TÉCNICO LISBOA

Accuracy Improvement Strategies for the Immersed Layers Method with Finite Volume Discretisation

Miguel Alexandre Santos Sousa

Thesis to obtain the Master of Science Degree in

Aerospace Engineering

Supervisor: Doctor Duarte Manuel Salvador Freire Silva de Albuquerque

Examination Committee

Chairperson: Professor Filipe Szolnoky Ramos Pinto Cunha
Supervisor: Doctor Duarte Manuel Salvador Freire Silva de Albuquerque
Member of the Committee: Professor José Manuel da Silva Chaves Ribeiro Pereira

December 2022

Declaration

I declare that this document is an original work of my own authorship and that it fulfils all the requirements of the Code of Conduct and Good Practices of the Universidade de Lisboa.

Dedicated to my family.

Acknowledgments

First of all, I would like to thank my family for always inciting me to follow my dreams, motivating me to study and supporting my decisions. The person I am today is the result of the education I had and the values that were transmitted to me by those who were always there.

To my friends, both to those who have been with me since childhood and to those I met during university, a big thanks for the friendship and for all the good times lived together.

To all those who were part of my school and academic path, colleagues and professors, my thanks for accompanying me on this journey and for all the shared knowledge.

To my supervisor, Dr. Duarte Albuquerque, my thanks for the exciting and challenging proposed dissertation topics and my deep gratitude for the all the guidance provided throughout the last months, the constant availability to discuss results, give explanations, suggest new ideas and for the scientific review of the Thesis document.

To *Fundação para a Ciência e Tecnologia* (FCT), my thanks for the financial support given to scientific research, in particular to the project in which this research work is inserted: High-order Immersed Boundary method for Moving Body Problems (HIBforMBP), identified as PTDC/EME-EME/32315/2017.

Resumo

O Método de Filme Imerso desenvolvido por Jeff D. Eldredge constitui um método de fronteira imersa de primeira ordem, que permite a imposição de diferentes valores de condição de fronteira de cada lado da interface, viabilizando a resolução de problemas para os quais métodos anteriores originam resultados incorretos. Neste trabalho, diversas estratégias unidimensionais e bidimensionais de melhoria de precisão são deduzidas para o Método de Filme Imerso, considerando campos não suaves através da interface imersa, aplicado à resolução da equação de Poisson numa formulação de volume finito. O procedimento de melhoria consiste em substituir a segunda equação original, baseada numa operação de interpolação com recurso a funções delta discretas, por uma equação alternativa.

No caso unidimensional, são deduzidas estratégias que realizam uma interpolação polinomial dos subcampos para a interface e estratégias que expressam a descontinuidade na derivada normal, ambas utilizando métodos diretos e ambas conduzindo a segunda ordem de precisão na região supérflua. No caso bidimensional, são deduzidas estratégias assentes em métodos diretos e de mínimos quadrados, que expressam a descontinuidade na derivada normal através da interface. Em algumas implementações bidimensionais, foram obtidos decaimentos supérfluos muito próximos da segunda ordem.

Numa segunda fase, os valores do campo numérico na região de transição são substituídos por estimativas calculadas a partir da região supérflua e, nalgumas versões, usando também informação armazenada nos pontos sólidos. Este passo final permite recuperar o aspeto descontínuo da interface e propagar os decaimentos supérfluos para todo o domínio computacional.

Palavras-chave: Método de Fronteira Imersa, Método de Filme Imerso, Função Delta Discreta, Método de Volume Finito, Dinâmica de Fluidos Computacional.

Abstract

The Immersed Layers Method developed by Jeff D. Eldredge constitutes a first-order immersed boundary method that allows the imposition of different boundary condition values on each side of the interface, enabling the solving of problems for which previous methods originate incorrect results. In this work, one derives several one-dimensional and two-dimensional accuracy improvement strategies to the Immersed Layers Method, considering non-smooth fields across the interface, when solving the Poisson equation in a finite volume framework. The improvement procedure consists of replacing the original second equation, relying on interpolation with discrete delta functions, by an alternative one.

In the one-dimensional case, one derives strategies that perform a polynomial interpolation of the subfields to the interface and strategies expressing the normal derivative jump, with both relying on direct methods and leading to second-order accuracy in the superfluous region. In the two-dimensional case, one derives strategies relying on direct and least squares interpolation methods, to express the normal derivative jump across the interface. For some two-dimensional implementations, superfluous decays very close to second-order were obtained.

In a second stage, the numerical field values in the transition region are replaced by estimates computed from the superfluous field and, in some versions, using the information stored in the solid points as well. This final step allows the recovering of interface sharpness and the propagation of the superfluous decays to the entire domain.

Keywords: Immersed Boundary Method, Immersed Layers Method, Discrete Delta Function, Finite Volume Method, Computational Fluid Dynamics.

Contents

Declaration . . . iii
Acknowledgments . . . vii
Resumo . . . ix
Abstract . . . xi
List of Tables . . . xvii
List of Figures . . . xix
Acronyms . . . xxi
Nomenclature . . . xxiii

1 Introduction **1**
1.1 Motivation . . . 1
1.2 Background . . . 2
1.3 Objectives . . . 4
1.4 Present Contributions . . . 5
1.5 Thesis Outline . . . 5

2 Immersed Layers Method **7**
2.1 Problem Description . . . 7
2.2 Indicator Function . . . 8
2.3 Generalised Functions . . . 9
2.4 Concept of Masked Variable . . . 10
2.5 Poisson Equation of a Masked Variable . . . 11
2.6 Implementation Strategy of the Masked Poisson Equation . . . 12
2.6.1 Closing the System of Equations . . . 12
2.6.2 Domain Discretisation . . . 12
2.6.3 Treatment of the System of Equations . . . 13
2.6.4 Operators Discretisation . . . 14

3 1D Improvement Strategies **21**
3.1 Problem Description . . . 21
3.2 Equation for the Normal Derivative Jump . . . 22
3.2.1 Using One Superfluous Point from Each Subdomain . . . 23

3.2.2 Using Two Superfluous Points from Each Subdomain . . . 24
3.3 Equation Imposing the Interface Value . . . 24
3.3.1 Using Two Superfluous Points from Each Subdomain . . . 25
3.3.2 Using Three Superfluous Points from Each Subdomain . . . 26
3.4 Smearing Correction . . . 26
3.4.1 Using One Superfluous Point from Each Subdomain and the Solid Point . . . 27
3.4.2 Using Two Superfluous Points from Each Subdomain . . . 28
3.4.3 Using Two Superfluous Points from Each Subdomain and the Solid Point . . . 29

4 1D Results **31**
4.1 General Considerations . . . 31
4.2 Original Formulation . . . 34
4.3 Equation for the Normal Derivative Jump . . . 38
4.4 Equation Imposing the Interface Value . . . 40
4.5 Comparison between 1D Methods . . . 42
4.6 Smearing Correction . . . 43

5 2D Improvement Strategies **47**
5.1 Problem Description . . . 47
5.2 Direct Method . . . 49
5.3 Least Squares Method . . . 51
5.3.1 Strategy with No Polynomial Imposition of the Solid Point Value . . . 52
5.3.2 Strategy with Polynomial Imposition of the Solid Point Value . . . 54
5.3.3 Selection of the Sample Points . . . 55
5.4 1D-Inspired Method . . . 56
5.5 Smearing Correction . . . 59
5.5.1 Correction with a Direct Method . . . 60
5.5.2 Correction along Each Cartesian Direction . . . 61

6 2D Results **63**
6.1 General Considerations . . . 63
6.2 Original Formulation . . . 65
6.3 Direct Method . . . 67
6.4 Least Squares Method . . . 69
6.4.1 Using Three Polynomial Terms . . . 70
6.4.2 Using Six Polynomial Terms . . . 72
6.5 1D-Inspired Method . . . 74
6.6 Comparison between 2D Methods . . . 75
6.7 Smearing Correction . . . 76

7 Conclusions **79**
7.1 Achievements . . . 79
7.2 Future Work . . . 80

Bibliography **81**

A Generalised Functions **83**
A.1 Heaviside Function . . . 83
A.2 Dirac Delta Function . . . 83
A.2.1 Unitary Integral . . . 83
A.2.2 Immersion and Restriction Operations . . . 84
A.3 Discrete Delta Function . . . 85

B Calculus **89**
B.1 Gauss Divergence Theorem . . . 89
B.2 Face Directional Derivative in Non-Interior Cells . . . 89

C 1D Results - Additional Data **91**
C.1 Comparison between Strategies . . . 91
C.2 Comparison between Corrective Procedures . . . 93

List of Tables

4.1 Expression of each analytical subfield considered in functions F1, F2, F3, F4. 32
4.2 Error decay for function F1, in the original IL formulation, using finite differences ($\mu = 0$). . 34
4.3 Error decay for function F2, in the original IL formulation, using finite differences ($\mu = 0$). . 35
4.4 Error decay for function F3, in the original IL formulation, using finite differences ($\mu = 0$). . 36
4.5 Error decay for function F4, in the original IL formulation, using finite differences ($\mu = 0$). . 37
4.6 Error decay for function F1, in the original IL formulation, using finite volume ($\mu = 1$). . . . 37
4.7 Error decay for function F2, in the original IL formulation, using finite volume ($\mu = 1$). . . . 38
4.8 Error decay for function F3, in the original IL formulation, using finite volume ($\mu = 1$). . . . 38
4.9 Error decay for function F4, in the original IL formulation, using finite volume ($\mu = 1$). . . . 38
4.10 Error decay for function F4, using the σ-equation with one superfluous point ($\mu = 1$). . . . 38
4.11 Error decay for function F4, using the σ-equation with two superfluous points ($\mu = 1$). . . . 39
4.12 Error decay for function F4, with $\mu = 1$, using the $\bar{\phi}$-equation with two superfluous points. 41
4.13 Error decay for function F4, with $\mu = 1$, using the $\bar{\phi}$-equation with three superfluous points. 41
4.14 Error decay for function F4, with $\mu = 1$, using strategy σ-equation (2), with correction C1S. 43
4.15 Error decay for function F4, with $\mu = 1$, using strategy σ-equation (2), with correction C2. . 43
4.16 Error decay for function F4, with $\mu = 1$, using strategy σ-equation (2), with correction C2S. 44

5.1 Terms to achieve an order of accuracy ρ in the first derivative, with a direct method. 50
5.2 Terms to achieve an accuracy ρ in the first derivative with least squares, adapted from [16]. 51
5.3 Comparison between the Direct Method and the Least Squares Method. 52

6.1 Error decay for function F5, in the original IL formulation, using finite differences. 65
6.2 Error decay for function F5, in the original IL formulation, using finite volume. 66
6.3 Decays for function F5, with the Direct Method for 3×2 and 3×3 blocks. 67
6.4 Superfluous decays using method LSM1 with three terms, for $s_\eta = Zh$, $s_\xi = (\frac{w}{2} + Z)h$. . . 70
6.5 Superfluous decays using method LSM3 with three terms, for $s_\eta = Zh$, $s_\xi = (\frac{w}{2} + Z)h$. . . 71
6.6 Superfluous decays using method LSM1 with six terms, for $s_\eta = Zh$, $s_\xi = (\frac{w}{2} + Z)h$. . . . 72
6.7 Superfluous decays using method LSM3 with six terms, for $s_\eta = Zh$, $s_\xi = (\frac{w}{2} + Z)h$. . . . 73
6.8 Superfluous decays using the 1D-Inspired Method, considering $Z \times Z$ superfluous blocks. 74
6.9 Superfluous comparison between the original IL formulation and the best methods, with δ_3^*. 76
6.10 Comparison between the two-dimensional correction strategies for method LSM-Y with δ_3^*. 77

List of Figures

2.1 Illustration of a general two-dimensional problem ($\lambda = 2$). 8
2.2 Finite differences and finite volume approaches used to discretise the computational domain. 13
2.3 Representation of discrete delta function δ_3^*, in one and two dimensions, taking $h = 1$. . . 16

3.1 One-dimensional problem in hands, approached by a finite volume reasoning. 21
3.2 Illustration of the smearing phenomenon near the interface. The expected behaviour appears in yellow and the smeared one in orange. The affected region appears in shades of grey ($w = 2$). 22
3.3 Illustration of the three one-dimensional correction strategies implemented ($w = 2$). 27
3.4 General first half of the transition region, with k points needing correction. 28

4.1 Graphical representation of all the four one-dimensional functions considered. 32
4.2 Results for function F1 under a finite differences approach, using δ_3^* and $n = 100$ cells. . . 34
4.3 Results for function F2 under a finite differences approach, using δ_3^* and $n = 100$ cells. . . 35
4.4 Results for function F3 under a finite differences approach, using δ_3^* and $n = 100$ cells. . . 35
4.5 Transition region around the first solid point, $|x - x_{sp_1}| \leq (\frac{w}{2})h$, using δ_3^* for different grids. 36
4.6 Results for function F4 under a finite differences approach, using δ_3^* and $n = 100$ cells. . . 37
4.7 Superfluous decay using the σ-equation with one superfluous point. 40
4.8 Superfluous decay using the σ-equation with two superfluous points. 40
4.9 Superfluous decay using the $\bar{\phi}$-equation with two superfluous points. 41
4.10 Superfluous decay using the $\bar{\phi}$-equation with three superfluous points. 42
4.11 Gathering of all the superfluous decays with σ-equation and $\bar{\phi}$-equation strategies for δ_{2002}. 42
4.12 Gathering of all the global decays using different corrective procedures, for δ_{4206}. 44
4.13 Interface before and after correction, with strategy σ-equation (2), δ_{4206} and $n = 201$. The numerical solutions are represented in orange and the analytical function appears represented in blue. 45

5.1 Illustration of the local coordinate system centred in a given solid point $(x_{sp}, y_{sp}) \in \Gamma$. . . . 48
5.2 Exterior blocks used in the Direct Method. The solid points appear in blue, the orange points come from the first superfluous layer and yellow points come from the second superfluous layer. 50

5.3 Elliptical cloud with semiaxes $s_\eta = 2h$ and $s_\xi = (\frac{w}{2} + 3)h$, for $w = 2$. 55
5.4 Illustration of a 1D-inspired stencil, exemplifying with 2×2 superfluous blocks. 56
5.5 Superfluous blocks used to determine the minimum distance of point ϕ_1^+ from the solid point. 56
5.6 Cartesian local coordinate system (x^*, y^*), with origin at the geometric centre of the blocks. 58
5.7 Illustration of the Cartesian correction along each direction, for the exterior subdomain. . 61

6.1 Graphical representation of function F5, used throughout this chapter. 64
6.2 Results for function F5 under a finite differences approach, using δ_3^* and $N = 100^2$ cells. . 65
6.3 Results for function F5 under a finite volume approach, using δ_3^* and $N = 100^2$ cells. . . . 66
6.4 Superfluous comparison between the IL-FD and the IL-FV for discrete delta function δ_3^*. . 67
6.5 Superfluous comparison between IL-FV and method DM-32, for δ_3^*. 69
6.6 Superfluous comparison between IL-FV and method LSM-Y, for δ_3^*. 72
6.7 Superfluous comparison between IL-FV and method LSM-O, for δ_3^*. 73
6.8 Superfluous comparison between IL-FV and method 1DIM for 4×4 blocks and δ_3^*. 75
6.9 Comparison between the original IL formulation and the best methods, with δ_3^*. 75
6.10 Comparison between corrective procedures C-Poly and C-Cart, on strategy LSM-Y with δ_3^*. 77
6.11 Numerical result obtained by strategy LSM-Y before and after correction C-Poly, using δ_3^*. 78
6.12 Error field obtained by strategy LSM-Y before and after correction C-Poly, using δ_3^*. 78

A.1 Representation of discrete delta function δ_{2002}, in one and two dimensions, taking $h = 1$. . 87
A.2 Representation of discrete delta function δ_{2103}, in one and two dimensions, taking $h = 1$. . 87
A.3 Representation of discrete delta function δ_{4206}, in one and two dimensions, taking $h = 1$. . 88
A.4 Representation of discrete delta function δ_3^*, in one and two dimensions, taking $h = 1$. . . 88

B.1 Layout, along x, of a in the easternmost boundary of the computational domain. 89

C.1 Gathering of all the superfluous decays with σ-equation and $\bar{\phi}$-equation strategies for δ_{2103}. 91
C.2 Gathering of all the superfluous decays with σ-equation and $\bar{\phi}$-equation strategies for δ_3^*. . 92
C.3 Gathering of all the superfluous decays with σ-equation and $\bar{\phi}$-equation strategies for δ_{4206}. 92
C.4 Gathering of all the global decays using different corrective procedures, for δ_{2002}. 93
C.5 Gathering of all the global decays using different corrective procedures, for δ_{2103}. 93
C.6 Gathering of all the global decays using different corrective procedures, for δ_3^*. 94

Acronyms

1D	One-Dimensional.
1D-IM	1D-Inspired Method.
2D	Two-Dimensional.
CFD	Computational Fluid Dynamics.
DM	Direct Method.
FCT	*Fundação para a Ciência e Tecnologia*.
FD	Finite Differences.
FV	Finite Volume.
HIBforMBP	High-order Immersed Boundary method for Moving Body Problems.
IB	Immersed Boundary.
IBM	Immersed Boundary Method.
IBPM	Immersed Boundary Projection Method.
IIM	Immersed Interface Method.
IL	Immersed Layers.
LS	Least Squares.
LSM	Least Squares Method.
ND	No Decay.

