_2 through the same boundary condition modification described above.

The MFS implementation generates convergence plots showing solution's accuracy improvement with increased discretization, evaluates solutions at the same test points A, B and C used in the BEM validation, and provides normal derivative verification at boundary points for comprehensive comparison with the BEM results.

```
clear;
addpath('../functions/')
NS=[10 , 30, 90, 270];
A=[1.5, 0.5];
B=[1.5, 1.5];                    % test points
C=[0.5, 1.5];
solA=[]; solB=[]; solC=[];
for n=NS        % n number of source points per boundary segment
    N=7*n;      % N total number of source points
    m=2*n;      % m number of collocation points per boundary segment
    M=7*m;      % M total number of collocation points
```

```
    [XS, YS, X, Y, Z1, Z2, CU,lambda] =laplaceMFS1(n,m);
    uA=RESMFS(A(1),A(2),XS,YS,CU);
    uB=RESMFS(B(1),B(2),XS,YS,CU);
    uC=RESMFS(C(1),C(2),XS,YS,CU);
    fprintf('M = %2d        X               Y               U \n',n);
    fprintf('  A  %14.7f %14.7f %14.4f\n',A(1),A(2),uA);
    fprintf('  B  %14.7f %14.7f %14.4f\n',B(1),B(2),uB);
    fprintf('  C  %14.7f %14.7f %14.4f\n\n\n',C(1),C(2),uC);
    solA= [solA uA];
    solB= [solB uB];
    solC= [solC uC];

    dS=sqrt((X(2:M+1)-X(1:M)).^2 + (Y(2:M+1)-Y(1:M)).^2);
    nxv=(Y(2:M+1)-Y(1:M))./dS;
    nyv=(X(1:M)-X(2:M+1))./dS;
    [uB, dudnB] = res1(Z1,Z2,XS,YS,CU, nxv, nyv);
end
```

A.3 Code for chapter 3

The annular domain Dirichlet problem models electrostatic configurations with a rigid inclusion, where the constant potential on the inner boundary a may be known or unknown. Three examples demonstrate the BEM implementation: the first solves a direct problem with known inner boundary constant a, while the second and third examples determine the unknown constant a through flux conservation constraint. The second example provides analytical verification, whereas the third represents a realistic case without analytical solution available.

The specialized evaluation routine `RESAnnularD` computes solutions at interior points for annular domains, where elements 1 to N represent the outer boundary and elements $N+1$ to $2N$ represent the inner boundary discretization. The function the integrates fundamental solution contributions over both boundaries using the standard BEM integral formulation.

```
%% Subroutine RESAnnularD gives the solution in the domain when using
   BEM
function u0 = RESAnnularD(M,X,Y,u,du,X0,Y0)
    pi=4.0*atan(1.0);
    S=0.0;
    for j=1:M
        j1=j;
        j2=j+1;
        if j>M/2
            j1=j+1;
```

```
            j2=j+2;
        end
        H=sqrt((X(j2)-X(j1))^2+(Y(j2)-Y(j1))^2);
        A0=sqrt((X(j1)-X0)^2+(Y(j1)-Y0)^2);
        B0=sqrt((X(j2)-X0)^2+(Y(j2)-Y0)^2);
        CB=(A0^2+H^2-B0^2)/(2.0*A0*H);
        SB=sin(DAC(CB));
        CZ=(A0^2+B0^2-H^2)/(2.0*A0*B0);
        psi=DAC(CZ);
        A=(-A0*CB*log(A0/B0)+H-H*log(B0)-A0*psi*SB)/(2.0*pi);
        dx1=X(j1)-X0;
        dy1=Y(j1)-Y0;
        dx2=X(j2)-X0;
        dy2=Y(j2)-Y0;
        if abs(dx1*dy2-dx2*dy1) < 1e-14
            B=0.0;
        else
            if dx1*dy2-dx2*dy1 > 0
                B=psi/(2.0*pi);
            else
                B=-psi/(2.0*pi);
            end
        end
        S=S+A*du(j)+B*u(j);
    end
    u0=S;
    return;
end
```

Function `laplaceDirect` implements the direct problem where both inner and outer boundary potentials are prescribed. The outer boundary condition is applied as `u(1:N) = 4.*Z1(1:N) - Z1(1:N)./(Z1(1:N).^2 + Z2(1:N).^2)` while the inner boundary uses `u(N+1:2*N) = 4.*Z1(N+1:2*N) - Z1(N+1:2*N)./(Z1(N+1:2*N).^2 + Z2(N+1:2*N).^2)`. Standard BEM matrices A and B are constructed through double integration loops, then the linear system `du = A\(-B*u)` is solved to obtain the flux distribution. Solution's verification employs the analytical solution $u(r,\theta) = 4r\cos(\theta) - \cos(\theta)/r$ at test points A, B and C through the evaluation routine `RESAnnularD`.

```
close all;
clear;
addpath('../functions/')

N=[10,20,40,80];    % Number of the boundary element

R0=1.0;
R1=0.5;

A=[0.8, 0.0];
B=[0.6, 0.3];                   % test points
C=[-0.3, 0.6];
```

```
solA=[]; solB=[]; solC=[];
N_4plot=[];
err_4plot=[];
U0 = @(r,theta) (4*r - 1./r).*cos(theta);  % exact solution

for n=N
    m=2*n;
    [X, Y, Z1, Z2, u, du] =laplaceDirect(R0,R1,n);
    uA=RESAnnularD(m,X,Y,u,du,A(1),A(2));
    thetaA=atan2(A(2),A(1));
    rA=sqrt(A(1)^2+A(2)^2);
    uA0=(4*rA - 1./rA).*cos(thetaA);
    uB=RESAnnularD(m,X,Y,u,du,B(1),B(2));   % BEM solution in test
   points
    thetaB=atan2(B(2),B(1));
    rB=sqrt(B(1)^2+B(2)^2);
    uB0=(4*rB - 1./rB).*cos(thetaB);
    uC=RESAnnularD(m,X,Y,u,du,C(1),C(2));
    thetaC=atan2(C(2),C(1));
    rC=sqrt(C(1)^2+C(2)^2);
    uC0=(4*rC - 1./rC).*cos(thetaC);

    fprintf('N = %2d     X         Y         U              U0     \n',n);
    fprintf('A %14.7f %14.7f %14.7f %14.7f %14.7f\n',A(1),A(2),uA,uA0);
    fprintf('B %14.7f %14.7f %14.7f %14.7f %14.7f\n',B(1),B(2),uB,uB0);
    fprintf('C %14.7f %14.7f %14.7f %14.7f %14.7f\n\n',C(1),C(2),uC,uC0
   );
    solA= [solA uA];
    solB= [solB uB];
    solC= [solC uC];

    errA = abs(uA - uA0);
    errB = abs(uB - uB0);     % compute the absolute error at points
    errC = abs(uC - uC0);
    N_4plot = [N_4plot n];
    err_4plot = [err_4plot errA];
end

function [X, Y, Z1, Z2, u, du] =laplaceDirect(R0,R1, N)

    M=2*N;
    X=zeros(M+2,1);
    Y=zeros(M+2,1);
    Z1=zeros(M,1);
    Z2=zeros(M,1);
    A=zeros(M,M);
    B=zeros(M,M);
    u=zeros(M,1);
    du=zeros(M,1);


    % Write the coordinate of endpoints of the elements
    %----on outer circle R0
    theta=0.0;
    for i=1:N
        theta=((i-1)/N)*2*pi;
        X(i)=R0*cos(theta);
        Y(i)=R0*sin(theta);
```

```
end
X(N+1)=X(1);
Y(N+1)=Y(1);

%----on inner circle R1
for i=1:N
    theta=-((i-1)/N)*2*pi;
    X(N+1+i)=R1*cos(theta);
    Y(N+1+i)=R1*sin(theta);
end
X(2*N+2)=X(N+2);
Y(2*N+2)=Y(N+2);


% Write the coordinate of midpoints of each elements
for i=1:N
    Z1(i)=(X(i)+X(i+1))/2.0;
    Z2(i)=(Y(i)+Y(i+1))/2.0;
end
for i=N+1:2*N
    Z1(i)=(X(i+1)+X(i+2))/2.0;
    Z2(i)=(Y(i+1)+Y(i+2))/2.0;
end


% Set boundary condition on du/dn
u(1:N)=4.*Z1(1:N)-Z1(1:N)./(Z1(1:N).^2+Z2(1:N).^2);
u(N+1:2*N)=4.*Z1(N+1:2*N)-Z1(N+1:2*N)./(Z1(N+1:2*N).^2+Z2(N+1:2*N)
.^2);

% Write the matrices A,B
for i=1:M
    for j=1:M
        j1=j;
        j2=j+1;
        if j>N
            j1=j+1;
            j2=j+2;
        end
        H=sqrt((X(j2)-X(j1))^2+(Y(j2)-Y(j1))^2);
        A0=sqrt((X(j1)-Z1(i))^2+(Y(j1)-Z2(i))^2);
        B0=sqrt((X(j2)-Z1(i))^2+(Y(j2)-Z2(i))^2);
        CB=(A0^2+H^2-B0^2)/(2.0*H*A0);
        SB=sin(DAC(CB));
        CZ=(A0^2+B0^2-H^2)/(2.0*A0*B0);
        psi=DAC(CZ);
        if (i==j)
            B(i,j)=-0.50;
            A(i,j)=(H-H*log(H/2.0))/(2.0*pi);
        else
            dx1=X(j1)-Z1(i);
            dy1=Y(j1)-Z2(i);
            dx2=X(j2)-Z1(i);
            dy2=Y(j2)-Z2(i);
            if abs(dx1*dy2-dx2*dy1) < 1e-14
                B(i,j)=0.0;
            else
```

```
                    if dx1*dy2-dx2*dy1 > 0
                        B(i,j)=psi/(2.0*pi);
                    else
                        B(i,j)=-psi/(2.0*pi);
                    end
                end
                A(i,j)=(-A0*CB*log(A0/B0)+H-H*log(B0)-A0*SB*psi)/(2.0*
    pi);
            end
        end
    end

    f=-B*u;
    du = A\f;

    % calculate the exact and approximate..
    du0=-2*ones(M,1);
    du0(1:N)=2*(Z1(1:N)./sqrt(Z1(1:N).^2+Z2(1:N).^2));
    r=A*du+B*u;
    r0=A*du0+B*u;
end
```