Nomenclature

Greek symbols

α, β, γ	Auxiliary variables used in several derivations.
δ	Dirac delta function.
η, ξ	Tangential and normal coordinates in the local referential of a solid point.
κ	Shape factor in the expression used to compute the weights of the least squares method.
λ	Dimension of a multidimensional space.
μ	Parameter set to obtain representative one-dimensional decays.
ν	Number of grid size units between a transition point and its closest superfluous points.
ρ	Order of accuracy (or order of decay).
σ	Normal derivative jump across the interface (or single-layer potential strength).
τ	Variable indicating if a discrete delta function verifies (1) or not (0) the sum of squares condition.
ϕ	Interest unknown scalar field.
φ	Elementary function from which a discrete delta function is obtained.
χ	Indicator function (or level set function).
Γ	Interface separating the fluid and the solid.
Ω	Computational domain.
ε	Error field.
$\boldsymbol{\zeta}$	Coordinate(s) parameterising the interface.
$\boldsymbol{\delta s}$	Vector of surface elements around each solid point.
$\boldsymbol{\Delta} S$	Matrix of surface elements around each solid point.
$\nabla \cdot$	Divergence operator.
∇	Gradient operator.

∇^2 Laplacian operator.

Roman symbols

C Coefficient of a polynomial expansion.

c, d Entries of the submatrices $\mathbf{M}_{\mathbf{C},\mathbf{D}}$.

d_Γ Minimum distance between a given computational point and the interface.

h Grid size.

L Length of the computational domain along each dimension.

m Moment order of a discrete delta function.

m_η, m_ξ Exponents in the last polynomial term of a given expansion to which the tangential and normal coordinates are raised to.

m_{ij} Entries of an inverse matrix, used in several derivations.

N Total number of cells constituting the computational domain.

n Number of cells along each direction of the computational domain.

n_C Number of coefficients used in a polynomial expansion.

n_s Number of sample points used in a direct or least squares method.

P Perimeter of a two-dimensional solid-fluid interface.

p Number of solid points in which the interface is discretised.

p_{ij} Entries of the pseudoinverse matrix in a least squares method.

R Radius of the circular solid body in the two-dimensional problem.

r Ratio between a component of the position vector and the grid size.

S Surface enclosing a computational cell.

s Smoothing order of a discrete delta function.

s_η, s_ξ Tangential and normal semiaxes of an elliptical cloud.

V Volume of a computational cell.

w Support width of a discrete delta function.

x, y, z Cartesian spatial coordinates.

$\mathfrak{n}$ Exterior unit normal of the interface.

$\mathbf{e}$ Unit vector along a given direction.

$\mathbf{n}$ Exterior unit normal of a computational cell.

$\mathbf{r}$ Residual vector.

$\mathbf{u}$ General vector field used in several derivations.

$\mathbf{x}$ Computational point.

$\mathbf{v}$ Right-hand side vector of the discretised system of equations.

$\mathbf{v_a}$ Right-hand side subvector of the discretised Poisson equation.

$\mathbf{v_b}$ Right-hand side subvector of the discretised additional equation to close the system.

$\mathbf{c}$ Vector containing all the coefficients of a polynomial expansion.

$\mathbf{I}$ Identity matrix.

$\mathbf{M}$ Global matrix of the discretised system of equations.

$\mathbf{M_{A,B}}$ Submatrices of the discretised Poisson equation.

$\mathbf{M_{C,D}}$ Submatrices of the discretised additional equation to close the system.

$\mathbf{P_I}$ Pseudoinverse matrix in a least squares method.

$\mathbf{D}$ Distances matrix in a least squares method.

$\mathbf{Int}$ Interpolation matrix.

$\mathbf{Reg}$ Regularisation matrix.

$\mathbf{W_{LS}}$ Weight function matrix in a weighted least squares method.

F General function assuming values exclusively at the interface.

g General function defined for all the multidimensional space.

H Heaviside function.

$\mathbb{N}$ Set of natural numbers.

$\mathbb{R}$ Set of real numbers.

$\mathbb{S}$ Superfluous region.

$\mathbb{T}$ Transition region.

Subscripts

sa Sample point.

sp Solid point.

a One-dimensional grid point that is located immediately after the solid point (in the sense of increasing x) and simultaneously located out of the support width of the discrete delta function.

aa One-dimensional grid point that is located immediately after point a (sense of increasing x).

aaa One-dimensional grid point that is located immediately after point aa (sense of increasing x).

b One-dimensional grid point that is located immediately before the solid point (in the sense of decreasing x) and simultaneously located out of the support width of the discrete delta function.

bb One-dimensional grid point that is located immediately before point b (sense of decreasing x).

bbb One-dimensional grid point that is located immediately before point bb (sense of decreasing x).

BC Boundary condition.

C Cell centroid.

E, W East and west neighbouring grid points.

e, w East and west face centroids of a cell.

F General neighbouring cell centroid (in summation symbols, to particularise into E, W, N, S).

f General cell face (in summation symbols, to particularise into e, w, n, s).

N, S North and south neighbouring grid points.

n, s North and south face centroids of a cell.

η, ξ Tangential and normal components to the solid interface in a two-dimensional problem.

i, j, k Computational indexes.

x, y, z Cartesian components.

Superscripts

$+$ Exterior subfield (fluid region).

$-$ Interior subfield (solid region).

a Analytical value.

l, r Left and right.

LS Least squares method.

T Transpose.

Sup Relative to the superfluous region.

$Trans$ Relative to the transition region.

* Expressed in the local Cartesian referential of the 1D-Inspired Method.

Chapter 1

Introduction

1.1 Motivation

In nature, most processes can be modelled by a set of mathematical relations, that usually take the form of partial differential equations. However, for complex problems, its analytical solution is usually not possible to express or too difficult to obtain. Therefore, numerical methods constitute a powerful tool to compute approximate solutions to the set of equations describing a given phenomenon. Fluid dynamics, for example, covers physical problems that tend to be ruled by highly complex equations, like the Navier-Stokes equations. As such, the discipline of Computational Fluid Dynamics (CFD) arose, to deal with the numerical solving of fluid flows, assuming extreme importance in several engineering domains, like the aerospace sector. From the simulation of a laminar flow over a flat plate, to the simulation of a three-dimensional turbulent flow around an entire aircraft, CFD methods constitute a remarkable help to design, optimise and test aerospace vehicles and components, before going to wind tunnels.

When numerically solving a CFD problem, the domain is subdivided into a given number of cells, for which the system of equations ruling the phenomenon is solved. In that regard, the operators appearing in the equations are locally discretised under a given order of accuracy, leading to a simpler approximate version of the original system of equations. For example, the derivatives in the centroid of a given cell are expressed in terms of field values of some of its nearest neighbours, according to an expression with known accuracy. To reduce the numerical error, one may refine the mesh, or use formulations of higher accuracy. The mesh refinement makes the distance between centroids to decrease and, therefore, leads to a lower and lower relevance of the high-order terms that are neglected in the discretisation. The use of formulations with higher accuracy consists in keeping more high-order contributions in the discretisation of operators, keeping them closer and closer to their analytical definition. Given the fact that the computational power is always limited, the finest mesh refinements to perform the numerical simulations will also be associated with maximum admissible computational times and memory. Thus, formulations with high-order accuracy may be a promising alternative to achieve better numerical results.

This Thesis aims to derive one-dimensional (1D) and two-dimensional (2D) accuracy improvement strategies to the Immersed Layers (IL) method developed by Eldredge [1], applied to a Poisson problem.

1.2 Background

The most common CFD problems consist of fluid flows passing around solid bodies and, in conventional approaches, the mesh is generated in such a way that it conforms to the solid body. However, when dealing with problems in which the solid presents a prescribed motion or problems in which it is allowed to deform due to the forces exerted by the fluid, the solid-fluid interface must be constantly updated. Thus, a remeshing process needs to be performed at the beginning of each time step, in order to generate a new mesh that conforms the new interface, which presents a high computational cost.

In article [2], Peskin introduced the Immersed Boundary Method (IBM), to study flow patterns around heart valves, modelled as elastic massless moving boundaries, immersed inside the computational domain, exerting forces in the fluid. To model the effect of the interface, a singular forcing term arises in the equations, that consists of an integral over the immersed boundary relying on the Dirac delta function. To numerically deal with this term, the Dirac delta function is substituted by a discrete analogue, the interface is discretised into a finite number of points and the integral over the interface is converted into a summation over those points. With this, information stored in the immersed points is regularised onto the grid. Moreover, to impose the boundary conditions at the interface, the velocity field must be interpolated from the grid to the solid points and this operation is performed using a discrete delta function as well. After that, Peskin performed a numerical analysis of blood flow in the heart, [3], and Peskin and Lai applied the IBM to simulate the flow past a cylinder, in [4]. Later on, the detailed mathematical structure of the IBM was provided by Peskin in [5].

After the works of Peskin, several methods arose in which the mesh also does not need to conform to the body. As such, nowadays, the expression *Immersed Boundary* (IB) *methods* encompasses all those methods in which the interface is immersed into the grid, as stated by Mittal and Iccarino in article [6]. There, the authors divide IB methods into two main categories: continuous forcing and discrete forcing methods. In the continuous forcing methods, a forcing term is added to the equations ruling the flow in the entire domain, prior to the discretisation. On the other hand, in the discrete forcing methods, the ruling equations are discretised without considering any additional forcing term and, then, only the discretised equations of cells near the IB receive forcing terms, to ensure the fulfilment of the boundary conditions on the immersed surface. The original IBM of Peskin [2] belongs to the first family. Sotiropoulos and Yang [7] also summarise several IB methods, organising them into two main families: diffused interface methods and sharp interface methods. Once more, the original IBM of Peskin belongs to the first family, in which the distribution of singular forces to the grid via discrete delta functions prevents obtaining sharp variations, being said to *smear* the interface, [7].

Given that the equations to be discretised in the continuous forcing methods are the same for all the cells and no additional adaptations are needed, these methods are usually simpler to implement. However, as explained in [6], continuous forcing methods suffer from lower accuracy near the immersed boundary, since they are unable to represent sharp interfaces. The substitution of the Dirac delta function by a discrete analogue, that constitutes a continuous distribution mimicking the singularity, is always responsible for smearing the interface of non-smooth fields (that is, fields with a discontinuity in value

and/or a discontinuity in normal derivative across the interface). As an example, Peskin and Griffith [8] implement an IB method with discrete delta functions and report second-order accuracy as long as the solution is smooth. In [5], Peskin states that methods lying on the use of discrete delta functions to perform interpolation operations will present global first-order accuracy for non-smooth fields across the interface. To obtain second-order accuracy, Peskin refers to the Immersed Interface Method (IIM) derived by Leveque and Li in [9]. However, this method does not belong to the continuous forcing family, since it only computes corrective terms for the discretised equations ruling cells in neighbourhood of the interface. To achieve second-order, the expression of the truncation error that is originated by using the usual schemes in points near the interface is determined (via polynomial expansions for each interface side and knowing the jump conditions in value and normal first derivative) and corrective terms are computed so that undesirable terms cancel out. This method is able to lead to sharp interfaces. In later works, this method was extended to deal with the Navier-Stokes equations, as presented in [10] by Xu and Wang, solving an incompressible viscous flow with second-order spatial and temporal accuracy near the boundaries. Another second-order accurate implementation of the IIM to solve the incompressible Navier-Stokes equation is found in [11], by Li and Lai. Recently, Balam and Zapata [12] derived a fourth-order IIM formulation with compact schemes capable of solving two-dimensional Poisson problems.

When compared to continuous forcing methods, discrete forcing methods are more complex to implement, since a forcing term must be derived for each cell near the interface after the discretisation, but they are able to represent sharp interfaces. Moreover, contrary to what happens with continuous forcing, these methods are not limited to first-order accuracy in case of non-smooth fields, [6], with their performance depending on the accuracy of the methods used to compute the forcing terms.

Taira and Colonius [13] derived an Immersed Boundary Projection Method (IBPM) consisting of a general framework to deal with rigid bodies subjected to a prescribed motion. This method does not distinguish the interior from the exterior side of the interface, setting a given motion constraint at the interface that is valid for both sides. This causes the IBPM to be unable to distinguish between forces exerted at the interface by the exterior fluid subdomain from forces exerted by the interior solid subdomain. According to Eldredge [1], when dealing with deformable bodies, it is not possible to equal the force exclusively exerted by the fluid to the one arising from the elastic equations that rule the deformation, since integrals along the surface consist of a mix of contributions from both sides. Moreover, Eldredge [1] shows that problems also arise when considering a rigid body rotation, since the interior velocity field does not correspond to the true one of a rigid body rotating. As such, when taking integrals over the interface to compute the force and the moment exerted by the fluid, the incorrect force component coming from the interior subdomain originates incorrect results.

To tackle the problems that the IBPM is not able to solve, Eldredge [1] derived a more general formulation, the Immersed Layers Method, in which the interior and exterior subfields are distinguished, enabling the specification of different constraints on each side of the interface. The IL method belongs to the continuous forcing family, using discrete delta functions to regularise forcing terms onto the grid. The original IL formulation is presented in this work, for the Poisson equation. When deriving the IL Poisson equation, the discontinuity in normal derivative across the interface emerges in the expression, being

treated as an unknown variable. Eldredge [1] closes the system of equations by imposing the field value at the interface, via an interpolation operation using discrete delta functions. As predicted by Peskin [5], this method presents first-order accuracy when the field is non-smooth across the interface. The constraints imposed by the IL method at both sides of the interface are field values, that is, Dirichlet type boundary conditions. Methods imposing Neumann type boundary conditions at an immersed interface also exist, as the Immersed Boundary Double Layer Method, recently developed by Leathers [14].

In [5], Peskin emphasises the role of discrete delta functions used to perform interpolations of non-smooth fields in the reversal to first-order accuracy. As such, in this work, the second equation used by Eldredge [1] is discarded and replaced by another one not relying on discrete delta functions, to find out if some accuracy improvement is achieved. As one will see, since discrete delta functions continue to be used in the Poisson equation (performing regularisation operations), first-order decays continue to be obtained in the entire domain, but significant accuracy improvements are achieved away from the interface, that is, in the region over which the discrete delta functions are not acting directly.

1.3 Objectives

Given the first-order accuracy of the IL method, this work aims to achieve second-order accuracy in one- and two-dimensional problems. To reach this main objective, several minor ones were set:

- Implementation of the IL method in its original formulation, in a finite differences (FD) framework, in order to replicate the first-order decay that was reported by Eldredge in article [1];
- Derive, implement and test a finite volume (FV) approach to the IL method, both in one and two dimensions, comparing its results with the ones arising from finite differences;
- Derive, implement and test alternative second equations to close the system originated by the appearance of an unknown in the IL Poisson equation, in one and two dimensions, pursuing second-order accuracy away from the interface, even when dealing with non-smooth fields;
- Derive, implement and test one- and two-dimensional corrective procedures, to be executed in the end of the numerical simulation, that are able to recover interface sharpness and propagate second-order decays to the points near the interface and, consequently, to the entire domain.

1.4 Present Contributions

The first objective, about the finite differences implementation of the IL method, constitutes a replication of the formulation derived by Eldredge [1]. All the other objectives are contributions of this work.

Regarding the finite volume approach, the equations coming from the original IL formulation are integrated, simplified via the divergence theorem and discretised. In what concerns the one- and two-dimensional improvement strategies, alternative additional equations are derived, making use of polynomial expansions around each solid point, to express field values or normal first derivatives in terms of its grid and/or solid neighbours. The polynomial coefficients are computed with direct methods or Least Squares (LS) formulations. In the corrective procedures, only direct methods are applied.

1.5 Thesis Outline

This Thesis is subdivided into seven chapters. The first one, which this outline concludes, concerns an introduction to the dissertation, starting with a brief motivation on the use of numerical methods to simulate physical problems, namely in Fluid Dynamics. After that, some background on IB methods is provided, the objectives of this work are enumerated and the present contributions are pointed out.

The second chapter aims to provide the reader a sufficient theoretical background concerning the IL method, developed by Eldredge in article [1], when applied to Poisson problems. In that regard, the main derivations of the method are presented, in the original finite differences formulation. Moreover, an alternative finite volume formulation is also presented, constituting a contribution of the present work.

In the third chapter, several one-dimensional strategies to increase the accuracy of the IL Method are derived. This improvement is achieved by means of considering two additional equations to close the system, that are different from the one considered in article [1]. Furthermore, a final corrective step is suggested, in order to recover the interface sharpness and to propagate second-order accuracy to the cells near the interface and, therefore, ensuring second order in the entire domain.

In the fourth chapter, the one-dimensional improvements strategies are compared with the original formulation and among themselves, so that the best procedure can be determined. Then, the several corrective strategies are compared and the one leading to the lowest error is identified.

In the fifth chapter, taking into account the one-dimensional results, the additional equation that presented the best results is generalised for two-dimensional problems, using several possible strategies, namely a Direct Method (DM), a weighted Least Squares Method (LSM) and a method faithfully replicating the one-dimensional reasoning. A two-dimensional final corrective procedure is also derived.

The sixth chapter presents the numerical results of the two-dimensional improvement strategies, by determining which one leads to the best accuracy and lowest error. A comparison between the final corrective procedures that were derived to be applied in two-dimensional problems is performed.

In the seventh chapter, the main conclusions of this work are highlighted and some future work tasks are suggested, in order to improve the current results even further, specially in two dimensions.

Chapter 2

Immersed Layers Method

In this chapter, the Immersed Layers Method developed by Eldredge in article [1] is presented, aiming to provide the reader a theoretical background concerning its original formulation. The presentation provided in this chapter intends to be concise and self-sufficient, emphasising the particular case of the Poisson equation. Naturally, the reading of article [1] is always encouraged, as a wider number of applications is explored there (the convection-diffusion equation and the incompressible Navier-Stokes equations) and a whole appendix on generalised functions and discretised operators is provided.

Firstly, in subchapter 2.1, the description of a general immersed boundary problem is presented, with a brief geometrical insight. In subchapter 2.2, an indicator function is defined for all the domain, in order to readily distinguish the solid and fluid subdomains from each other. In subchapter 2.3, some generalised functions are reminded, namely the Heaviside function and the Dirac delta function, along with some relevant properties. In subchapter 2.4, the concept of masked variable is introduced, by making use of generalised functions applied to the previously defined indicator function.

In subchapter 2.5, the Poisson equation of a masked variable is derived, by computing first the gradient and then the Laplacian of a variable written in the masked form. In subchapter 2.6, the numerical implementation of the derived Poisson equation is discussed. Since one of the terms arising in the masked Poisson equation will be treated as an unknown, this subchapter begins by closing the system with an additional equation. Then, two discretisation strategies are presented: a finite differences approach, similar to what was performed in article [1], and an alternative finite volume one, that constitutes a contribution of the present work. Each discretisation strategy will lead to a different grid layout, a different treatment of the system of equations and, thus, to different operators needing discretisation.

2.1 Problem Description

The domain of immersed boundary methods addresses physical problems in which at least two media with different properties coexist separated by an interface. Despite the possibility of considering geometries with huge complexity, with several solid bodies and even several fluids, the simplest and most studied example consists of one solid body immersed within a single fluid. For the sake of sim-

plicity and easier numerical treatment in the further up deduced improvement strategies, this is the problem addressed throughout this work. In general, one may consider problems in one, two, or three dimensions, that is, the computational domain Ω is such that $\Omega \subset \mathbb{R}^{\lambda}$ with $\lambda = 1, 2, 3$. In figure 2.1, the illustration of a two-dimensional immersed boundary problem is provided.

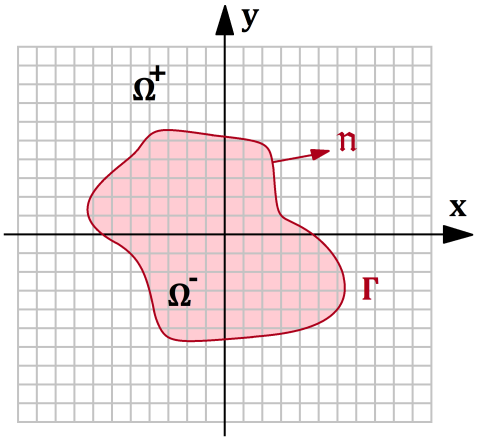


Figure 2.1: Illustration of a general two-dimensional problem ($\lambda = 2$).

As illustrated in figure 2.1, the computational domain Ω is assumed to be composed of an interior subdomain Ω^{-} corresponding to the solid and an exterior subdomain Ω^{+} corresponding to the fluid, both separated by a smooth interface Γ. The exterior unit normal of the solid surface, $\mathfrak{n}$, is also represented. For a general multidimensional space $\mathbb{R}^{\lambda}$, the position vector $\mathbf{x}$ of any point inside the computational domain is given by $\mathbf{x} = \sum_{i=1}^{\lambda} x_i \mathbf{e}_i$, with $\mathbf{e}_i$ being the unit vector along direction i. Throughout this work, the position vector components x_1, x_2, x_3 will be identified as x, y, z, respectively.

2.2 Indicator Function

The first step to be taken consists of identifying the solid and the fluid regions, or, in other words, the interior and the exterior subdomains, respectively. For that, an indicator (or level set) function $\chi(\mathbf{x}) \in \mathbb{R}$ is defined for all the domain, assuming negative values in the solid, positive values within the fluid and being null when taken at the interface Γ. In this work, the indicator function was defined by making use of a properly signed minimum Euclidean distance to the interface Γ, as presented in equation 2.1. For a given point $\mathbf{x} \in \Omega$, one defines $\mathbf{x}_{\mathbf{sp}}^{min}(\mathbf{x})$ as a point at the interface surface for which the distance between $\mathbf{x}$ and the interface reaches the minimum value, that is, $\mathbf{x}_{\mathbf{sp}}^{min}(\mathbf{x}) = \{\mathbf{x}_{\mathbf{sp}} \in \Gamma : ||\mathbf{x} - \mathbf{x}_{\mathbf{sp}}^{min}|| = \min ||\mathbf{x} - \mathbf{x}_{\mathbf{sp}}||\}$.

$$\chi(\mathbf{x}) = \begin{cases} ||\mathbf{x} - \mathbf{x}_{\mathbf{sp}}^{min}(\mathbf{x})|| & , \ \mathbf{x} \in \Omega^{+} \\ -||\mathbf{x} - \mathbf{x}_{\mathbf{sp}}^{min}(\mathbf{x})|| & , \ \mathbf{x} \in \Omega^{-} \end{cases} \tag{2.1}$$

Defining the indicator function this way, it is essential to notice that its gradient is normal to the interface, points out of the solid and has a unitary magnitude. That is, when taken at the interface, the gradient of the indicator function corresponds to the exterior unit normal of the solid surface, $\nabla\chi|_{\Gamma} = \mathfrak{n}$.

2.3 Generalised Functions

The relations arising from the Immersed Layers Method use two important generalised functions: the Heaviside function and the Dirac delta function. As one will see later on, the first one is responsible for the identification of each subdomain, whereas the second plays an important role in the terms arising to model the influence of the interface. Both functions act over real objects and, in the context of the Immersed Layers Method, both appear applied to the indicator function, $\chi(\mathbf{x}) \in \mathbb{R}$. For the sake of compactness, the dependency of the indicator function on the position will be dropped most of the times. The Heaviside function, $H(\chi)$, is defined in equation 2.2. Notice that $H(\chi) = 1 - H(-\chi)$.

$$H(\chi) = \begin{cases} 1 & , \chi > 0 \\ 1/2 & , \chi = 0 \\ 0 & , \chi < 0 \end{cases} \tag{2.2}$$

On the other hand, the definition of the Dirac delta function, $\delta(\chi)$, is provided in equation 2.3. Notice that this function is singular, with an infinite value at one point of the real line. Moreover, $\delta(\chi) = \delta(-\chi)$. Recall that the integral of the Dirac delta function along the real line in unitary, that is, $\int_{\mathbb{R}} \delta(\chi) d\chi = 1$. The derivation of this last property concerning the integral over $\mathbb{R}$ may be consulted in appendix A.2.

$$\delta(\chi) = \begin{cases} +\infty & , \chi = 0 \\ 0 & , \chi \neq 0 \end{cases} \tag{2.3}$$

It is also possible to define a multidimensional version of the Dirac delta function, acting on vector objects. Thinking in a vector belonging to a general multidimensional space of dimension λ, that is $\mathbf{x} \in \mathbb{R}^\lambda$ with $\lambda = \{1, 2, 3\}$, one defines the multidimensional Dirac delta function as the multiplication between the one-dimensional Dirac delta applied to each dimension, as stated in equation 2.4. By definition, the multidimensional Dirac delta inherits the property of unitary integral over $\mathbb{R}^\lambda$, that is, $\int_{\mathbb{R}^\lambda} \delta(\mathbf{x}) d\mathbf{x} = 1$.

$$\delta(\mathbf{x}) = \prod_{i=1}^{\lambda} \delta(x_i) \tag{2.4}$$

This multidimensional version allows the derivation of important relations converting surface integrals over the interface Γ into volume integrals over the entire space $\mathbb{R}^\lambda$ and vice-versa. By making use of manipulations of this kind, it is possible to reach two important results, whose derivation is presented in appendix A.2, closely following the deductions presented in article [1]. Consider that the interface Γ is parameterised by surface coordinate(s) $\boldsymbol{\zeta}$, so that interface points are given in $\mathbb{R}^\lambda$ by $\mathbf{x} = \mathbf{X}(\boldsymbol{\zeta})$. The first important derived relation concerns the so-called immersion of interface information into the multi-dimensional space. Consider a given function $F(\boldsymbol{\zeta})$, exclusively defined at the interface. The immersion of F into the multidimensional space $\mathbb{R}^\lambda$, denoted by $F(\boldsymbol{\zeta})\delta\big(\chi(\mathbf{x})\big)$, is given by equation 2.5.

$$F(\boldsymbol{\zeta})\delta\big(\chi(\mathbf{x})\big) = \int_{\Gamma} F(\boldsymbol{\zeta})\delta\big(\mathbf{x} - \mathbf{X}(\boldsymbol{\zeta})\big)dS(\boldsymbol{\zeta}) \tag{2.5}$$

The second important relation concerns the so-called restriction of a field defined in the multidimensional space to the interface. Consider a given function $g(\mathbf{x})$, defined in the multidimensional space. The restriction of g to the interface Γ, with notation $g(\mathbf{x})\delta^T\big(\chi(\mathbf{x})\big)$ as in article [1], is given by equation 2.6.

$$g(\mathbf{x})\delta^T\big(\chi(\mathbf{x})\big) \equiv g\big(\mathbf{X}(\boldsymbol{\zeta})\big) = \int_{\mathbb{R}^\lambda} g(\mathbf{x})\delta\big(\mathbf{X}(\boldsymbol{\zeta}) - \mathbf{x}\big)d\mathbf{x} \tag{2.6}$$

As suggested by figure 2.1, the interface Γ is completely located inside the computational domain $\Omega \subset \mathbb{R}^\lambda$. Therefore, since the Dirac delta function is null when it is not evaluated at the interface, the integrals over $\mathbb{R}^\lambda$ may interchange with integrals over Ω. As one will see further up, since it is impossible to numerically deal with singularities, the Dirac delta function will be converted into a discrete delta function, acting in a small known vicinity of the interface. Thus, to keep the interchangeability of integrals over $\mathbb{R}^\lambda$ and over Ω valid, one simply needs to define the computational domain in such way that the interface keeps a sufficient distance from its boundaries.

As a final remark, note that, according to the definitions of the Heaviside and Dirac delta functions in equations 2.2 and 2.3, these functions may be related by $\frac{\partial H}{\partial \chi}(\chi) = \delta(\chi)$. So, the gradient of the Heaviside function, $\nabla H(\pm\chi)$, may be expressed in terms of the Dirac delta function, as in equation 2.7.

$$\nabla H(\pm\chi) = \pm\frac{\partial H}{\partial \chi}\nabla\chi = \pm\delta(\chi)\nabla\chi = \pm\delta(\chi)\nabla\chi|_\Gamma = \pm\delta(\chi)\mathfrak{n} \tag{2.7}$$

2.4 Concept of Masked Variable

Having already defined an indicator function and reminded some important generalised functions, the identification of the interior and exterior subdomains is readily performed by applying the Heaviside function to the indicator. Recalling the definition provided in equation 2.2, it is straightforward to conclude that $H(\chi)$ is unitary in the fluid region and null in the solid, whereas $H(-\chi)$ is unitary in the solid region and null in the fluid. At the interface, both $H(\chi)$ and $H(-\chi)$ are equal to $1/2$.

In theory, the interior and exterior subfields may behave completely differently, with values and derivatives being different from each other in the vicinity of the interface. For this reason, it is pertinent to define a masked variable, $\bar{\phi}$, that assumes one subfield or the other, according to the subdomain in which it is evaluated. The definition, relying on the use of Heaviside functions, is provided in equation 2.8.

$$\bar{\phi} = \phi^+ H(\chi) + \phi^- H(-\chi) \tag{2.8}$$

Given the definition of masked variable, for each subdomain one has $\mathbf{x} \in \Omega^\pm \Rightarrow \bar{\phi}(\mathbf{x}) = \phi^\pm(\mathbf{x})$. On the other hand, at the interface, the expression leads to $\mathbf{x} \in \Gamma \Rightarrow \bar{\phi}(\mathbf{x}) = \frac{1}{2}[\phi^+(\mathbf{x}) + \phi^-(\mathbf{x})]$. Note that, by working with this masked variable, one is able to treat problems for the entire domain at once, without the need of caring about each subdomain independently.