The second and third examples in function `laplaceInv` address unknown constant potential scenarios, where the BEM system is extended with a constraint equation for flux conservation, extending the coefficient matrix with an additional row and column.

The computational procedure involves three key steps: (i) constructing the constraint vector `v(N+1:end) = 1` to enforce unit coefficients for inner boundary elements while `v(1:N) = 0` for outer boundary elements, (ii) assembling the extended coefficient matrix where `K(1:M,1:M) = A` contains the standard BEM matrix, `K(M+1,1:N) = 1` incorporates the flux conservation constraint $\sum_{j=1}^{N} \frac{\partial u}{\partial n}(\tilde{p}_j) = 0$, and `K(1:M,M+1) = B*v` adds the column for the unknown constant potential, and (iii) solving the extended system `Sol = K\f` where the right-hand side `f(1:M) = -B*u` contains contributions from known outer boundary values and `f(M+1) = 0` represents the flux conservation condition. The solution vector provides both the flux distribution `du = Sol(1:M)` and the unknown constant `a = Sol(M+1)`, which is then assigned to all inner boundary elements via `u(N+1:M) = a`.

The second example employs the outer boundary condition $u(1, \theta) = 3\cos(\theta)$ with analytical validation available, while the third example uses the boundary condition $u(1, \theta) = 10[\exp(-4\sin^2(\theta/2)) - \exp(-4\cos^2(\theta/2))]$ representing a realistic electrostatic configuration without closed-form solution. We perform a convergence analysis across discretizations $N \in \{10, 20, 40, 80\}$ with solution evaluation at interior test points and