2.5 Poisson Equation of a Masked Variable

Assuming that the field $\bar{\phi}$ is piecewise twice-differentiable in each subdomain, one is able to derive the Poisson equation of a masked variable. In that regard, before reaching the Laplacian, one will firstly focus on deducing the gradient expression of a masked variable. Making use of relation 2.7, the application of the chain derivation rule leads to the gradient expression of equation 2.9.

$$\begin{aligned}
&\nabla\bar{\phi} = \nabla\phi^+ H(\chi) + \phi^+\nabla H(\chi) + \nabla\phi^- H(-\chi) + \phi^-\nabla H(-\chi) \\
\Leftrightarrow\ &\nabla\bar{\phi} = \Big(\nabla\phi^+ H(\chi) + \nabla\phi^- H(-\chi)\Big) + \phi^+\nabla H(\chi) + \phi^-\nabla H(-\chi) \\
\Leftrightarrow\ &\nabla\bar{\phi} = \overline{\nabla\phi} + (\phi^+ - \phi^-)\delta(\chi)\mathfrak{n}
\end{aligned} \tag{2.9}$$

Note that the first term in equation 2.9 corresponds to the masked gradient, taking into account the gradients of each subfield independently. The second term accounts for the influence of the discontinuity in value between subfields, ensuring that the gradient readjusts to the correct value when the interface is crossed. By applying the divergence operator to equation 2.9, one obtains the Laplacian expression, presented in equation 2.10. This relation corresponds to the Poisson equation of a masked variable.

$$\begin{aligned}
&\nabla^2\bar{\phi} = \nabla\cdot\left[\Big(\nabla\phi^+ H(\chi) + \nabla\phi^- H(-\chi)\Big) + (\phi^+ - \phi^-)\delta(\chi)\mathfrak{n}\right] \\
\Leftrightarrow\ &\nabla^2\bar{\phi} = \nabla^2\phi^+ H(\chi) + \nabla\phi^+\cdot\nabla H(\chi) + \nabla^2\phi^- H(-\chi) + \nabla\phi^-\cdot\nabla H(-\chi) + \nabla\cdot\Big((\phi^+ - \phi^-)\delta(\chi)\mathfrak{n}\Big) \\
\Leftrightarrow\ &\nabla^2\bar{\phi} = \overline{\nabla^2\phi} + (\nabla\phi^+ - \nabla\phi^-)\cdot\delta(\chi)\mathfrak{n} + \nabla\cdot\Big((\phi^+ - \phi^-)\delta(\chi)\mathfrak{n}\Big)
\end{aligned} \tag{2.10}$$

The first term in equation 2.10 is the masked Laplacian, a known quantity. However, the adjustment across the interface is now ensured not only by a term relying on the discontinuity in value, but also by a term relying on the discontinuity in normal derivative across the interface. In article [1], Eldredge names these terms as the double- and single-layer potentials, respectively. The double-layer potential strength corresponds to $(\phi^+ - \phi^-)$, whereas the single-layer potential strength is given by $(\nabla\phi^+ - \nabla\phi^-)\cdot\mathfrak{n}$.

Usually, when dealing with a single well-behaved field, its Poisson equation simply corresponds to writing that the Laplacian of the unknown variable is equal to a given known source term. Note that, in equation 2.10, this is what happens when one considers the interior or the exterior subfields alone. Indeed, away from the interface, the Dirac delta function makes the additional terms disappear, whereas the Heaviside function makes the masked Laplacian to assume the correct source term inside each subdomain. Far from the interface, general relation 2.10 degenerates into $\mathbf{x}\in\Omega^\pm \ \Rightarrow\ \nabla^2\bar{\phi} = \nabla^2\phi^\pm$.

Notice how complicated it would be to independently handle the Poisson problems that occur inside each subdomain. In an immersed boundary context, the interface does not correspond to cell faces, and the imposition of boundary conditions there would be extremely complex to perform. On the other hand, the Poisson equation 2.10 allows to work with a single problem, expressed in terms of a global masked variable $\bar{\phi}$, being the interface treatment automatically performed by the two additional terms that arise.

2.6 Implementation Strategy of the Masked Poisson Equation

2.6.1 Closing the System of Equations

In the numerical implementation of the Poisson equation 2.10, the interface Γ is discretised into a given number of solid points, whose position and exterior normal to the surface are known. In this work, the solid points are assumed to be equally spaced, placed at a distance δs from each other. Therefore, for each solid point, it is possible to evaluate the exterior and interior subfield values, ϕ^+_{sp} and ϕ^-_{sp}, as well as the exterior and interior subfield gradients, $\nabla\phi^+_{sp}$ and $\nabla\phi^-_{sp}$, respectively.

The original formulation of the Immersed Layers Method, in article [1], assumes that only the values of each subfield are known at the interface. Thus, the double-layer potential strength at each solid point, $(\phi^+_{sp} - \phi^-_{sp})$, is a known quantity, whereas the single-layer potential strength should be treated as an unknown during the implementation, $\sigma \equiv (\nabla\phi^+_{sp} - \nabla\phi^-_{sp}) \cdot \mathfrak{n}$. Treating σ as an unknown translates to adding one extra unknown for each solid point. Consequently, one needs to express, as well, one additional equation for each solid point, in order to close the system of equations.

The original formulation of the method closes the system by stating that, when restricted to a given solid point, the masked field must recover the interface value, given by the average of the exterior and interior subfield values at that same solid point. The restriction operation is performed by making use of Dirac delta functions. As such, the original formulation of the Immersed Layers Method, when applied to a Poisson problem with unknown single-layer potential strength, closes the system with equation 2.11.

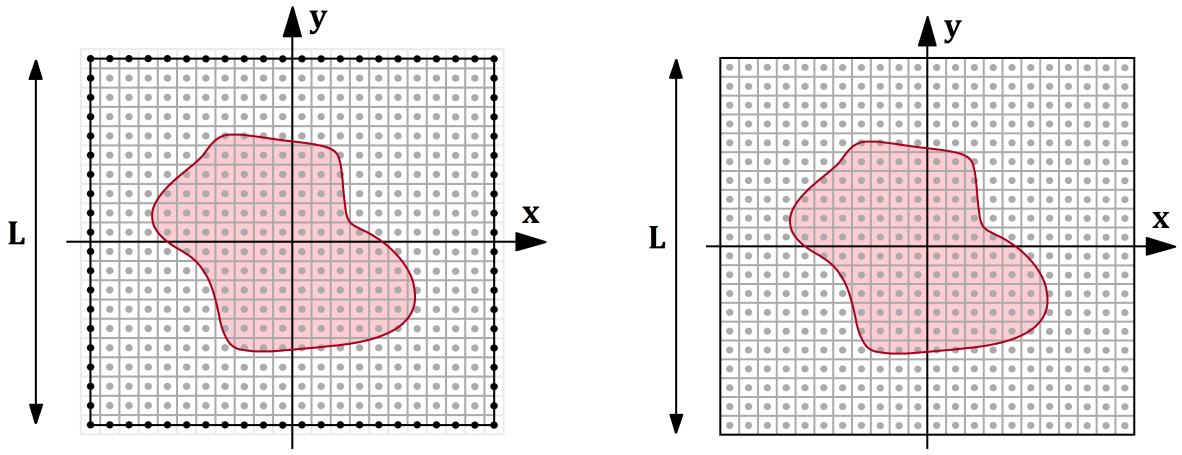
$$\bar{\phi}\,\delta^T(\chi) = \frac{1}{2}(\phi^+_{sp} + \phi^-_{sp}) \tag{2.11}$$

In the upcoming chapters, the use of this relation to close the system of equations will be questioned and new expressions will be derived in search of numerical results with greater accuracy, for both the one-dimensional and the two-dimensional cases.

2.6.2 Domain Discretisation

In order to numerically solve the system of equations arising from the Poisson equation 2.10 together with equation 2.11, the computational domain needs to be discretised into a finite number of cells. Throughout this work, the computational domain Ω was considered to have a length L along each direction of $\mathbb{R}^\lambda$, being subdivided into a Cartesian grid with n square cells along each direction.

Given these assumptions, there are two main ways to discretise the computational domain: under a finite differences reasoning or under a finite volume reasoning. These two possible strategies are illustrated in figure 2.2, for a two-dimensional general problem. In the finite differences approach, there are cell centroids coinciding with the domain boundaries. On the other hand, in the finite volume approach, the boundary of the domain coincides with cell faces. This difference will translate into distinct procedures to implement the Dirichlet boundary conditions at the computational domain boundary, $\partial\Omega$.



(a) Finite differences approach.

(b) Finite volume approach.

Figure 2.2: Finite differences and finite volume approaches used to discretise the computational domain.

For a general domain $\Omega \subset \mathbb{R}^\lambda$, the total number of cells is given by n^λ. The characteristic length of each cell, or grid size h, is given by $h = L/(n-1)$ in finite differences and by $h = L/n$ in finite volume.

2.6.3 Treatment of the System of Equations

Each domain discretisation strategy is associated with a different treatment of the equations describing the Poisson problem. The finite differences strategy is the one followed in article [1]. In that regard, the relations derived for the finite volume implementation constitute a contribution of the present work. The general finite volume equations are derived and then particularised into second-order accuracy.

Finite Differences

By establishing the single-layer potential strength as an unknown in the Poisson equation and by setting the second equation 2.11, relying on the restriction of the global field to the interface, one arrives to the final system to be numerically solved, as presented in equations 2.12.

$$\begin{cases} \nabla^2\bar{\phi} - \sigma\delta(\chi) = \overline{\nabla^2\phi} + \nabla\cdot\left((\phi_{sp}^+ - \phi_{sp}^-)\delta(\chi)\mathfrak{n}\right) \\ \bar{\phi}\,\delta^T(\chi) = \frac{1}{2}(\phi_{sp}^+ + \phi_{sp}^-) \end{cases} \tag{2.12}$$

This system may be displayed in the matrix form of equation 2.13, in which the square matrix gathers the operations being performed to the unknowns: Laplacian ∇^2, immersion $\delta(\chi)$ and restriction $\delta^T(\chi)$.

$$\begin{pmatrix} \nabla^2 & \delta(\chi) \\ \delta^T(\chi) & 0 \end{pmatrix} \begin{pmatrix} \bar{\phi} \\ -\sigma \end{pmatrix} = \begin{pmatrix} \overline{\nabla^2\phi} + \nabla\cdot\left((\phi_{sp}^+ - \phi_{sp}^-)\delta(\chi)\mathfrak{n}\right) \\ \frac{1}{2}(\phi_{sp}^+ + \phi_{sp}^-) \end{pmatrix} \tag{2.13}$$

Under the finite differences approach, the discretisation will be directly applied on these equations, without any previous treatment. The operators appearing in the square matrix, as well as the immersion and the divergence operator showing up in the right-hand side vector, will need to be discretised.

Finite Volume

Under a finite volume approach, the system of equations 2.12 should be integrated over each cell of the computational domain. Consider that any square cell is characterised by a volume V, enclosed by a surface $S = \partial V$. Integrating equation 2.10 in volume and applying the divergence theorem (recalled in appendix B.1), one arrives to the Poisson equation 2.14, written in a finite volume approach for the most general three-dimensional case. The vector $\mathbf{n}$ stands for the exterior unit normal of surface S. Notice that the last volume integral cannot be transformed into a surface integral according to the divergence theorem, since the vector field at which the divergence is applied is not continuously differentiable.

$$\begin{aligned} \iiint_V \nabla^2 \bar{\phi}\, dV &= \iiint_V \left[\overline{\nabla^2 \phi} + \sigma\delta(\chi) + \nabla \cdot \left((\phi_{sp}^+ - \phi_{sp}^-)\delta(\chi)\mathfrak{n} \right) \right] dV \\ \Leftrightarrow \oiint_{\partial V} \nabla\bar{\phi} \cdot \mathbf{n}\, dS - \iiint_V \sigma\delta(\chi)\, dV &= \iiint_V \overline{\nabla^2 \phi}\, dV + \iiint_V \nabla \cdot \left((\phi_{sp}^+ - \phi_{sp}^-)\delta(\chi)\mathfrak{n} \right) dV \end{aligned} \tag{2.14}$$

Concerning additional equation 2.11, the finite volume procedure will simply result in equation 2.15.

$$\iiint_V \bar{\phi}\delta^T(\chi)\, dV = \iiint_V \frac{1}{2}(\phi_{sp}^+ + \phi_{sp}^-)\, dV \tag{2.15}$$

Notice that equations 2.14 and 2.15 constitute the starting point for a general finite volume implementation of a masked Poisson problem. In this work, since one is pursuing second-order results, the equations will be subjected to an additional treatment, wherein all the integrals are approximated with second-order accuracy. This step gives rise to the system of equations 2.16. The subscript C stands for cell centroid, whereas subscript f stands for cell face. More specifically, in the one-dimensional case, the sum over f particularises in east and west faces (e and w, respectively). In the two-dimensional case, the sum over f particularises in east, west, north and south faces (e, w, n and s, respectively).

$$\begin{cases} \sum_f \left[(\nabla\bar{\phi})_f \cdot \mathbf{n_f} \right] - \sigma\delta(\chi_C)h^\lambda = \left(\overline{\nabla^2\phi}\right)_C h^\lambda + \nabla \cdot \left[(\phi_{sp}^+ - \phi_{sp}^-)\delta(\chi_C)\mathfrak{n} \right] h^\lambda \\ \bar{\phi}\delta^T(\chi_C)h^\lambda = \frac{1}{2}(\phi_{sp}^+ + \phi_{sp}^-)h^\lambda \end{cases} \tag{2.16}$$

Under this finite volume approach, in which the integrals were treated with second-order accuracy, system of equations 2.16 is the starting point to proceed with the discretisation of operators. Contrary to what happened in finite differences, the Laplacian is not present in the left side of the first equation and a gradient operator, ∇, arises. The immersion, restriction and divergence operators are still present.

2.6.4 Operators Discretisation

In this section, the operators arising in the systems of equations obtained following a finite differences or finite volume approach are discretised with second-order accuracy. First, the Dirac delta function is converted into a discrete delta function. Then, the immersion and restriction operators are discretised. The discretisation follows with the Laplacian, the gradient and finally the divergence. The discretisation of all these operators is presented for both one- and two-dimensional scenarios. In the end of this section, a comment on the numerical implementation of the Heaviside function is made.

The objective of the discretisation process is to express systems 2.12 and 2.14 in the matrix form of equation 2.17, where discretised operators acting on unknowns are represented by matrices $\mathbf{M_A}$, $\mathbf{M_B}$, $\mathbf{M_C}$, $\mathbf{M_D}$ and operators appearing in the right-hand side are contained in vectors $\mathbf{v_a}$ and $\mathbf{v_b}$, alongside with constant source terms. In this work, this system is solved by inverting the global $\mathbf{M}$ matrix.

$$\begin{pmatrix} [\mathbf{M_A}]_{N\times N} & [\mathbf{M_B}]_{N\times p} \\ [\mathbf{M_C}]_{p\times N} & [\mathbf{M_D}]_{p\times p} \end{pmatrix} \begin{pmatrix} \{\bar{\phi}\}_{N\times 1} \\ \{\boldsymbol{\sigma}\}_{p\times 1} \end{pmatrix} = \begin{pmatrix} \{\mathbf{v_a}\}_{N\times 1} \\ \{\mathbf{v_b}\}_{p\times 1} \end{pmatrix} \quad \Leftrightarrow \quad \mathbf{M} \begin{pmatrix} \bar{\phi} \\ \boldsymbol{\sigma} \end{pmatrix} = \mathbf{v} \tag{2.17}$$

Matrices $\mathbf{M_A}$ and $\mathbf{M_B}$, together with right-hand side vector $\mathbf{v_a}$, gather the Poisson equation for all the N cells of the computational domain. On the other hand, $\mathbf{M_C}$ and $\mathbf{M_D}$, together with right-hand side vector $\mathbf{v_b}$, gather the additional equation written for all the p solid points, in order to close the system.

For cells in the boundary of the computational domain, the Poisson equation will be adapted to incorporate a Dirichlet boundary condition. The adaptation depends on whether the strategy under use is a finite differences or finite volume approach. Given equation 2.11, both approaches have $\mathbf{M_D} = \mathbf{0}$.

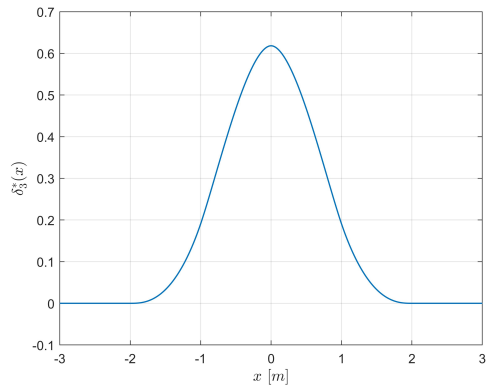
Dirac Delta Function

The multidimensional Dirac delta function, previously defined in equation 2.4, presents a singular behaviour. Since it is impossible to numerically deal with infinite values, the Dirac delta function, $\delta(\mathbf{x})$, needs to be discretised, giving rise to a so-called discrete delta function, $\delta_h(\mathbf{x})$. The discrete delta function is finite and not identically null in a small vicinity of $\mathbf{x} = \mathbf{0}$, being equal to zero elsewhere. Note that centring a Dirac (or discrete) delta function in any point $\mathbf{x}_0 \in \mathbb{R}^\lambda$ is achieved by writing $\delta(\mathbf{x} - \mathbf{x}_0)$. The discrete delta function is constructed by means of a product of real functions $\varphi(r)$, with $r \in \mathbb{R}$ being a non-dimensional quantity (a ratio of distance values), as presented in equation 2.18.