condition number monitoring to assess numerical stability. The implementation reports the computed constant potential value and matrix condition number via: `fprintf(' a =%14.5e  cond(A)=%14.7f \ n', a, cond(A))`.

```
close all;
clear;
addpath('../functions/')

N=[10,20,40,80];                % Number of boundary element
A=[0.8, 0.0];
B=[0.6, 0.3];                   % test points
C=[-0.3, 0.6];

solA=[]; solB=[]; solC=[];


R0=1;
R1=0.5;
U0 = @(r,theta) (4*r - 1./r).*cos(theta);        %exact solution
N_4plot=[];
err_4plot=[];

for n=N
    m=2*n;
    [X, Y, Z1, Z2, u, du, a] =laplaceInv(R0,R1,n);
    uA=RESAnnulardD(m,X,Y,u,du,A(1),A(2));
    thetaA=atan2(A(2),A(1));
    rA=sqrt(A(1)^2+A(2)^2);
    uA0=(4*rA - 1./rA).*cos(thetaA);
    uB=RESAnnulardD(m,X,Y,u,du,B(1),B(2));      % BEM solution at test
   points
    thetaB=atan2(B(2),B(1));
    rB=sqrt(B(1)^2+B(2)^2);
    uB0=(4*rB - 1./rB).*cos(thetaB);
    uC=RESAnnulardD(m,X,Y,u,du,C(1),C(2));
    thetaC=atan2(C(2),C(1));
    rC=sqrt(C(1)^2+C(2)^2);
    uC0=(4*rC - 1./rC).*cos(thetaC);


    fprintf('N = %2d      X         Y         U              U0      \n',n);
    fprintf('A %14.7f %14.7f %14.7f %14.7f %14.7f\n',A(1),A(2),uA,uA0);
    fprintf('B %14.7f %14.7f %14.7f %14.7f %14.7f\n',B(1),B(2),uB,uB0);
    fprintf('C %14.7f %14.7f %14.7f %14.7f %14.7f\n\n',C(1),C(2),uC,uC0
   );
    solA= [solA uA];
    solB= [solB uB];
    solC= [solC uC];
    errA = abs(uA - uA0);
    errB = abs(uB - uB0);       % compute the absolute error at points
    errC = abs(uC - uC0);
    N_4plot = [N_4plot m];
    err_4plot = [err_4plot errA];

end
```

```
function [X, Y, Z1, Z2, u, du, a] =laplaceInv(R0,R1, N)

    M=2*N;
    X=zeros(M+2,1);
    Y=zeros(M+2,1);
    Z1=zeros(M,1);
    Z2=zeros(M,1);
    A=zeros(M,M);
    B=zeros(M,M);


    % Write the coordinates of the endpoints of the boundary elements
    %----on outer circle R0
    for i=1:N
        theta=((i-1)/N)*2*pi;
        X(i)=R0*cos(theta);
        Y(i)=R0*sin(theta);
    end
    X(N+1)=X(1);
    Y(N+1)=Y(1);

    %----on inner circle R1
    for i=1:N
        theta=-((i-1)/N)*2*pi;
        X(N+1+i)=R1*cos(theta);
        Y(N+1+i)=R1*sin(theta);
    end
    X(2*N+2)=X(N+2);
    Y(2*N+2)=Y(N+2);


    % Write the coordinates of the midpoints of the boundary elements
    for i=1:N
        Z1(i)=(X(i)+X(i+1))/2.0;
        Z2(i)=(Y(i)+Y(i+1))/2.0;
    end
    for i=N+1:2*N
        Z1(i)=(X(i+1)+X(i+2))/2.0;
        Z2(i)=(Y(i+1)+Y(i+2))/2.0;
    end

    % Write the matrices A,B
    for i=1:M
        for j=1:M
            j1=j;
            j2=j+1;
            if j>N
                j1=j+1;
                j2=j+2;
            end
            H=sqrt((X(j2)-X(j1))^2+(Y(j2)-Y(j1))^2);
            A0=sqrt((X(j1)-Z1(i))^2+(Y(j1)-Z2(i))^2);
            B0=sqrt((X(j2)-Z1(i))^2+(Y(j2)-Z2(i))^2);
            CB=(A0^2+H^2-B0^2)/(2.0*H*A0);
            SB=sin(DAC(CB));
            CZ=(A0^2+B0^2-H^2)/(2.0*A0*B0);
            psi=DAC(CZ);
```

```
            if (i==j)
                B(i,j)=-0.50;
                A(i,j)=(H-H*log(H/2.0))/(2.0*pi);
            else
                dx1=X(j1)-Z1(i);
                dy1=Y(j1)-Z2(i);
                dx2=X(j2)-Z1(i);
                dy2=Y(j2)-Z2(i);
                if abs(dx1*dy2-dx2*dy1) < 1e-14
                    B(i,j)=0.0;
                else
                    if dx1*dy2-dx2*dy1 > 0
                        B(i,j)=psi/(2.0*pi);
                    else
                        B(i,j)=-psi/(2.0*pi);
                    end
                end
                A(i,j)=(-A0*CB*log(A0/B0)+H-H*log(B0)-A0*SB*psi)/(2.0*
   pi);
            end
        end
    end

    u=zeros(M,1);
    u(1:N)=(4.*Z1(1:N))-(Z1(1:N)./(Z1(1:N).^2+Z2(1:N).^2));
    du=zeros(M,1);
    v=zeros(M,1);
    v(N+1:end)=1;
    K=zeros(M+1,M+1);
    K(1:M,1:M)=A;
    K(M+1,1:N)=1;
    K(1:M,M+1)=B*v;
    f=zeros(M+1,1);
    f(1:M)=-B*u;
    Sol = K\f;
    du=Sol(1:M);
    a=Sol(M+1);
    u(N+1:M)=a;

    fprintf(' a =%14.5e       cond(A)=%14.7f \n',a ,cond(A));
end
```

A.4 Code for chapter 4

The inverse Cauchy problem for Laplace's equation (2.1) seeks to reconstruct the missing boundary data on Γ_0 from overspecified Cauchy data $(u, \partial_n u)$ given on the accessible portion Γ_2. Such problems are inherently ill-posed, with small noise in the input leading to large deviations in the solution's output. The general routines `SetGeo`, `DAC`, and `RES` are employed to discretize the geometry, while the direct linear solver `laplBEM_LBC_Disc`

fabricates synthetic numerically simulated data using $N = 90$. The inverse reconstruction is performed with $N = 30$ in order to avoid committing an inverse crime.

The boundary data for the inverse problem is extracted from the high-resolution direct problem solution to which noise is added. To ensure reproducible random number generation, the MATLAB random number generator is reset to its default state using `rng('default')` and then seeded with a specific value using `rng(1)`. The Cauchy data is generated as

```
h = u90((2*90+2):3:(4*90)) + u90((2*90+2):3:(4*90))
    .*(p/100)*(2*rand(2*N,1)-1),
```

where `h` represents the potential values on Γ_2 extracted from the high-resolution direct solution `[u90,du90]`, and p percent perturbations are added through uniformly distributed random variations in the interval $[-1, 1]$. This potential data serves as the additional boundary data for the inverse problem.

A.4.1 Inverse problem routine

We want to solve

$$A \cdot \frac{\partial u}{\partial n} + B \cdot u = f \tag{A.1}$$

where A and B are coefficient matrices from the BEM, u is the potential (unknown on some boundaries) and $\frac{\partial u}{\partial n}$ is the current flux (unknown on some boundaries).

Step 1: Build the coefficient matrix T corresponding to unknown values

$$T = \begin{pmatrix} A_{1,1} & \cdots & A_{1,N} & B_{1,1} & \cdots & B_{1,2N} & B_{1,4N+1} & \cdots & B_{1,M} \\ A_{2,1} & \cdots & A_{2,N} & B_{2,1} & \cdots & B_{2,2N} & B_{2,4N+1} & \cdots & B_{2,M} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ A_{M,1} & \cdots & A_{M,N} & B_{M,1} & \cdots & B_{M,2N} & B_{M,4N+1} & \cdots & B_{M,M} \end{pmatrix}, \tag{A.2}$$

where the matrix blocks correspond to

$$T = \underbrace{\begin{bmatrix} A(:,1:N) \end{bmatrix}}_{\frac{\partial u}{\partial n}(1:N)} \quad \underbrace{B(:,1:2N)}_{u(1:2N)} \quad \underbrace{B(:,4N+1:M)\Big]}_{u(4N+1:M)} \cdot \tag{A.3}$$

This yields the system

$$T \cdot v = f, \tag{A.4}$$

where

$$v = \begin{bmatrix} \frac{\partial u}{\partial n}(1:N) \\ u(1:2N) \\ u(4N+1:M) \end{bmatrix},$$

contains the vector of unknowns.

Step 2: Solve using the Tikhonov regularization method,

$$(T^{\mathrm{T}}T + \mu I) \cdot v = T^{\mathrm{T}} \cdot f, \tag{A.5}$$

where $\mu \geq 0$ is a regularization parameter to be prescribed.

Step 3: Extract the solution from the boundary solution vector v:

$$\begin{cases} \dfrac{\partial u}{\partial n}(1:N) = v(1:N), \\ u(1:2N) = v(N+1:3N), \\ u(4N+1:M) = v(3N+1:M-N). \end{cases} \tag{A.6}$$

The inverse problem is solved using

```
[u, du, u1, du1, u2, du2] = laplInv,
```

which provides three different solution approaches: the linear least-squares returns `(u, du)`, the full SVD approach yields `(u1, du1)` and the TSVD solution produces `(u2, du2)`. Each method offers different advantages in handling the ill-conditioning inherent

in inverse boundary value problems. Artificial random noise of levels p = 1%, 3% and 5% is added to the simulated boundary potential on Γ_2 by (4.19).

```
clear;
clearvars -global;
addpath('../functions/')
global N h X Y Z1 Z2  lambda
global normalized_SD

% solve the direct with N=90
N=90;
M=7*N;
[X, Y, Z1, Z2] =SetGeo(N);

% direct solution
lambda=-1;
[u90, du90] =laplBEM_LBC_Disc(N, X, Y, Z1, Z2, lambda);

% solve the Inverse problem with N=30
N=30;
M=7*N;
[X, Y, Z1, Z2] =SetGeo(N);

% Parameters
p=1;   % percent of noise
ts=156;  %truncation parameter (number of singular values)


% take h(s) on Gamma_1 from direct solution
rng('default');
rng(1);
h=u90((2*90+2):3:(4*90))+u90((2*90+2):3:(4*90)).*(p/100)*(2*rand(2*N,1)
    -1);


function [u, du, u1, du1, u2, du2] =laplInv(N, X, Y, Z1, Z2, h,g,ts)
 global normalized_SD

    M=7*N;
    A=zeros(M,M);
    B=zeros(M,M);
    u=zeros(M,1);
    du=zeros(M,1);
    u1=zeros(M,1);
    du1=zeros(M,1);
    u2=zeros(M,1);
    du2=zeros(M,1);
    mu=0;

    % Set the boundary condition on u and du/dn
    u(2*N+1:4*N)=h;
    epsilon=0.1;
    for i=2*N+1:4*N
        du(i)=0;
        if abs(Z2(i)-1/3)<=epsilon
            du(i)=1/(2*epsilon);
        end
```

```
        if abs(Z2(i)-2/3)<=epsilon
             du(i)=-1/(2*epsilon);
        end
    end

    % Write the matrices A,B
    for i=1:M
        for j=1:M
            H=sqrt((X(j+1)-X(j))^2+(Y(j+1)-Y(j))^2);
            A0=sqrt((X(j)-Z1(i))^2+(Y(j)-Z2(i))^2);
            B0=sqrt((X(j+1)-Z1(i))^2+(Y(j+1)-Z2(i))^2);
            CB=(A0^2+H^2-B0^2)/(2.0*H*A0);
            SB=sin(DAC(CB));
            CZ=(A0^2+B0^2-H^2)/(2.0*A0*B0);
            psi=DAC(CZ);
            if (i==j)
                B(i,j)=-0.50;
                A(i,j)=(H-H*log(H/2.0))/(2.0*pi);
            else
                dx1=X(j)-Z1(i);
                dy1=Y(j)-Z2(i);
                dx2=X(j+1)-Z1(i);
                dy2=Y(j+1)-Z2(i);
                if abs(dx1*dy2-dx2*dy1) < 1e-14
                    B(i,j)=0.0;
                else
                    if dx1*dy2-dx2*dy1 > 0
                        B(i,j)=psi/(2.0*pi);
                    else
                        B(i,j)=-psi/(2.0*pi);
                    end
                end
                A(i,j)=(-A0*CB*log(A0/B0)+H-H*log(B0)-A0*SB*psi)/(2.0*
pi);
            end
        end
    end

    f=-A*du-B*u;
    T=zeros(M,M-N);
    T(:,1:N)=A(:,1:N);
    T(:,N+1:3*N)=B(:,1:2*N);
    T(:,3*N+1:M-N)=B(:,4*N+1:M);
    F=T'*f;
    TT=T'*T+Lambda*eye(6*N);
    v = TT\F;
    u(1:2*N)=v(N+1:3*N);
    u(4*N+1:M)=v(3*N+1:M-N);
    du(1:N) =  v(1:N);

    % Solution by SVD:
    [U,S,V] = svd(T);
    UTf=U'*f;
    SD=diag(S);
    v1=V*(UTf(1:6*N)./SD);
    u1(1:2*N)=v1(N+1:3*N);
    u1(4*N+1:M)=v1(3*N+1:M-N);
    du1(1:N) =  v1(1:N);
```

```
    % Normalize singular values
    SD = diag(S);
    max_SV = max(SD);
    normalized_SD = SD / max_SV;

    % Solution by TSVD:
    [U,S,V] = svds(T,ts);
    UTf=U'*f;
    SD=diag(S);
    v2=V*(UTf(1:ts)./SD);
    u2(1:2*N)=v2(N+1:3*N);
    u2(4*N+1:M)=v2(3*N+1:M-N);
    du2(1:N) = v2(1:N);
end
```

For nonlinear problems, the direct solver `laplBEM_NLBC_Disc` is substituted, while the inverse framework remains unchanged.

A.5 Code for chapter 5

Chapter 5 addressed the inverse problem of identifying the space-dependent corrosion coefficient $\lambda(\theta)$ along the inaccessible but known boundary portion Γ_0. The reconstruction process utilizes Cauchy data, specifically equations (5.3) and (5.4), prescribed on the accessible boundary segment Γ_2 to recover the unknown coefficient distribution.

The problem is solved using specialized boundary element solvers that generate synthetic numerically simulated measurement data for the inverse analysis. The linear boundary condition case is handled by `laplCauchy_BEM_LBC`, which implements the formulation described in (5.7), while the nonlinear boundary condition case is addressed by `laplCauchy_BEM_NLBC`, corresponding to the formulation presented in (5.21).

```
function [u, du] =laplCauchy_BEM_LBC(N, X, Y, Z1, Z2, lambda)
    epsilon=0.1;
    M=7*N;
    A=zeros(M,M);
    B=zeros(M,M);
    u=zeros(M,1);
    du=zeros(M,1);

    % Set the boundary condition
    for i=N+1:2*N
        du(i)=0;
    end
    for i=2*N+1:4*N
        du(i)=0;
```

```
        if abs(Z2(i)-1/3)<=epsilon
             du(i)=1/(2*epsilon);
        end
        if abs(Z2(i)-2/3)<=epsilon
             du(i)=-1/(2*epsilon);
        end
    end
    for i=4*N+1:6*N
        du(i)=0;
    end
    for i=6*N+1:M
        du(i)=0;
    end

    % Write the matrices A and B
    for i=1:M
        for j=1:M
            H=sqrt((X(j+1)-X(j))^2+(Y(j+1)-Y(j))^2);
            A0=sqrt((X(j)-Z1(i))^2+(Y(j)-Z2(i))^2);
            B0=sqrt((X(j+1)-Z1(i))^2+(Y(j+1)-Z2(i))^2);
            CB=(A0^2+H^2-B0^2)/(2.0*H*A0);
            SB=sin(DAC(CB));
            CZ=(A0^2+B0^2-H^2)/(2.0*A0*B0);
            psi=DAC(CZ);
            if (i==j)
                B(i,j)=-0.50;
                A(i,j)=(H-H*log(H/2.0))/(2.0*pi);
            else
                dx1=X(j)-Z1(i);
                dy1=Y(j)-Z2(i);
                dx2=X(j+1)-Z1(i);
                dy2=Y(j+1)-Z2(i);
                if abs(dx1*dy2-dx2*dy1) < 1e-14
                    B(i,j)=0.0;
                else
                    if dx1*dy2-dx2*dy1 > 0
                        B(i,j)=psi/(2.0*pi);
                    else
                        B(i,j)=-psi/(2.0*pi);
                    end
                end
                A(i,j)=(-A0*CB*log(A0/B0)+H-H*log(B0)-A0*SB*psi)/(2.0*
   pi);
            end
        end
    end

    % find the solution of u(1:M)
    K=B;
    C= A(1:end,1:N)*diag(-lambda);
    K(1:end,1:N) = K(1:end,1:N) + C;
    f=-A*du;
    u = K\f;
    du(1:N) = -lambda.* u(1:N);
end
```

```
function [u, du] =laplCauchy_BEM_NLBC(N, X, Y, Z1, Z2, lambda)
```

```
    epsilon=0.1;
    M=7*N;
    A=zeros(M,M);
    B=zeros(M,M);
    u=zeros(M,1);
    du=zeros(M,1);

    % Set the boundary condition
    for i=N+1:2*N
        du(i)=0;
    end
    for i=2*N+1:4*N
        du(i)=0;
        if abs(Z2(i)-1/3)<=epsilon
            du(i)=1/(2*epsilon);
        end
        if abs(Z2(i)-2/3)<=epsilon
            du(i)=-1/(2*epsilon);
        end
    end
    for i=4*N+1:6*N
        du(i)=0;
    end
    for i=6*N+1:M
        du(i)=0;
    end

    % Write the matrices A and B
    for i=1:M
        for j=1:M
            H=sqrt((X(j+1)-X(j))^2+(Y(j+1)-Y(j))^2);
            A0=sqrt((X(j)-Z1(i))^2+(Y(j)-Z2(i))^2);
            B0=sqrt((X(j+1)-Z1(i))^2+(Y(j+1)-Z2(i))^2);
            CB=(A0^2+H^2-B0^2)/(2.0*H*A0);
            SB=sin(DAC(CB));
            CZ=(A0^2+B0^2-H^2)/(2.0*A0*B0);
            psi=DAC(CZ);
            if (i==j)
                B(i,j)=-0.50;
                A(i,j)=(H-H*log(H/2.0))/(2.0*pi);
            else
                dx1=X(j)-Z1(i);
                dy1=Y(j)-Z2(i);
                dx2=X(j+1)-Z1(i);
                dy2=Y(j+1)-Z2(i);
                % if abs(1-abs(dx1*dx2+dy1*dy2)/A0/B0) < 1e-12
                if abs(dx1*dy2-dx2*dy1) < 1e-14
                    B(i,j)=0.0;
                else
                    if dx1*dy2-dx2*dy1 > 0
                        B(i,j)=psi/(2.0*pi);
                    else
                        B(i,j)=-psi/(2.0*pi);
                    end
                end
                A(i,j)=(-A0*CB*log(A0/B0)+H-H*log(B0)-A0*SB*psi)/(2.0*
pi);
            end
```

```
        end
    end

    %solve the Inverse problem
    f = @(u) B*u + A * [-2 * lambda .* sinh(u(1:N)/2); du(N+1:M)];
     options = optimoptions('fsolve','MaxFunctionEvaluations',8e6,'
   MaxIterations',80000,'FunctionTolerance',1e-16, ...
    'Display','off','StepTolerance',1e-11,'OptimalityTolerance',1e-9);
    u = fsolve(f,u,options);
    du(1:N) = -2 * lambda .* sinh(u(1:N)/2);
end
```