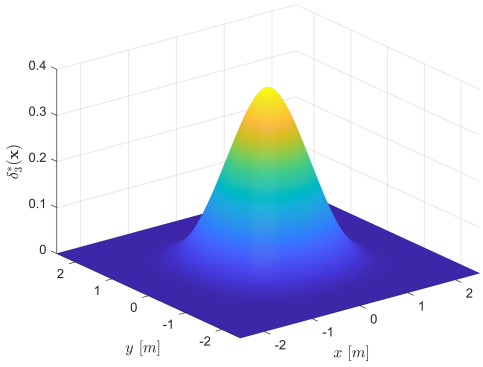
$$\delta_h(\mathbf{x}) = \frac{1}{h^\lambda} \prod_{i=1}^{\lambda} \varphi(r_i) \quad , \quad r_i = \frac{x_i}{h} \tag{2.18}$$

Naturally, for the properties of the Dirac delta function to remain valid in its discrete version, function $\varphi(r)$ should obey to a given number of rules, summarised in appendix A.3. Even so, there are several degrees of freedom that need to be set when deriving a function $\varphi(r)$. One important degree of freedom is the size of the vicinity wherein function $\varphi(r)$ is not identically null, known as the support width, w. Saying that a function $\varphi(r)$ has a support width w is equivalent to say that it is not identically null in the interval $-w < r < w$. Since $r_i = x_i/h$ along each direction i, a support width w translates into a not identically null region of characteristic dimension wh for the discrete delta function (an interval, circle or sphere, depending on weather it is one-, two- or three-dimensional, respectively).

Throughout this work, four discrete delta functions were considered: δ_3^*, δ_{2002}, δ_{2103}, δ_{4206}, with the first being the one used by Eldredge in article [1]. The indices of these discrete delta functions have to do with the procedure followed in their deduction and with the values that were assumed for the degrees of freedom. These explanations are provided in appendix A.3, as well as the extensive expressions of their corresponding $\varphi(r)$ functions. As an example, figure 2.3 represents the discrete delta function δ_3^*, both in one and two dimensions. The representation of all the others may be found in appendix A.3.



(a) One-dimensional version of δ_3^*.



(b) Two-dimensional version of δ_3^*.

Figure 2.3: Representation of discrete delta function δ_3^*, in one and two dimensions, taking $h = 1$.

Immersion and Restriction Operators

The immersion and restriction operations appear in both finite differences and finite volume strategies. In that regard, their discretisation is applicable for both approaches. Taking into account equation 2.5, the immersion operation $\delta(\chi)$ is simply discretised by approximating the respective integral with second-order accuracy, as desired in this work. This approximation is performed in equation 2.19, for a general function $F(\boldsymbol{\zeta})$ exclusively defined at the interface Γ, with $\boldsymbol{\zeta}$ being the surface coordinate(s) on the interface. Recall that the interface is assumed to be discretised into p equally-spaced solid points, at a distance δs from each other. Moreover, interface points are described by $\mathbf{x} = \mathbf{X}(\zeta)$. Throughout this work, the symbol $\overset{O2}{\sim}$ will be used to represent approximations with second-order accuracy.

$$F(\boldsymbol{\zeta})\delta\big(\chi(\mathbf{x})\big) = \int_{\Gamma} F(\boldsymbol{\zeta})\delta\big(\mathbf{x} - \mathbf{X}(\boldsymbol{\zeta})\big)dS(\boldsymbol{\zeta}) \overset{O2}{\sim} \sum_{k=1}^{p} F\big(\boldsymbol{\zeta}_k\big)\delta\big(\mathbf{x} - \mathbf{X}(\boldsymbol{\zeta}_k)\big)\delta s_k \tag{2.19}$$

Equation 2.19 was kept general, allowing different surface elements around each solid point δs_k. On the other hand, reminding equation 2.6, the restriction operation $\delta^T(\chi)$ of a general function $g(\mathbf{x})$ defined in $\mathbb{R}^\lambda$ to a given solid point $\mathbf{X}(\boldsymbol{\zeta})$ is approximated with second-order accuracy in equation 2.20.

$$g(\mathbf{x})\delta^T\big(\chi(\mathbf{x})\big) \equiv g(\mathbf{X}(\boldsymbol{\zeta})) = \int_{\mathbb{R}^\lambda} g(\mathbf{x})\delta\big(\mathbf{X}(\zeta) - \mathbf{x}\big)d\mathbf{x} \overset{O2}{\sim} \sum_{i=1}^{N} g\big(\mathbf{x}_i\big)\delta\big(\mathbf{x}_i - \mathbf{X}(\boldsymbol{\zeta})\big)h^\lambda \tag{2.20}$$

Recovering system 2.17, it is clear that the immersion operation will be included in matrix $\mathbf{M_B}$, whereas the restriction operation will be included in matrix $\mathbf{M_C}$. In the finite differences approach, the function defined at the interface that is being immersed into the computational domain is the normal derivative jump at each solid point, σ. Applying the result of equation 2.19, one arrives to equation 2.21.

$$-\mathbf{M_B}\boldsymbol{\sigma} = \begin{pmatrix} \sigma\delta(\chi(\mathbf{x}_1)) \\ \sigma\delta(\chi(\mathbf{x}_2)) \\ \vdots \\ \sigma\delta(\chi(\mathbf{x}_N)) \end{pmatrix} = \begin{pmatrix} \delta(\mathbf{x}_1 - \mathbf{X}(\boldsymbol{\zeta_1})) & \delta(\mathbf{x}_1 - \mathbf{X}(\boldsymbol{\zeta_2})) & \dots & \delta(\mathbf{x}_1 - \mathbf{X}(\boldsymbol{\zeta_p})) \\ \delta(\mathbf{x}_2 - \mathbf{X}(\boldsymbol{\zeta_1})) & \delta(\mathbf{x}_2 - \mathbf{X}(\boldsymbol{\zeta_2})) & \dots & \delta(\mathbf{x}_2 - \mathbf{X}(\boldsymbol{\zeta_p})) \\ \vdots & \vdots & \ddots & \vdots \\ \delta(\mathbf{x}_N - \mathbf{X}(\boldsymbol{\zeta_1})) & \delta(\mathbf{x}_N - \mathbf{X}(\boldsymbol{\zeta_2})) & \dots & \delta(\mathbf{x}_N - \mathbf{X}(\boldsymbol{\zeta_p})) \end{pmatrix} \begin{pmatrix} \sigma_1\delta s_1 \\ \sigma_2\delta s_2 \\ \vdots \\ \sigma_p\delta s_p \end{pmatrix} \tag{2.21}$$

Equation 2.21 reveals the existence of an important matrix with dimensions $N \times p$ relating all the computational points with all the solid points, to perform the immersion operation. Keeping the nomenclature of Eldredge in article [1], this matrix is called the regularisation matrix, $\mathbf{Reg}$. Using the element-by-element multiplication, $\circ$, one obtains equation 2.22. Continuing consistent with bold lower case letters to represent vectors and bold capital letters to represent matrices, the arising vector of surface elements around each solid point is represented by $\{\boldsymbol{\delta s}\}_{p\times 1}$, whereas the matrix is represented by $[\boldsymbol{\Delta S}]_{N\times p}$.

$$-\mathbf{M_B}\boldsymbol{\sigma} = \big[\mathbf{Reg}\big]_{N\times p}\left[\begin{pmatrix}\delta s_1\\ \delta s_2\\ \vdots\\ \delta s_p\end{pmatrix}\circ\begin{pmatrix}\sigma_1\\ \sigma_2\\ \vdots\\ \sigma_p\end{pmatrix}\right] = \left[\big[\mathbf{Reg}\big]_{N\times p}\circ\begin{pmatrix}\delta s_1 & \delta s_2 & \dots & \delta s_p\\ \delta s_1 & \delta s_2 & \dots & \delta s_p\\ \vdots & \vdots & \ddots & \vdots\\ \delta s_1 & \delta s_2 & \dots & \delta s_p\end{pmatrix}_{N\times p}\right]\begin{pmatrix}\sigma_1\\ \sigma_2\\ \vdots\\ \sigma_p\end{pmatrix} \tag{2.22}$$

Notice that performing the immersion operation may be interpreted as multiplying the regularisation matrix on the left of a vector corresponding to an element-by-element multiplication between vector $\boldsymbol{\delta s}$ and the vector with the values one is aiming to immerse. In order to construct the discrete system of equations for the finite differences approach, one may compactly write that $\mathbf{M_B} = -(\mathbf{Reg}\circ\boldsymbol{\Delta S})$.

Concerning the restriction operation of field $\bar{\phi}$ to the interface, when performing this operation at once for all the solid points, under the finite differences approach, one obtains equation 2.23.

$$\mathbf{M_C}\bar{\boldsymbol{\phi}} = \begin{pmatrix}\bar{\phi}\big(\mathbf{X}(\boldsymbol{\zeta_1})\big)\\ \bar{\phi}\big(\mathbf{X}(\boldsymbol{\zeta_2})\big)\\ \vdots\\ \bar{\phi}\big(\mathbf{X}(\boldsymbol{\zeta_p})\big)\end{pmatrix} = h^{\lambda}\begin{pmatrix}\delta(\mathbf{x}_1-\mathbf{X}(\boldsymbol{\zeta_1})) & \delta(\mathbf{x}_2-\mathbf{X}(\boldsymbol{\zeta_1})) & \dots & \delta(\mathbf{x}_N-\mathbf{X}(\boldsymbol{\zeta_1}))\\ \delta(\mathbf{x}_1-\mathbf{X}(\boldsymbol{\zeta_2})) & \delta(\mathbf{x}_2-\mathbf{X}(\boldsymbol{\zeta_2})) & \dots & \delta(\mathbf{x}_N-\mathbf{X}(\boldsymbol{\zeta_2}))\\ \vdots & \ddots & \vdots & \\ \delta(\mathbf{x}_1-\mathbf{X}(\boldsymbol{\zeta_p})) & \delta(\mathbf{x}_2-\mathbf{X}(\boldsymbol{\zeta_p})) & \dots & \delta(\mathbf{x}_N-\mathbf{X}(\boldsymbol{\zeta_p}))\end{pmatrix}\begin{pmatrix}\bar{\phi}_1\\ \bar{\phi}_2\\ \vdots\\ \bar{\phi}_N\end{pmatrix} \tag{2.23}$$

In equation 2.23, a matrix containing delta functions relating all the solid points with all the computational points arises. Note that this matrix is the transpose of the regularisation matrix. Therefore, performing the restriction operation corresponds to multiply h^{λ} and the transpose of the regularisation matrix in the left of the vector containing the computational field that one is aiming to restrict. Thus, the so-called interpolation matrix that performs the restriction operation, here represented by $\mathbf{Int}$, is simply given by $\mathbf{Int} = h^{\lambda}\mathbf{Reg}^T$. In finite differences, one has the relation $\mathbf{M_C} = \mathbf{Int}$.

Concerning the finite volume implementation strategy, by analysis of the system of equations 2.16, one has the following expressions: $\mathbf{M_B} = -h^{\lambda}(\mathbf{Reg}\circ\boldsymbol{\Delta S})$ and $\mathbf{M_C} = h^{\lambda}\mathbf{Int}$.

Laplacian Operator

The Laplacian operator, appearing only in the finite differences strategy, is equal to the sum of all partial second derivatives. As such, when discretised with second-order accuracy at a given computational cell with centroid C, the one-dimensional Laplacian is approximated according to equation 2.24. When one is dealing with a two-dimensional problem, the Laplacian is approximated according to equation 2.25. Subscripts E, W, N, S stand for the centroids of the east, west, north and south neighbouring cells. Notice that all the second derivatives are approximated using second-order central schemes.

$$\left(\nabla^2\bar{\phi}\right)_C^{\mathbf{1D}} = \left.\frac{\partial^2\bar{\phi}}{\partial x^2}\right|_C \overset{O2}{\sim} \frac{\bar{\phi}_E + \bar{\phi}_W - 2\bar{\phi}_C}{h^2} \tag{2.24}$$

$$\left(\nabla^2\bar{\phi}\right)_C^{\mathbf{2D}} = \left.\frac{\partial^2\bar{\phi}}{\partial x^2}\right|_C + \left.\frac{\partial^2\bar{\phi}}{\partial y^2}\right|_C \overset{O2}{\sim} \frac{\bar{\phi}_E + \bar{\phi}_W + \bar{\phi}_N + \bar{\phi}_S - 4\bar{\phi}_C}{h^2} \tag{2.25}$$

In finite differences, these expressions are used to fill matrix $\mathbf{M_A}$. Naturally, these equations only apply to interior cells, that is, cells that have neighbours in all directions of the λ-dimensional space.

In the finite differences strategy, the centroid of non-interior cells coincides with the boundary of the computational domain. Therefore, the Poisson equation for these kind of cells is simply set as $\bar{\phi}_C = \bar{\phi}_{BC}$, with $\bar{\phi}_{BC}$ being the known field value at that location of the domain boundary. This corresponds to impose a Dirichlet boundary condition. The lines of $\mathbf{M_A}$ describing these kind of cells receive exclusively a unitary value in the diagonal element and vector $\mathbf{v_a}$ receives $\bar{\phi}_{BC}$. Note that the corresponding line of matrix $\mathbf{M_B}$ should be already null, since the computational domain is supposed to be sufficiently large, so that the interface influence is not felt at the domain boundaries.

Gradient Operator

The gradient operator only appears in the finite volume implementation, being evaluated at cell faces. Remind that the gradient is a vector whose component along direction i is equal to the partial first derivative along that direction. Its definition is presented in equation 2.26 for a λ-dimensional space.

$$\nabla\bar{\phi} = \sum_{i=1}^{\lambda} \frac{\partial\bar{\phi}}{\partial x_i}\mathbf{e}_i \tag{2.26}$$

Notice, as well, that the gradient operator in the finite volume approach always appears in a dot product with the face external unit normal. So, the relevant quantity to discretise at each cell face is $\left(\nabla\bar{\phi}\right)_f \cdot \mathbf{n}_f$, that corresponds to a directional first derivative. Since the grid is Cartesian with face normal vectors aligned with one of the axes, this directional derivative ends up corresponding to a derivative along one of the coordinate axes. By computing the directional derivative with second-order accuracy using a central scheme, one arrives to equation 2.27, valid for both the one- and two-dimensional cases. The only difference is that, in the one-dimensional case, both indices f and F will be simultaneously particularised into east and west; whereas, in the two-dimensional case, north and south will also arise.

$$\left(\nabla\bar{\phi}\right)_f \cdot \mathbf{n}_f = \left.\frac{\partial\bar{\phi}}{\partial n_f}\right|_f \overset{O2}{\sim} \frac{\bar{\phi}_F - \bar{\phi}_C}{h} \tag{2.27}$$

In finite volume, applying equation 2.27 to all the faces of each cell and summing up all the terms leads to the coefficients that will incorporate matrix $\mathbf{M_A}$. Equation 2.27 is only applicable to interior cells.

Contrary to what happens in the finite differences strategy, the centroid of non-interior cells does not coincide with the domain boundary. So, in a finite volume approach, the known field values at the domain boundary have to be imposed on cell faces. This was achieved using a non-centred second-order scheme to express the directional derivatives for which expression 2.27 is not applicable.