The inverse problem is formulated as a constrained nonlinear least-squares optimization problem to recover the corrosion coefficient $\lambda(\theta)$. The reconstruction procedure begins with an initial guess $\lambda^0(\theta)$ (stored in `Lambda_guess`), chosen as a linear function `theta/(pi/2)` of the normalized angular coordinate. To ensure physical consistency, the optimization enforces bounds that restrict the coefficient to remain within the admissible range $0 \leq \lambda \leq 1$ through lower bounds (`Lambda_law`) and upper bounds (`Lambda_up`).

The optimization is performed using MATLAB's `lsqnonlin` function with carefully tuned tolerances to ensure numerical stability and reliable convergence:

```
[Lambda_res, resnorm, residual, exitflag, output] = lsqnonlin(@flsq,
    Lambda_guess, Lambda_law, Lambda_up, options);
```

The function returns several important quantities that characterize the optimization process. The primary output `Lambda_res` contains the reconstructed corrosion coefficient, while `resnorm` provides the norm of the residual at the solution, indicating the quality of the fit. The vector `residual` contains the individual residual components at the final iteration, and `exitflag` reports the termination condition, with positive values indicating successful convergence. The structure `output` provides additional information about the optimization process, including the number of iterations performed and function evaluations.

The objective function `flsq` quantifies the discrepancy between the computed bound-

ary data and the measured Dirichlet data through the composite cost functional:

```
fv = [(2/(2*N))*(u(2*N+1:4*N)-h); mu*(diff(lambda))];
```

This formulation incorporates two essential components: the first term enforces data fidelity by minimizing the mismatch between the computed potential values `u(2*N+1:4*N)` and the measured Dirichlet data `h`, while the second term `mu*(diff(lambda))` introduces first-order Tikhonov regularization to promote smoothness in the reconstructed coefficient. The non–negative regularization parameter `mu` controls the balance between data fitting and solution regularity.

Throughout the optimization process, the algorithm's progress is monitored using a custom output function `outfun` that tracks the evolution of both the coefficient estimates and the residual norm. At each iteration, the current estimate is stored in the array `li`, and the logarithm of the residual norm is recorded through:

```
li(i+1,:)= x;
logLambda_res(i+1) = log10(optimValues.resnorm);
```

This logging mechanism captures `optimValues.resnorm`, which represents the current residual norm provided by the optimizer at each iteration, enabling detailed analysis of the convergence behaviour. Upon successful completion of the optimization, the algorithm stores the final reconstruction `Lambda_res` along with associated convergence metrics and auxiliary data for comprehensive post-processing analysis and validation of the results.

```
clear;
clearvars -global;
addpath('../functions/')

global N h X Y Z1 Z2 u  mu du combined_norm reg_norm
global logLambda_res Lambda_guess

% Parameters
p=0;   % percent of noise
mu=0;  % parameter of regularization

% solve the direct with N=90
N=90;
```

```
M=7*N;
[X, Y, Z1, Z2] =SetGeo(N);
theta=(pi/2)*(1-((1:N)/N)+(1/(2*N)))';
lambda=sin(theta);
[u90, du90] =laplCauchy_BEM_LBC(N, X, Y, Z1, Z2, lambda);

% solve the Inverse problem with N=30
N=30;
M=7*N;
[X, Y, Z1, Z2] =SetGeo(N);
theta=(pi/2)*(1-((1:N)/N)+(1/(2*N)))';
lambda=sin(theta);

% take h(s) on Gamma_1 from direct solution
rng('default');
rng(1);
h=u90((2*90+2):3:(4*90))+u90((2*90+2):3:(4*90)).*(p/100)*(2*rand(2*N,1)
    -1);

% Initial guess for Lambda at midpoints of boundary elements
theta=(pi/2)*(1-((1:N)/N)+(1/(2*N)))';
Lambda_guess =theta/(pi/2);
Lambda_law=0*ones(N,1);
Lambda_up=1*ones(N,1);

options = optimoptions('lsqnonlin','Display','iter','MaxIterations'
    ,10000, ...
'MaxFunctionEvaluations',100000000,'FunctionTolerance',1e-30, ...
'StepTolerance',1e-20,'OptimalityTolerance',1e-15,'OutputFcn',@outfun);
[Lambda_res,resnorm,residual,exitflag,output]=lsqnonlin(@flsq,
    Lambda_guess,Lambda_law,Lambda_up,options);


function [fv]=flsq(lambda)
global N h X Y Z1 Z2 u mu combined_norm reg_norm
[u, ~] =laplCauchy_linear(N, X, Y, Z1, Z2, lambda);
fv=[(2/(2*N))*(u(2*N+1:4*N)-h); mu*(diff(lambda))];
combined_norm=norm((2/(2*N))*(u(2*N+1:4*N)-h));
reg_norm=norm(diff(lambda));
end

function stop = outfun(x,optimValues,state)
stop = false;
global logLambda_res li
i=optimValues.iteration;
if strcmp(state,'iter') && i>=0
    li(i+1,:)=x;
    logLambda_res(i+1)=log10(optimValues.resnorm);
end
end
```

A.6 Code for chapter 6

Building upon the parameter identification techniques of Chapter 5, Chapter 6 addressed the reconstruction of the unknown corroded boundary geometry Γ_0 parameterized as $r(\theta)$ in polar coordinates. The methodology extends beyond coefficient identification on fixed geometries to shape reconstruction problems where the boundary itself becomes part of the unknown solution vector, utilizing a data fusion approach that combines measurements from multiple experiments to enhance reconstruction's accuracy and robustness. This problem framework utilizes two distinct measurement configurations that provide complementary information about the unknown boundary geometry. The first measurement scenario employs the solver `laplLBC_Disc`, which implements the direct identification formulation for discontinuous flux boundary condition, as described in (2.27) (presented as `S1`), while the second configuration utilizes `laplLBC_Cont` for the continuous flux boundary condition according to (2.28) (presented as `S2`).

A crucial distinction from the methodology of Chapter 5 lies in the incorporation of the unknown radius vector `R` as an explicit parameter in both solvers through the functions `laplLBC_Disc(N, R, X, Y, Z1, Z2, lambda)` and `laplLBC_Cont (N, R, X, Y, Z1, Z2, lambda)`. This modification enables the boundary element formulation to account for the varying geometry during the optimization process, where the radius vector components $\mathtt{R}_i(\theta)$ are redefined over the index range $i \in [2 : N-1]$ to maintain consistency with the polar parameterization of the corroded boundary.

```
% Using the BEM to solve the direct-linear problem for the flux(du/dn=g
   =piece-wise)
function [u, du] =laplLBC_Disc(N, R, X, Y, Z1, Z2, lambda)
    epsilon=0.1;
    M=7*N;
    A=zeros(M,M);
    B=zeros(M,M);
    u=zeros(M,1);
    du=zeros(M,1);

    % redefine coordinates of points on quarter circle according to Ri
    for i=2:N
        theta=(pi/2)*(1-(i-1)/N);
        X(i)=R(i-1)*cos(theta);
        Y(i)=R(i-1)*sin(theta);
    end
```

```
    for i=1:N
        Z1(i)=(X(i)+X(i+1))/2.0;
        Z2(i)=(Y(i)+Y(i+1))/2.0;
    end

    % Set boundary condition on du/dn
    for i=N+1:2*N
        du(i)=0;
    end
    for i=2*N+1:4*N
        du(i)=0;
        if abs(Z2(i)-1/3)<=epsilon
            du(i)=1/(2*epsilon);
        end
        if abs(Z2(i)-2/3)<=epsilon
            du(i)=-1/(2*epsilon);
        end
    end
    for i=4*N+1:6*N
        du(i)=0;
    end
    for i=6*N+1:M
        du(i)=0;
    end

    % Write the matrices A and B
    for i=1:M
        for j=1:M
            H=sqrt((X(j+1)-X(j))^2+(Y(j+1)-Y(j))^2);
            A0=sqrt((X(j)-Z1(i))^2+(Y(j)-Z2(i))^2);
            B0=sqrt((X(j+1)-Z1(i))^2+(Y(j+1)-Z2(i))^2);
            CB=(A0^2+H^2-B0^2)/(2.0*H*A0);
            SB=sin(DAC(CB));
            CZ=(A0^2+B0^2-H^2)/(2.0*A0*B0);
            psi=DAC(CZ);
            if (i==j)
                B(i,j)=-0.50;
                A(i,j)=(H-H*log(H/2.0))/(2.0*pi);
            else
                dx1=X(j)-Z1(i);
                dy1=Y(j)-Z2(i);
                dx2=X(j+1)-Z1(i);
                dy2=Y(j+1)-Z2(i);
                if abs(dx1*dy2-dx2*dy1) < 1e-14
                    B(i,j)=0.0;
                else
                    if dx1*dy2-dx2*dy1 > 0
                        B(i,j)=psi/(2.0*pi);
                    else
                        B(i,j)=-psi/(2.0*pi);
                    end
                end
                A(i,j)=(-A0*CB*log(A0/B0)+H-H*log(B0)-A0*SB*psi)/(2.0*
pi);
            end
        end
    end
```

```
    % find the solution of u(1:M)
    K=B;
    C= A(1:end,1:N)*(-lambda);
    K(1:end,1:N) = K(1:end,1:N) + C;
    f=-A*du;
    u = K\f;
    du(1:N) = - lambda.* u(1:N);
end
```

```
% Using the BEM to solve the direct-linear problem for the flux(du/dn=g
   =y(1-y))
function [u, du] =laplLBC_Cont(N,R, X, Y, Z1, Z2, lambda)
    epsilon=0.1;
    M=7*N;
    A=zeros(M,M);
    B=zeros(M,M);
    u=zeros(M,1);
    du=zeros(M,1);

    % redefine coordinates of points on quarter circle according to Ri
    for i=2:N
        theta=(pi/2)*(1-(i-1)/N);
        X(i)=R(i-1)*cos(theta);
        Y(i)=R(i-1)*sin(theta);
    end
    for i=1:N
        Z1(i)=(X(i)+X(i+1))/2.0;
        Z2(i)=(Y(i)+Y(i+1))/2.0;
    end

    % Set the boundary condition on du/dn
    for i=N+1:2*N
        du(i)=0;
    end
    for i=2*N+1:4*N
        du(i)=Z2(i)*(1-Z2(i));
    end
    for i=4*N+1:6*N
        du(i)=0;
    end
    for i=6*N+1:M
        du(i)=0;
    end

    % Write the matrices A and B
    for i=1:M
        for j=1:M
            H=sqrt((X(j+1)-X(j))^2+(Y(j+1)-Y(j))^2);
            A0=sqrt((X(j)-Z1(i))^2+(Y(j)-Z2(i))^2);
            B0=sqrt((X(j+1)-Z1(i))^2+(Y(j+1)-Z2(i))^2);
            CB=(A0^2+H^2-B0^2)/(2.0*H*A0);
            SB=sin(DAC(CB));
            CZ=(A0^2+B0^2-H^2)/(2.0*A0*B0);
            psi=DAC(CZ);
            if (i==j)
                B(i,j)=-0.50;
                A(i,j)=(H-H*log(H/2.0))/(2.0*pi);
```

```
            else
                dx1=X(j)-Z1(i);
                dy1=Y(j)-Z2(i);
                dx2=X(j+1)-Z1(i);
                dy2=Y(j+1)-Z2(i);
                if abs(dx1*dy2-dx2*dy1) < 1e-14
                    B(i,j)=0.0;
                else
                    if dx1*dy2-dx2*dy1 > 0
                        B(i,j)=psi/(2.0*pi);
                    else
                        B(i,j)=-psi/(2.0*pi);
                    end
                end
                A(i,j)=(-A0*CB*log(A0/B0)+H-H*log(B0)-A0*SB*psi)/(2.0*
   pi);
            end
        end
    end

    % find the solution of u(1:M)
    K=B;
    C= A(1:end,1:N)*(-lambda);
    K(1:end,1:N) = K(1:end,1:N) + C;
    f=-A*du;
    u = K\f;
    du(1:N) = - lambda.* u(1:N);
end
```

The synthetic data h_{S1} and h_{S2} represent the potential values extracted from high-resolution direct solutions at discrete points along Γ_2, where $h_{S1} = u^{(1)}|_{\Gamma_2}$ corresponds to the first sensor configuration and $h_{S2} = u^{(2)}|_{\Gamma_2}$ represents the second measurement setup.