As an example, consider a cell located at the easternmost boundary of the domain. The evaluation of term $\left(\nabla\bar{\phi}\right)_e \cdot \mathbf{n}_e$ needs to be adapted, as there is no cell centroid to the east, E, to apply expression 2.27. In that regard, the second-order non-centred scheme of equation 2.28 is used, depending on the centroid of the western neighbour, the cell centroid of the non-interior cell under study and the value on its east face (in which the boundary condition can be substituted $\bar{\phi}_e = \bar{\phi}_{BC}$). This non-centred formula relies on three points and, as such, it is second-order accurate to compute a first derivative. The deduction of expression 2.28 and respective order of accuracy is performed in appendix B.2. Since the term $(\frac{8}{3})\frac{\bar{\phi}_{BC}}{h}$ is known, it moves to the right side of the discretised equation, being included in vector $\mathbf{v_a}$.

$$\left(\nabla\bar{\phi}\right)_e \cdot \mathbf{n}_e = \left.\frac{\partial\bar{\phi}}{\partial n_e}\right|_e \overset{O2}{\sim} \frac{(1/3)\bar{\phi}_W + (-3)\bar{\phi}_C + (8/3)\bar{\phi}_e}{h} \tag{2.28}$$

For all non-interior cells, a similar procedure is followed, approximating the directional derivative for which equation 2.27 does not stand by an expression like 2.28. The expressions for the other faces appearing in the one- and/or two-dimensional problems are presented in equations 2.29, 2.30, 2.31.

$$\left(\nabla\bar{\phi}\right)_w \cdot \mathbf{n}_w = \left.\frac{\partial\bar{\phi}}{\partial n_w}\right|_w \overset{O2}{\sim} \frac{(1/3)\bar{\phi}_E + (-3)\bar{\phi}_C + (8/3)\bar{\phi}_w}{h} \tag{2.29}$$

$$\left(\nabla\bar{\phi}\right)_n \cdot \mathbf{n}_n = \left.\frac{\partial\bar{\phi}}{\partial n_n}\right|_n \overset{O2}{\sim} \frac{(1/3)\bar{\phi}_S + (-3)\bar{\phi}_C + (8/3)\bar{\phi}_n}{h} \tag{2.30}$$

$$\left(\nabla\bar{\phi}\right)_s \cdot \mathbf{n}_s = \left.\frac{\partial\bar{\phi}}{\partial n_s}\right|_s \overset{O2}{\sim} \frac{(1/3)\bar{\phi}_N + (-3)\bar{\phi}_C + (8/3)\bar{\phi}_s}{h} \tag{2.31}$$

In two dimensions, when dealing with a non-interior cell located at a corner of the computational domain, two non-centred schemes must be picked up from equations 2.28 to 2.31 and two additional terms of the form $(\frac{8}{3})\frac{\bar{\phi}_{BC}}{h}$ (that may be different from each other) migrate to the right-had side vector $\mathbf{v_a}$.

Divergence Operator

The divergence operator appears in both finite differences and finite volume approaches, in the right-hand side of the Poisson equation, applied to a term relying on an immersion operation. Since the immersion operation was already treated, consider the vector on which the divergence acts as a general vector $\mathbf{u}$, defined in $\mathbb{R}^\lambda$. The definition of the divergence operator is reminded in equation 2.32.

$$\nabla \cdot \mathbf{u} = \sum_{i=1}^{\lambda} \frac{\partial u_i}{\partial x_i} \tag{2.32}$$

The second-order discretisation of the divergence operator is presented in equations 2.33 and 2.34, for the one-dimensional and two-dimensional cases, respectively, using central schemes.

$$\left(\nabla \cdot \mathbf{u}\right)_C^{\mathbf{1D}} = \left.\frac{\partial u_x}{\partial x}\right|_C \overset{O2}{\sim} \frac{(u_x)_E - (u_x)_W}{2h} \tag{2.33}$$

$$\left(\nabla \cdot \mathbf{u}\right)_C^{\mathbf{2D}} = \left.\frac{\partial u_x}{\partial x}\right|_C + \left.\frac{\partial u_y}{\partial y}\right|_C \overset{O2}{\sim} \frac{(u_x)_E - (u_x)_W}{2h} + \frac{(u_y)_N - (u_y)_S}{2h} \tag{2.34}$$

Numerical Implementation of the Heaviside Function

When the Heaviside function and the one-dimensional Dirac delta functions were introduced, it was referred that both appeared applied to the indicator function. However, using equations 2.5 and 2.6, both restriction $\delta(\chi)$ and immersion $\delta^T(\chi)$ were transformed into integrals depending on a multidimensional Dirac delta function, that receives a position vector as argument. So, in fact, the indicator function is not being used in the numerical implementation of neither the immersion nor the restriction operations.

For complicated geometries, like a solid body with a very irregular shape, the definition of an indicator function may be a complex task. As such, if one was also able to compute the Heaviside function without using the indicator function, it would remain a mathematical variable used to perform the deduction of the method, whose numerical computation is, indeed, never needed.

As shown by Eldredge in article [1], it is possible to compute the Heaviside function fields, $H(\chi)$ and $H(-\chi)$, for all the computational points, without the need of explicitly having the indicator function. That procedure corresponds to solving equation 2.35 for $H(-\chi)$, and then computing $H(\chi) = 1 - H(-\chi)$. Equation 2.35 was simply obtained by taking the divergence of equation 2.7.

$$\nabla^2 H(-\chi) = -\nabla \cdot (\delta(\chi)\mathfrak{n}) \tag{2.35}$$

Notice that the discretisation of the Laplacian, divergence and immersion operators was already treated. As such, equation 2.35 is simply solved with second-order accuracy by applying those discretisation results and imposing null Dirichlet boundary conditions at the computational domain boundary.

Thus, one has two alternatives to set the Heaviside function: by explicitly computing the indicator χ and applying the Heaviside function definition of equation 2.2; or by solving system of equations 2.35. The first procedure presents a second-order error arising from the discretisation of operators, whereas the second is exact when χ is exactly defined. However, in complex geometries for which the interface is irregular, the computation of χ may have to be iterative (in order to find point $\mathbf{x_{sp}}$ in equation 2.1) and that process will also present an associated error. In this work, the Heaviside function fields, $H(\chi)$ and $H(-\chi)$, are computed as suggested by Eldredge [1], applying equation 2.35.

Chapter 3

1D Improvement Strategies

The first improvement strategies to be derived, implemented and tested were designed for the one-dimensional case, due to its inherent simplicity when compared to higher-dimensional scenarios. Only after the one-dimensional tests had been successfully completed, one advanced to two dimensions.

This chapter starts by geometrically describing the one-dimensional problem, in subchapter 3.1. Then, two alternative equations to close the system coming from the Immersed Layers Method are presented. In subchapter 3.2, an equation to express the normal derivative jump across the interface is derived, being presented two different strategies. In subchapter 3.3, an equation that imposes field values at the interface is studied, with two distinct strategies being given as well. Finally, in subchapter 3.4, a correction method to be run in the end of the execution is presented, that replaces the values near the interface, affected by the discrete delta functions, being provided three correction subvariants.

3.1 Problem Description

The one-dimensional problem considered in this work consists of a solid body surrounded by fluid from both sides, as presented in figure 3.1. The computational domain $\Omega = \{x \in \mathbb{R} : -\frac{L}{2} \leq x \leq \frac{L}{2}\}$ is divided into n identical cells, with the body occupying the region $x_{sp_1} \leq x \leq x_{sp_2}$. Figure 3.1 presents the domain discretisation in a finite volume strategy, with domain boundaries coinciding with cell faces.

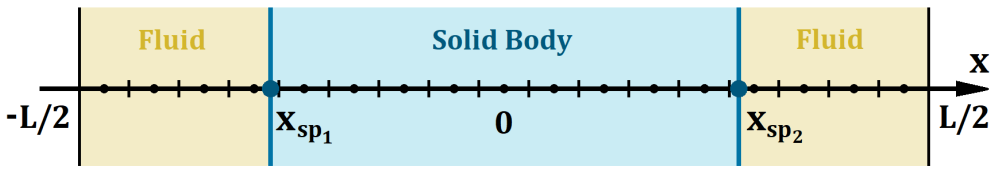


Figure 3.1: One-dimensional problem in hands, approached by a finite volume reasoning.

As suggested, depending on the number of cells in which the computational domain is divided, the location of the solid-fluid interfaces (fixed for a given geometry) may fall in a cell centroid, in a cell face or, in general, in an arbitrary location within the cell they pass through. So, in fact, one is dealing with an immersed boundary problem, wherein solid boundaries do not coincide, in general, with cell faces.

3.2 Equation for the Normal Derivative Jump

The first one-dimensional strategy that was tested consists in closing the system of equations by means of a finite differences equation to express the normal derivative jump across the interface or, in other words, the strength of the single-layer potential, σ. Contrary to what happens with the strength of the double-layer potential (assumed to be known), the single-layer potential strength is treated as an unknown. As such, it is immediate to think about an additional equation expressing this quantity.

In the one-dimensional case, the expression for the normal derivative jump simplifies to the difference between the derivative in the right and the derivative in the left of the solid point, as presented in equation 3.1. Now, the challenge lies in traducing these derivatives in terms of neighbouring grid points.

$$\sigma = (\nabla\phi^{+}_{sp} - \nabla\phi^{-}_{sp}) \cdot \mathfrak{n} = \frac{\partial\bar{\phi}}{\partial x}\bigg|^{r}_{sp} - \frac{\partial\bar{\phi}}{\partial x}\bigg|^{l}_{sp} \tag{3.1}$$

For discontinuous fields across the interface, the use of discrete delta functions in the first equation of the system, that comes from the Immersed Layers Method, implies that the numerical solution will always present a smeared interface. Thus, the grid points in the vicinity of the interface that are contained inside the support width of the discrete delta function are not representative of the slope tendency coming from each subdomain. As such, these points should not be used when traducing the derivatives in equation 3.1 into finite differences. The smearing phenomenon is qualitatively illustrated in figure 3.2, in which it is visible a numerical field (in orange) unable to present the expected sharp interface (in yellow). The region affected by the smearing appears in shades of grey, taking a support width $w = 2$ as an example.

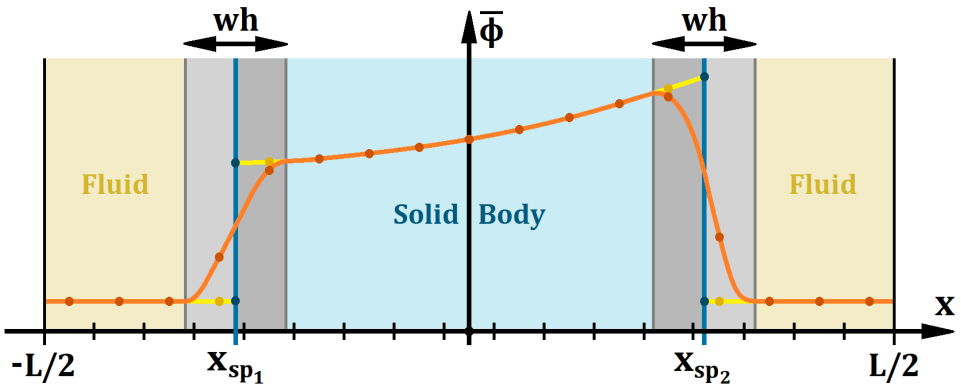


Figure 3.2: Illustration of the smearing phenomenon near the interface. The expected behaviour appears in yellow and the smeared one in orange. The affected region appears in shades of grey ($w = 2$).

A simple way to overcome this problem, when traducing equation 3.1 into finite differences, is to select grid points that are not contained in the support width of the discrete delta function, that is, grid points from out of the region in shades of grey appearing in figure 3.2. So, besides the information coming from the solid point, the additional equation with which one will close the system should only invoke grid points from the so-called superfluous region, $\mathbb{S}$. In this work, the superfluous region is considered to encompass both the fluid and solid subregions that are not affected by the discrete delta functions,

$\mathbb{S}^+$ and $\mathbb{S}^-$ respectively, as expressed in equation 3.2. The transition region, in which the smearing phenomenon takes place, is the complement set of the superfluous region, $\mathbb{T} = \Omega \setminus \mathbb{S}$. Consequently, the domain corresponds to the union of sets $\Omega = \mathbb{S} \cup \mathbb{T}$.

$$\mathbb{S} = \mathbb{S}^+ \cup \mathbb{S}^- \text{ , with } \begin{cases} \mathbb{S}^- = \{x \in \mathbb{R} : x_{sp_1} + \frac{w}{2}h \leq x \leq x_{sp_2} - \frac{w}{2}h\} \\ \mathbb{S}^+ = \{x \in \mathbb{R} : -\frac{L}{2} \leq x \leq x_{sp_1} - \frac{w}{2}h \ \vee \ x_{sp_2} + \frac{w}{2}h \leq x \leq \frac{L}{2}\} \end{cases} \tag{3.2}$$

In sections 3.2.1 and 3.2.2, equation 3.1 for the normal derivative jump is traduced into finite differences by making use of one and two superfluous points from each subdomain, respectively.

3.2.1 Using One Superfluous Point from Each Subdomain

The simplest way to discretise equation 3.1 consists of using unilateral first-order finite differences to express each derivative. Defining the grid point immediately before the interface that is out of the support width by x_b, and the point immediately after the interface that is out of the support width by x_a, one reaches the expressions presented in equation 3.3. The terms ϕ^l_{sp} and ϕ^r_{sp} stand respectively for the values of the left and right subfields at the solid point, that are known. Note that the left/right notation is used to allow the derivation of a general expression applicable for both solid points, since the use of superscripts $+$ or $-$ depends on whether the solid point performs a fluid-solid or solid-fluid transition.

$$\sigma = \frac{\bar{\phi}_a - \phi^r_{sp}}{x_a - x_{sp}} - \frac{\phi^l_{sp} - \bar{\phi}_b}{x_{sp} - x_b} \tag{3.3}$$

Concerning the problem in hands, with a solid body surrounded by a fluid on both sides, the first solid point performs a fluid-solid transition, whereas the second performs a solid-fluid one. Thus, the expressions for the normal derivative jump at the two solid points are given in equations 3.4. In each expression, superscripts l and r were simply particularised in the respective subdomains, $+$ or $-$.

$$\sigma_1 = \frac{\bar{\phi}_{a_1} - \phi^-_{sp_1}}{x_{a_1} - x_{sp_1}} - \frac{\phi^+_{sp_1} - \bar{\phi}_{b_1}}{x_{sp_1} - x_{b_1}} \quad , \quad \sigma_2 = \frac{\bar{\phi}_{a_2} - \phi^+_{sp_2}}{x_{a_2} - x_{sp_2}} - \frac{\phi^-_{sp_2} - \bar{\phi}_{b_2}}{x_{sp_2} - x_{b_2}} \tag{3.4}$$

However, continuing with the general form presented initially in equation 3.3, one may write it now as shown in equation 3.5, wherein (for the sake of compactness) the coefficients multiplying $\bar{\phi}_a$, $\bar{\phi}_b$, ϕ^l_{sp}, and ϕ^r_{sp} were hidden in variables c_a, c_b, c^l_{sp}, and c^r_{sp}, respectively. As unknowns, one has $\bar{\phi}_a$, $\bar{\phi}_b$ and σ.

$$\sigma = c_a\bar{\phi}_a + c_b\bar{\phi}_b + c^r_{sp}\phi^r_{sp} + c^l_{sp}\phi^l_{sp} \tag{3.5}$$

Putting, as usual, unknown variables in the left-hand side of the equality, one reaches equation 3.6, where it is immediate to identify that coefficients c_a and c_b will incorporate matrix $\mathbf{M_C}$, coefficient $d = -1$ will incorporate matrix $\mathbf{M_D}$ and that the right-hand side vector $\mathbf{v_b}$ will receive the terms concerning the information about the solid point for which the single-layer potential strength is being computed.

$$c_a\bar{\phi}_a + c_b\bar{\phi}_b + d\sigma = -(c^r_{sp}\phi^r_{sp} + c^l_{sp}\phi^l_{sp}) \tag{3.6}$$

3.2.2 Using Two Superfluous Points from Each Subdomain

In this strategy, despite the solid point information, two superfluous points from each subdomain are used to express σ. Besides $\bar{\phi}_b$ and $\bar{\phi}_a$, the points respectively before and after themselves, $\bar{\phi}_{bb}$ and $\bar{\phi}_{aa}$, are used. As such, a polynomial expansion as in equation 3.7 is performed for each side of the interface. In all the polynomial expansions performed throughout this work, the coefficients are represented by C.

$$\begin{cases} \phi^r(x) = C_0^r + C_1^r(x - x_{sp}) + C_2^r(x - x_{sp})^2 \\ \phi^l(x) = C_0^l + C_1^l(x - x_{sp}) + C_2^l(x - x_{sp})^2 \end{cases} \tag{3.7}$$