Each measurement–specific objective function quantifies the discrepancy between computed and measured boundary data. For instance, the function `flsq_S1`, which evaluates the mismatch for the first measurement, f_{v_S1}, quantifies the discrepancy between computed boundary data from its respective solvers and the corresponding measured values `h_S1` through

```
fv_S1 = (2/(2*N)) * (u(2*N+1:4*N) - h_S1);
```

while `flsq_S2` handles the second measurement with analogous formulation using `h_S2`.

The combined objective function is formulated as `[fv] = combinedObjective(R)`, implementing a data fusion approach that provides flexible utilization of available measurement data. This framework accommodates various measurement scenarios through adjustable parameters α and β. Researchers can employ single measurement configura-

tions by setting the appropriate parameter to unity for the desired setup while setting the unused parameter to zero, or combine both datasets with $\alpha = \beta = 1$ to leverage complementary information from multiple experiment arrangements.

The mathematical formulation takes the form:

$$f_v(\underline{r}) = \begin{bmatrix} \alpha \cdot \frac{2}{2N}(u^{(1)}(2N+1:4N) - h_{S1}) \\ \beta \cdot \frac{2}{2N}(u^{(2)}(2N+1:4N) - h_{S2}) \\ \mu(\underline{r} - \underline{r}^0) \end{bmatrix}. \tag{A.7}$$

This formulation integrates three essential components. The first term $\alpha \cdot f_{v\_S1}$ incorporates data fidelity from the discontinuous flux configuration, the second term $\beta \cdot f_{v\_S2}$ includes information from the continuous flux measurement setup, and the third term $\mu(\underline{r} - \underline{r}^0)$ provides zeroth-order Tikhonov regularization that stabilizes the inversion process by constraining the solution to remain close to an initial estimate $\underline{r}^0$, mitigating the ill-posed nature of the inverse problem.

The optimization utilizes MATLAB's `lsqnonlin` function with stringent tolerances to guarantee numerical stability and convergence:

```
[RR, resnorm, residual, exitflag, output] = lsqnonlin(@combinedObjective,
    R0, R1, R2, options);
```

The primary output `RR` contains the reconstructed boundary shape. The reconstruction process employs the trust-region-reflective algorithm with stringent convergence criteria including `MaxIterations = 1000`, `FunctionTolerance = 10^{-13}`, and `StepTolerance = 10^{-10}`. Upon successful completion of the optimization, the algorithm stores the final reconstruction `RR` along with convergence metrics `logRR`, which captures the optimization trajectory and enables detailed post-processing analysis of the convergence characteristics.

```
% The data fusion of EX1 for LBC with different initial guess
clear;
clearvars -global;
addpath('../functions/')
```

```
global N h_S1 h_S2 X Y Z1 Z2 u R0 mu lambda alpha beta
global logRR Ri  combined_norm  R_norm

%Parameters
p =0;
mu =0;
alpha=1;
beta=1;

% solve the direct solution with N = 90 for S1 and S2
N = 90;
M = 7 * N;
[X, Y, Z1, Z2] = SetGeo(N);
R = ones(N-1, 1);
lambda = 1;

% Calculate the direct solution for S1 and S2
[u90_S1, du90_S1] = laplLBC_Disc(N, R, X, Y, Z1, Z2, lambda);
[u90_S2, du90_S2] = laplLBC_Cont(N, R, X, Y, Z1, Z2, lambda);

% solve the inverse problem with N = 30
N = 30;
M = 7 * N;
[X, Y, Z1, Z2] = SetGeo(N);
R = ones(N-1,1);

% Take h(s) from direct solutions for S1 and S2
rng('default');
rng(1);
h_S1 = u90_S1((2*90+2):3:(4*90));
h_S1 = h_S1 + h_S1 .* ((p / 100) * (2 * rand(2*N,1) - 1));

h_S2 = u90_S2((2*90+2):3:(4*90));
h_S2 = h_S2 + h_S2 .* ((p / 100) * (2 * rand(2*N,1) - 1));

% Initial guess and bounds for R
% R0 = 1.1 * ones(N-1, 1);   % Initial guess R0-1
theta=(pi/2)*(1-((2:N)-1)/N);
R0 = (1 ./ (sqrt(2) * sin(theta + pi/4)))';  % Initial guess R0-2
R1 = 0.5 * ones(N-1, 1);   % Lower bound
R2 = 1.5 * ones(N-1, 1);   % Upper bound


options = optimoptions('lsqnonlin', 'Display', 'iter', 'MaxIterations',
    10000, ...
    'MaxFunctionEvaluations', 1000000, 'FunctionTolerance', 1e-25, ...
    'StepTolerance', 1e-18, 'OptimalityTolerance', 1e-15, 'OutputFcn',
   @outfun);
[RR, resnorm, residual, exitflag, output] = lsqnonlin(
   @combinedObjective, R0, R1, R2, options);

% Combined objective functions S = S1 + S2
function [fv] = combinedObjective(R)
    global R0 mu combined_norm  R_norm  alpha beta
    fv_S1 = flsq_S1(R);
    fv_S2 = flsq_S2(R);
    fv = [alpha*fv_S1; beta*fv_S2; mu * (R-R0)];
    combined_norm=norm([alpha*fv_S1; beta*fv_S2]);
```

```
    R_norm=norm(R-R0);
end

% Objective function for S1
function [fv_S1] = flsq_S1(R)
    global N h_S1 X Y Z1 Z2 u lambda
    [u, ~] = laplLBC_Disc(N, R, X, Y, Z1, Z2, lambda);
    fv_S1 = (2/(2*N)) * (u(2*N+1:4*N) - h_S1);
end

% Objective function for S2
function [fv_S2] = flsq_S2(R)
    global N h_S2 X Y Z1 Z2 u lambda
    [u, ~] = laplLBC_Cont(N, R, X, Y, Z1, Z2, lambda);
    fv_S2 = (2/(2*N)) * (u(2*N+1:4*N) - h_S2);
end

function stop = outfun(x, optimValues, state)
    stop = false;
    global logRR Ri
    i = optimValues.iteration;
    if strcmp(state, 'iter') && i >= 0
        Ri(i+1, :) = x;
        logRR(i+1) = log10(optimValues.resnorm);
    end
end
```

A.7 Code for chapter 7

In Chapter 7, we addressed the simultaneous reconstruction of both the boundary geometry $r(\theta)$ and the boundary condition $\lambda(\theta)$ representing a more challenging inverse problem than those from the previous Chapters 5 and 6. The direct problem solutions are computed using `laplLBC_ID_Simu` for discontinuous flux and `laplLBC_Cnvx_Simu` for continuous flux configurations.

The implementation first generates the measurement data by solving direct problems with $N = 90$ for both flux scenarios, producing h_{S1} and h_{S2} on Γ_2 with noise added. For the inverse problem, $N = 30$ is used with the combined parameter vector

$$\underline{x} = \begin{bmatrix} \underline{r} \\ \\ \underline{\lambda} \end{bmatrix}$$

containing $\overline{1, N-1}$ radius values and $\overline{N, 2N-1}$ boundary corrosion coefficients. Initial

guesses are set as $r^0 = 1.1$ and $\lambda^0 = \frac{\theta}{\pi/2}$, combined into the initial vector `x0 = [r_0; lambda_0]`. Box constraints are defined separately for each parameter type with bounds $[0.5, 1.5]$ for r and $[0, 1]$ for λ, which are then combined into the vectors `lb = [R_law; lambda_law]` and `ub = [R_up; lambda_up]`.

The objective function extends the formulations from both preceding chapters by combining residuals from dual measurements with regularization terms for both unknowns. The structure merges Chapter 6's geometric reconstruction functional with Chapter 5's coefficient recovery approach, where α and β weight the contributions from S_1 and S_2 (data fusion from Chapter 6), while μ_r penalizes radius deviations (Chapter 6) and $\mu_\lambda = \mu_r$ enforces smoothness in $\lambda(\theta)$ through a discrete first–order derivative (Chapter 5).

The regularization term $\mu_\lambda \,\mathrm{diff}(\lambda)$ computes successive differences

$$\mathrm{diff}(\lambda) = [\lambda_2 - \lambda_1,\ \lambda_3 - \lambda_2,\ \ldots,\ \lambda_N - \lambda_{N-1}] \approx \frac{d\lambda}{d\theta},$$

implementing the smoothness penalty $\mu_\lambda \sum_{j=1}^{N-1} (\lambda_{j+1} - \lambda_j)^2$.

At each iteration, `lsqnonlin` calls both direct solvers to compute solutions for current estimates of both r and λ, comparing them with measurement data using the normalized residual formulation established in Chapter 6, thereby solving a higher-dimensional inverse problem that encompasses both geometric and physical parameter identification.

The `output` function tracks convergence by logging both parameter sets throughout the optimization, while post-processing extracts r_{res} and λ_{res} from the solution vector and computes the reconstruction errors $\|r - r_{\mathrm{res}}\|_2$ and $\|\lambda - \lambda_{\mathrm{res}}\|_2$.

```
clear;
clearvars -global;
addpath('../functions/')
global N X Y Z1 Z2 u  mu du  alpha beta h_S1 h_S2  mu_r mu_L R
global logobj_res li Ri lambda_0 R_0 x0  combined_norm reg_norm

% Parameters
p=0;
mu_r=0;
mu_L=0;
alpha=1;
beta=1;
```

```
% solve the direct with N=90
N=90;
M=7*N;
[X, Y, Z1, Z2] =SetGeo(N);
theta=(pi/2)*(1-((1:N)/N)+(1/(2*N)))';
R=ones(N-1,1);
lambda=sin(theta);

% direct solution
[u90_S1, du90_S1] = laplLBC_ID_Simu(N, R, X, Y, Z1, Z2, lambda);
[u90_S2, du90_S2] = laplLBC_Cnvx_Simu(N, R, X, Y, Z1, Z2, lambda);

% solve the Inverse problem with N=30
N=30;
M=7*N;
[X, Y, Z1, Z2] =SetGeo(N);
R=ones(N-1,1);
theta = (pi/2)*(1-(1:N)/N+(1/(2*N)))';
lambda=sin(theta);

% take h(s) on Gamma_1 from direct solution
rng('default');
rng(1);
h_S1 = u90_S1((2*90+2):3:(4*90));
h_S1 = h_S1 + h_S1 .* (p / 100) * (2 * rand(2*N,1) - 1);

h_S2 = u90_S2((2*90+2):3:(4*90));
h_S2 = h_S2 + h_S2 .* (p / 100) * (2 * rand(2*N,1) - 1);

% Initial guesses
R_0 = 1.1*ones(N-1,1);
lambda_0 = theta/(pi/2);
x0 = [R_0;lambda_0];        % Combined vectors

% lawer and upper Bounds for lambda and R
R_law = 0.5*ones(N-1,1);
R_up = 1.5*ones(N-1,1);
lambda_law = 0*ones(N,1);
lambda_up = 1*ones(N,1);

% Combined vectors of lawer and upper Bounds
lb = [R_law;lambda_law];
ub = [ R_up;lambda_up];

options = optimoptions('lsqnonlin','Display', 'iter','MaxIterations',
    10000, ...
     'MaxFunctionEvaluations', 1000000, 'FunctionTolerance', 1e-20, ...
     'StepTolerance', 1e-20,'OptimalityTolerance', 1e-15, 'OutputFcn',
    @outfun);
[x_res, resnorm, residual, exitflag, output] = lsqnonlin(
    @combinedObjective, x0, lb, ub, options);

% To Extract results
R_res = x_res(1:N-1);
lambda_res = x_res(N:end);

% save data
filename=['./data/inverseRecons_LBC_alpha',num2str(alpha),'_beta',
```

```
    num2str(beta),'_p',num2str(p),'_mu_r',num2str(mu_r),'_mu_L',num2str(
    mu_L),'.mat'];
save(filename,'x_res','logobj_res','p','mu','mu_r','mu_L','N','X','Y','
    R','lambda','h_S1','h_S2','u','du','combined_norm','reg_norm')

% Combined objective functions S = S1 + S2
function [fv] = combinedObjective(x)
    global combined_norm reg_norm alpha beta mu_r mu_L R_0 R lambda
    lambda_0 mu x_0
    fv_S1 = flsq_S1(x);
    fv_S2 = flsq_S2(x);
    fv = [alpha*fv_S1; beta*fv_S2; [mu_r * (R-R_0); mu_L * diff(lambda)
    ]];
    combined_norm=norm([alpha*fv_S1; beta*fv_S2]);
    reg_norm=norm([mu_r * (R-R_0); mu_L * diff(lambda)]);
end

% Objective function for S1 (Discontinuous flux)
function [fv_S1] = flsq_S1(x)
    global N h_S1 X Y Z1 Z2 u lambda  du
    R = x(1:N-1);
    lambda = x(N:end);
    [u,du] = laplLBC_ID_Simu(N, R, X, Y, Z1, Z2, lambda);
    fv_S1 = (2/(2*N)) * (u(2*N+1:4*N) - h_S1);
end

% Objective function for S2 (Continuous flux)
function [fv_S2] = flsq_S2(x)
    global N h_S2 X Y Z1 Z2 u lambda  du
    R = x(1:N-1);
    lambda = x(N:end);
    [u, du] = laplLBC_Cnvx_Simu(N, R, X, Y, Z1, Z2, lambda);
    fv_S2 = (2/(2*N)) * (u(2*N+1:4*N) - h_S2);
end

% Output function for logging
function stop = outfun(x, optimValues, state)
    stop = false;
    global logobj_res li Ri N
    i = optimValues.iteration;
    if  strcmp(state, 'iter') && i >= 0
        Ri(i+1, :) = x(1:N-1);
        li(i+1, :) = x(N:end);
        logobj_res(i+1) = log10(optimValues.resnorm);
    end
end
```

For the nonlinear boundary condition case, the same inverse problem framework applies with the direct solvers `laplLBC_ID_Simu` and `laplLBC_Cnvx_Simu` replaced by their nonlinear counterparts `laplNLBC_ID_Simu` and `laplNLBC_Cnvx_Simu`, which handle the additional complexity introduced by the nonlinear terms in the boundary conditions while maintaining the same optimization structure and regularization strategy.