As an example, consider the polynomial development to the right. Equation 3.8 presents the relation between the coefficients C_0^r, C_1^r and C_2^r and the values $\bar{\phi}_a$, $\bar{\phi}_{aa}$ and ϕ_{sp}^r, in a matrix form.

$$\begin{pmatrix} 1 & 0 & 0 \\ 1 & (x_a - x_{sp}) & (x_a - x_{sp})^2 \\ 1 & (x_{aa} - x_{sp}) & (x_{aa} - x_{sp})^2 \end{pmatrix} \begin{pmatrix} C_0^r \\ C_1^r \\ C_2^r \end{pmatrix} = \begin{pmatrix} \phi_{sp}^r \\ \bar{\phi}_a \\ \bar{\phi}_{aa} \end{pmatrix} \tag{3.8}$$

By inverting the square matrix in equation 3.8, the polynomial expansion coefficients C_0^r, C_1^r and C_2^r may be expressed in terms of the values $\bar{\phi}_a$, $\bar{\phi}_{aa}$ and ϕ_{sp}^r. This result is presented in equation 3.9, where the entries of the inverse matrix were generally designated (m_{ij}).

$$\begin{pmatrix} C_0^r \\ C_1^r \\ C_2^r \end{pmatrix} = \begin{pmatrix} m_{11}^r & m_{12}^r & m_{13}^r \\ m_{21}^r & m_{22}^r & m_{23}^r \\ m_{31}^r & m_{32}^r & m_{33}^r \end{pmatrix} \begin{pmatrix} \phi_{sp}^r \\ \bar{\phi}_a \\ \bar{\phi}_{aa} \end{pmatrix} \tag{3.9}$$

Performing an identical development to the left-hand side of the interface, one is able to express the normal derivative jump as function of $\bar{\phi}_{bb}$, $\bar{\phi}_b$, $\bar{\phi}_a$, $\bar{\phi}_{aa}$, ϕ_{sp}^l and ϕ_{sp}^r, as presented in equation 3.10.

$$\sigma = C_1^r - C_1^l = (m_{21}^r \phi_{sp}^r + m_{22}^r \bar{\phi}_a + m_{23}^r \bar{\phi}_{aa}) - (m_{21}^l \phi_{sp}^l + m_{22}^l \bar{\phi}_b + m_{23}^l \bar{\phi}_{bb}) \tag{3.10}$$

Rearranging equation 3.10 to the usual form, one arrives to equation 3.11, in which (for the sake of consistency) the previous notation with c and d coefficients was recovered. These coefficients are determined by comparison between equations 3.10 and 3.11, for example $c_a = m_{22}^r$ and $c_b = -m_{22}^l$. This arranged form allows an immediate filling of matrices $\mathbf{M_C}$, $\mathbf{M_D}$ and right-hand side vector $\mathbf{v_b}$.

$$c_a \bar{\phi}_a + c_b \bar{\phi}_b + c_{aa} \bar{\phi}_{aa} + c_{bb} \bar{\phi}_{bb} + d\sigma = -(c_{sp}^r \phi_{sp}^r + c_{sp}^l \phi_{sp}^l) \tag{3.11}$$

3.3 Equation Imposing the Interface Value

Another way to close the system of equations consists of imposing the field value at the interface, following the same reasoning that was tried in the original Immersed Layers Method. However, instead of using a discrete delta function to perform the interpolation, one uses polynomial interpolation. Moreover,

one separately states that the exterior subfield restricted to a certain solid point must give ϕ_{sp}^{+}, and the interior subfield restricted to that same solid point must give ϕ_{sp}^{-}. The value of each subfield, when evaluated at the solid point location, is only expressed in terms of superfluous grid points, to overcome the smearing effect. This strategy leads to the additional equation 3.12, where the linear combination of the superfluous grid points is represented by a general set of γ coefficients. The computation of these coefficients depends on the number of superfluous grid points considered in the linear combination.

$$\frac{1}{2}\Big[\big(\gamma_b\bar{\phi}_b + \gamma_{bb}\bar{\phi}_{bb} + \cdots\big) + \big(\gamma_a\bar{\phi}_a + \gamma_{aa}\bar{\phi}_{aa} + \cdots\big)\Big] = \frac{1}{2}(\phi_{sp}^{l} + \phi_{sp}^{r}) \tag{3.12}$$

In sections 3.3.1 and 3.3.2, the linear combinations in γ that appear in equation 3.12 are performed by considering two and three superfluous points from each subdomain, respectively.

3.3.1 Using Two Superfluous Points from Each Subdomain

In this strategy, the restriction of the each subfield to a solid point is done by using two superfluous points from each side of the interface. That is, linear polynomial expansions of equations 3.13 are made.

$$\begin{cases} \phi^{r}(x) = C_0^{r} + C_1^{r}(x - x_{sp}) \\ \phi^{l}(x) = C_0^{l} + C_1^{l}(x - x_{sp}) \end{cases} \tag{3.13}$$

Considering as an example the right-hand side derivation, the coefficients are expressed in terms of $\bar{\phi}_a$ and $\bar{\phi}_{aa}$ as presented in equation 3.14. Similarly, the left-hand side derivation will use $\bar{\phi}_b$ and $\bar{\phi}_{bb}$.

$$\begin{pmatrix} 1 & (x_a - x_{sp}) \\ 1 & (x_{aa} - x_{sp}) \end{pmatrix} \begin{pmatrix} C_0^{r} \\ C_1^{r} \end{pmatrix} = \begin{pmatrix} \bar{\phi}_a \\ \bar{\phi}_{aa} \end{pmatrix} \tag{3.14}$$

Inverting the square matrix in equation 3.14 and denoting its entries by $(m_{i,j})$, equation 3.15 results.

$$\begin{pmatrix} C_0^{r} \\ C_1^{r} \end{pmatrix} = \begin{pmatrix} m_{11}^{r} & m_{12}^{r} \\ m_{21}^{r} & m_{22}^{r} \end{pmatrix} \begin{pmatrix} \bar{\phi}_a \\ \bar{\phi}_{aa} \end{pmatrix} \tag{3.15}$$

Knowing the coefficients, one is able to express the interface value of each subfield in terms of superfluous points. Note that the value of each polynomial, when evaluated at the solid point, corresponds only to the first coefficient, that is $\phi^{l}(x_{sp}) = C_0^{l}$ and $\phi^{r}(x_{sp}) = C_0^{r}$. Thus, one arrives to equation 3.16.

$$\frac{1}{2}\big[(m_{11}^{l}\bar{\phi}_b + m_{12}^{l}\bar{\phi}_{bb}) + (m_{11}^{r}\bar{\phi}_a + m_{12}^{r}\bar{\phi}_{aa})\big] = \frac{1}{2}(\phi_{sp}^{l} + \phi_{sp}^{r}) \tag{3.16}$$

Finally, recovering the c coefficients notation, one arrives to equation 3.17, wherein the coefficients going to matrix $\mathbf{M_C}$ are immediate to identify by comparison with equation 3.16, as well as the right-hand side term going to vector $\mathbf{v_b}$. Matrix $\mathbf{M_D}$ receives no coefficients, since the unknown σ is not present.

$$c_b\bar{\phi}_b + c_{bb}\bar{\phi}_{bb} + c_a\bar{\phi}_a + c_{aa}\bar{\phi}_{aa} = \frac{1}{2}(\phi_{sp}^{l} + \phi_{sp}^{r}) \tag{3.17}$$

3.3.2 Using Three Superfluous Points from Each Subdomain

In this strategy, three superfluous points are used to express the value of each subfield at the interface. Thus, the implementation constitutes a generalisation of the previously presented one. Now, the polynomial expansion for each side of the solid points encompasses three terms, as in equations 3.18.

$$\begin{cases} \phi^r(x) = C_0^r + C_1^r(x - x_{sp}) + C_2^r(x - x_{sp})^2 \\ \phi^l(x) = C_0^l + C_1^l(x - x_{sp}) + C_2^l(x - x_{sp})^2 \end{cases} \tag{3.18}$$

The reasoning is similar to the case with two superfluous points, with the difference that now one has an additional coefficient; so, the matrices have one additional row and one additional column. Besides using $\bar{\phi}_a$ and $\bar{\phi}_{aa}$, the right-hand side development also uses $\bar{\phi}_{aaa}$ (grid point after $\bar{\phi}_{aa}$). The left-hand side uses $\bar{\phi}_b$, $\bar{\phi}_{bb}$ and also $\bar{\phi}_{bbb}$ (grid point before $\bar{\phi}_{bb}$). The right side case in exemplified in 3.19.

$$\begin{pmatrix} 1 & (x_a - x_{sp}) & (x_a - x_{sp})^2 \\ 1 & (x_{aa} - x_{sp}) & (x_{aa} - x_{sp})^2 \\ 1 & (x_{aaa} - x_{sp}) & (x_{aaa} - x_{sp})^2 \end{pmatrix} \begin{pmatrix} C_0^r \\ C_1^r \\ C_2^r \end{pmatrix} = \begin{pmatrix} \bar{\phi}_a \\ \bar{\phi}_{aa} \\ \bar{\phi}_{aaa} \end{pmatrix} \tag{3.19}$$

Denoting, as usual, the entries of the inverse of the square matrix in equation 3.19 by general $(m_{i,j})$ values, the coefficients of the right-hand side polynomial expansion are given by equation 3.20.

$$\begin{pmatrix} C_0^r \\ C_1^r \\ C_2^r \end{pmatrix} = \begin{pmatrix} m_{11}^r & m_{12}^r & m_{13}^r \\ m_{21}^r & m_{22}^r & m_{23}^r \\ m_{31}^r & m_{32}^r & m_{33}^r \end{pmatrix} \begin{pmatrix} \bar{\phi}_a \\ \bar{\phi}_{aa} \\ \bar{\phi}_{aaa} \end{pmatrix} \tag{3.20}$$

By performing an identical treatment to the left-hand side and evaluating each polynomial in the solid point they refer to, one arrives to equation 3.21. Note that, despite the additional coefficients, one continues to have $\phi^r(x_{sp}) = C_0^r$ and $\phi^l(x_{sp}) = C_0^l$, since the polynomial is being expressed in terms of local coordinates. So, when the polynomial is evaluated at x_{sp}, the higher-order terms vanish.

$$\frac{1}{2}\left[(m_{11}^l\bar{\phi}_b + m_{12}^l\bar{\phi}_{bb} + m_{12}^l\bar{\phi}_{bbb}) + (m_{11}^r\bar{\phi}_a + m_{12}^r\bar{\phi}_{aa} + m_{13}^r\bar{\phi}_{aaa})\right] = \frac{1}{2}(\phi_{sp}^l + \phi_{sp}^r) \tag{3.21}$$

In c coefficients notation, the final expression of this strategy has the appearance of equation 3.22.

$$c_b\bar{\phi}_b + c_{bb}\bar{\phi}_{bb} + c_{bbb}\bar{\phi}_{bbb} + c_a\bar{\phi}_a + c_{aa}\bar{\phi}_{aa} + c_{aaa}\bar{\phi}_{aaa} = \frac{1}{2}(\phi_{sp}^l + \phi_{sp}^r) \tag{3.22}$$

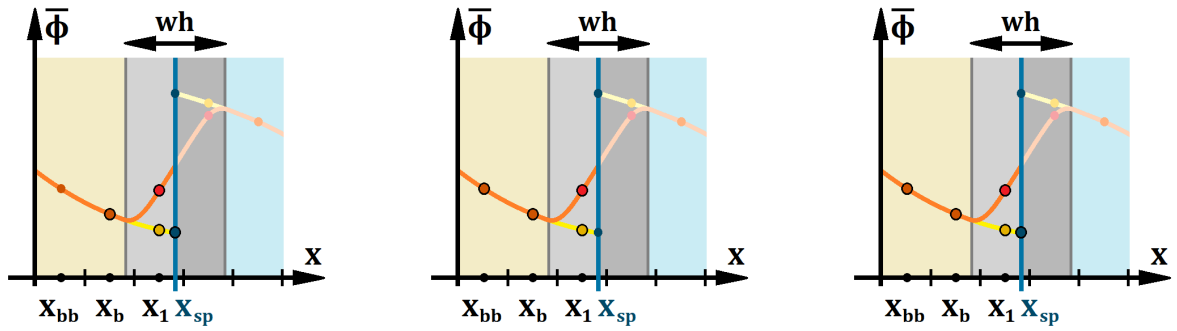
3.4 Smearing Correction

As previously seen in figure 3.2, the transition region is characterised by a locally smeared numerical solution. However, in the superfluous region, the solution is expected to present a proper behaviour, as the superfluous grid points do not directly feel the effect of the discrete delta functions. Therefore, it is plausible to implement a final corrective step, in which the values of the numerical solution inside the

transition region are discarded and replaced by estimates that take into account the superfluous region results, allowing to recover a sharp interface in the end of the numerical simulation.

As one will see in chapter 4 concerning one-dimensional results, the error decay is different when studying the superfluous region alone or the entire domain, since it includes the transition region affected by the discrete delta functions. Thus, this final correction of transition values allows, as well, to propagate a satisfactory superfluous decay to the transition region and, consequently, to the entire domain.

Three strategies were tested to correct the one-dimensional results: using one superfluous point together with the solid point (C1S); using two superfluous points (C2); and using two superfluous points together with the solid point (C2S). All these correction strategies are illustrated in figure 3.3.



(a) Correction C1S. (b) Correction C2. (c) Correction C2S.

Figure 3.3: Illustration of the three one-dimensional correction strategies implemented ($w = 2$).

For simplicity, in figure 3.3 only the left correction is drawn. The numerical values in the superfluous region are presented in orange. On the other hand, the numerical value inside the transition region (affected by smearing) is represented in red. The expected value inside the transition region appears represented in yellow and the values at the solid point appear represented in blue. The values implied in each correction strategy appear surrounded by a black circle. For example, strategy C1S (figure 3.3a) discards the red transition value and obtains the yellow expected one, by means of the solid point in blue and the closer superfluous point in orange. Figures 3.3b and 3.3c are interpreted in a similar way.

3.4.1 Using One Superfluous Point from Each Subdomain and the Solid Point

A simple way to correct values inside the transition region consists of using the closest superfluous point and the solid point information (strategy C1S in figure 3.3a). This correction consists of a linear interpolation between those values, being the corrected value in yellow approximated by equation 3.23.

$$\bar{\phi}_1^{new} = \left(\frac{x_{sp} - x_1}{x_{sp} - x_b}\right)\bar{\phi}_b + \left(\frac{x_1 - x_b}{x_{sp} - x_b}\right)\phi_{sp}^l \tag{3.23}$$

Keep in mind that figure 3.3 is a particular case, with only one point at each side of the interface needing correction. However, consider a general support width $w \in \mathbb{N}$ covering K grid points, k of which falling in the first half of the transition region, as shown in figure 3.4. Note that, depending of the relative location of the interface and on w being even or odd, one may have $K = 2k - 1$, $K = 2k$, or $K = 2k + 1$.

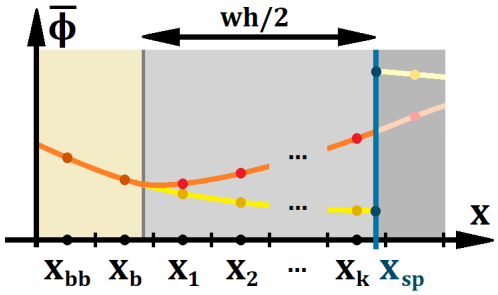


Figure 3.4: General first half of the transition region, with k points needing correction.

The procedure applied to derive equation 3.23 remains valid to correct all the points inside the first half of the transition region, being only important the relative distances between the point needing correction, the solid point ϕ^l_{sp}, and the superfluous point $\bar{\phi}_b$. When correcting the points of the second half of the transition region, the reasoning is similar, but now the superfluous point to be considered is $\bar{\phi}_a$ and the interest solid point value is ϕ^r_{sp}. If, by chance, a grid point coincides exactly with the interface, the mean between the left and right correction is assumed. The global expression to correct all the smeared values $\bar{\phi}_q$ (with $q \in \{1; ...; K\}$) around a given solid point located at x_{sp} is summarised in equation 3.24.

$$\bar{\phi}^{new}_q = \begin{cases} \left(\frac{x_{sp}-x_q}{x_{sp}-x_b}\right)\bar{\phi}_b + \left(\frac{x_q-x_b}{x_{sp}-x_b}\right)\phi^l_{sp} & \Leftarrow x_q < x_{sp} \\ \left(\frac{x_q-x_{sp}}{x_a-x_{sp}}\right)\bar{\phi}_a + \left(\frac{x_a-x_q}{x_a-x_{sp}}\right)\phi^r_{sp} & \Leftarrow x_q > x_{sp} \\ \frac{1}{2}(\phi^l_{sp} + \phi^r_{sp}) & \Leftarrow x_q = x_{sp} \end{cases} \tag{3.24}$$

3.4.2 Using Two Superfluous Points from Each Subdomain

Another way to correct transition values consists of pure extrapolation from the superfluous region without using the solid point information (strategy C2). Considering the example of figure 3.3b, the new value in yellow is simply approximated by linear extrapolation from the two points in orange, according to equation 3.25. The spacing between all the points involved is a known multiple of the grid size and, as such, obtaining expression 3.25 from the equation of a straight line is straightforward.

$$\bar{\phi}^{new}_1 = 2\bar{\phi}_b - \bar{\phi}_{bb} \tag{3.25}$$

Now, consider the general case of figure 3.4, with k points needing correction inside the first half of the transition region. By performing simple linear extrapolations from the two superfluous points to all the k points, one arrives to the expressions presented in the set of equations 3.26.

$$\bar{\phi}^{new}_1 = 2\bar{\phi}_b - \bar{\phi}_{bb} \quad , \quad \bar{\phi}^{new}_2 = 3\bar{\phi}_b - 2\bar{\phi}_{bb} \quad , \quad (\cdots) \quad , \quad \bar{\phi}^{new}_k = (k+1)\bar{\phi}_b - k\bar{\phi}_{bb} \tag{3.26}$$

Similarly, the values lying on the right half of the transition region are corrected by an extrapolation using the right-hand side superfluous region, that is, $\bar{\phi}_a$ and $\bar{\phi}_{aa}$. Again, if a grid point exactly coincides with the interface, its corrected value is taken as the mean between both left and right correction formulas. The expressions to be applied to each smeared point are summarised in equation 3.27.

$$\bar{\phi}_q^{new} = \begin{cases} \bar{\phi}_q^{new,l} = (q+1)\bar{\phi}_b - q\bar{\phi}_{bb} & \Leftarrow x_q < x_{sp} \\ \bar{\phi}_q^{new,r} = \big[(K-q+1)+1\big]\bar{\phi}_a - (K-q+1)\bar{\phi}_{aa} & \Leftarrow\ x_q > x_{sp} \\ \frac{1}{2}(\bar{\phi}_q^{new,l} + \bar{\phi}_q^{new,r}) & \Leftarrow x_q = x_{sp} \end{cases} \tag{3.27}$$

3.4.3 Using Two Superfluous Points from Each Subdomain and the Solid Point

The third strategy uses two superfluous points together with the solid point information. The previous strategies, C1S and C2, relied on information from two points and were equivalent to consider linear polynomials to perform the respective interpolation or extrapolation. However, current strategy C2S is based on information coming from three points, which translates into the use of quadratic polynomials. Thus, one should write one polynomial for each side of the interface, as presented in equation 3.28.

$$\begin{cases} \phi^l(x) = C_0^l + C_1^l x + C_2^l x^2 \\ \phi^r(x) = C_0^r + C_1^r x + C_2^r x^2 \end{cases} \tag{3.28}$$

Coefficients C_0^l, C_1^l, C_2^l are determined using $\bar{\phi}_b$, $\bar{\phi}_{bb}$, ϕ_{sp}^l, and coefficients C_0^r, C_1^r, C_2^r are determined by making use of $\bar{\phi}_a$, $\bar{\phi}_{aa}$, ϕ_{sp}^r. The procedure is exemplified in equation 3.29 for the left-hand side.

$$\begin{pmatrix} 1 & 0 & 0 \\ 1 & x_b & x_b^2 \\ 1 & x_{bb} & x_{bb}^2 \end{pmatrix} \begin{pmatrix} C_0^l \\ C_1^l \\ C_2^l \end{pmatrix} = \begin{pmatrix} \phi_{sp}^l \\ \bar{\phi}_b \\ \bar{\phi}_{bb} \end{pmatrix} \tag{3.29}$$

Inverting the square matrix in equation 3.29 and denoting its entries by (m_{ij}), one is able to express the coefficients in terms of the field values as it is shown in equation 3.30.

$$\begin{pmatrix} C_0^l \\ C_1^l \\ C_2^l \end{pmatrix} = \begin{pmatrix} m_{11}^l & m_{12}^l & m_{13}^l \\ m_{21}^l & m_{22}^l & m_{23}^l \\ m_{31}^l & m_{32}^l & m_{33}^l \end{pmatrix} \begin{pmatrix} \phi_{sp}^l \\ \bar{\phi}_b \\ \bar{\phi}_{bb} \end{pmatrix} \tag{3.30}$$

Proceeding in the same way to the right-hand side, coefficients C_0^r, C_1^r, C_2^r are discovered, as well. Therefore, for the transition region of figure 3.4 with a general number of K points needing correction, the new value for a given point x_q is simply obtained by applying equation 3.31.

$$\bar{\phi}_q^{new} = \begin{cases} \phi^l(x_q) & \Leftarrow x_q < x_{sp} \\ \phi^r(x_q) & \Leftarrow x_q > x_{sp} \\ \frac{1}{2}(\phi^l(x_q) + \phi^r(x_q)) & \Leftarrow x_q = x_{sp} \end{cases} \tag{3.31}$$

Note that this last strategy, by using three points to estimate a function value, is a third-order method. However, this correction step will not lead to third-order transition values, since it will act over second-order superfluous results. So, transition corrected values are limited to second-order accuracy as well.

Chapter 4

1D Results

In this chapter, the one-dimensional implementation of the Immersed Layers Method is performed, being compared the original formulation of chapter 2 and the several one-dimensional improvement strategies that were proposed in chapter 3. The one-dimensional results of the original formulation are presented for both finite differences and finite volume approaches. The finite differences test allows to confirm whether the numerical implementation is in line with the results presented by Eldredge in article [1], serving also as a mean to find out if the derived finite volume approach is behaving properly. Once this confirmation is done, all the results concerning the improvement strategies proceed using the finite volume approach in the masked Poisson equation.

In subchapter 4.1, general considerations are made, concerning the geometry and the presentation of the analytical functions that will be considered. Moreover, the error field is defined, alongside with relevant scalar error quantities, whose decay will be evaluated. In subchapter 4.2, the IL method is presented in its original formulation, with the system being closed with equation 2.11. In subchapter 4.3, the strategy of subchapter 3.2 is tested, concerning an additional equation expressing the normal derivative jump. In subchapter 4.4, one studies the strategy of subchapter 3.3, concerning an additional equation imposing the field value at the interface. In subchapter 4.5, all the 1D methods are compared. Finally, in subchapter 4.6 the 1D corrective procedures derived in subchapter 3.4 are tested, recovering interface sharpness and propagating superfluous decays to the transition region.

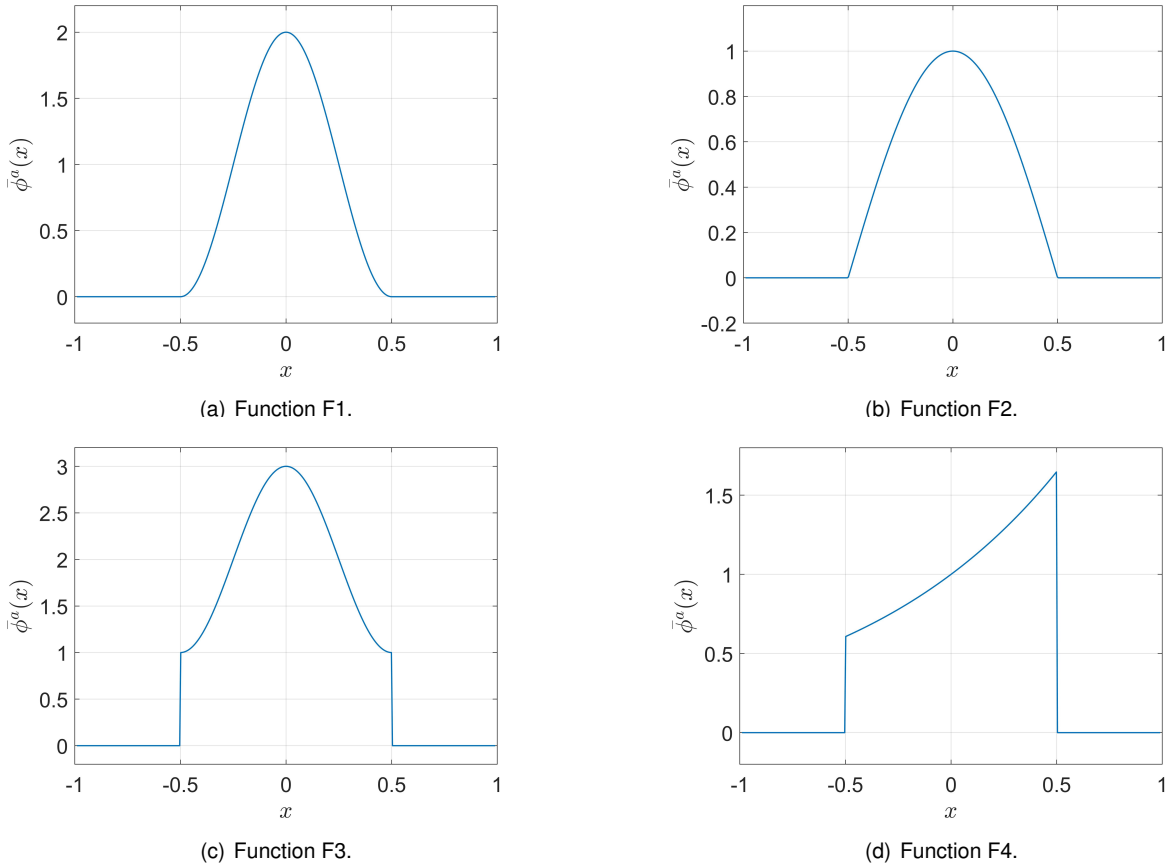
4.1 General Considerations

Firstly, the geometrical parameters left general in chapter 2 should be given numeric values. Recalling figure 3.1, the length of the computational domain was set as $L = 2$ and the solid points were placed at $x_{sp_1} = -0.5 = -x_{sp_2}$. Secondly, one must define analytical functions to perform the numerical tests, so that the source terms lying on the right-hand side vectors are computed and one knows exactly what solution should be expected to arise. In order to be valid, the derived improvement strategies must be able to deal with the most general field behaviours near the interface. So, the focus should be a worst case scenario function, with discontinuous value and discontinuous derivatives across the interface.

Consider four one-dimensional functions, named as F1, F2, F3 and F4. Let functions F1 and F2 be continuous through the interface, with F1 presenting a continuous first derivative across the interface, while F2 does not. Let F3 and F4 be discontinuous in value across the interface, with F3 presenting a continuous first derivative, whereas F4 presents discontinuous derivatives across the interface. This last function will be the focus of the numerical tests performed in this chapter. The analytical expressions considered for all these functions are summarised in table 4.1 and their graphs are shown in figure 4.1.

Table 4.1: Expression of each analytical subfield considered in functions F1, F2, F3, F4.

	Function F1	Function F2	Function F3	Function F4
$\phi^-(x)$	$\cos(2\pi\|x\|)+1$	$\cos(\pi x)$	$\cos(2\pi\|x\|)+2$	e^x
$\phi^+(x)$	0	0	0	0



(a) Function F1.

(b) Function F2.

(c) Function F3.

(d) Function F4.

Figure 4.1: Graphical representation of all the four one-dimensional functions considered.

Once an analytical expression is used to compute the source terms and the simulation generates a numerical solution, one is able to compute the error field, ε, taken as the difference between the numerical solution field, $\bar{\phi}$, and the analytical field, $\bar{\phi}^{\boldsymbol{a}}$, according to equation 4.1. Recall that all these entities are vectors with dimensions $N \times 1$, with $N = n$ in the one-dimensional case.

$$\boldsymbol{\varepsilon} = \bar{\boldsymbol{\phi}} - \bar{\boldsymbol{\phi}}^{\boldsymbol{a}} \tag{4.1}$$

To study an error field and to make comparisons between error fields arising from different implementations, three main scalar quantities are defined: the mean (also known as the L^1 norm), the L^2 norm and the maximum value (also known as the L^∞ norm), whose definitions are provided in equations 4.2a, 4.2b and 4.2c, respectively. These formulas may be used to compute the error quantities in a particular domain region. For example, consider the superfluous region containing N^{Sup} cells, characterised by a $N^{Sup} \times 1$ error vector. The mean value of the error in the superfluous region is computed with equation 4.2a, using ε^{Sup} instead of the global vector ε and using N^{Sup} instead of N.

$$\|\varepsilon\|_1 = \overline{\varepsilon} = \frac{1}{N}\sum_{i=1}^{N}|\varepsilon_i| \tag{4.2a}$$

$$\|\varepsilon\|_2 = \sqrt{\frac{1}{N}\sum_{i=1}^{N}|\varepsilon_i|^2} \tag{4.2b}$$

$$\|\varepsilon\|_\infty = \varepsilon_{max} = \max|\varepsilon_i| \tag{4.2c}$$

When a scalar error quantity is said to be asymptotically decaying, its plot as function of the grid size corresponds to a straight line, when represented in a system of logarithmic axes. The order of decay, ρ, is given by the slope of that straight line in the logarithmic graph. Considering two points within the asymptotic region, (h_1, ε_1) and (h_2, ε_2), the order of decay is computed according to equation 4.3.

$$\rho = \frac{\log(\varepsilon_2) - \log(\varepsilon_1)}{\log(h_2) - \log(h_1)} \tag{4.3}$$

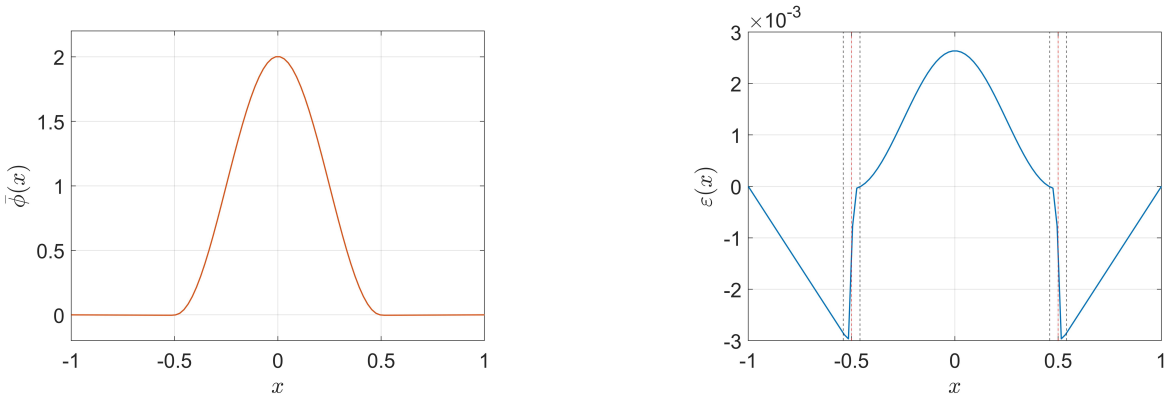
In the one-dimensional case, due to its inherent geometrical simplicity, it is important to make sure that the numerical tests address general relative locations of the solid points to its closest neighbouring cell centroids, so that no particular case is being unconsciously targeted. As such, in this chapter, the decays were evaluated by setting a parameter $\mu = \{0, 1, 2, 3\}$ and then running the code for consecutive numbers of cells in the set $n \in \{100 + \mu; 200 + \mu; 400 + \mu; 800 + \mu; 1600 + \mu\}$. The lower number of cells was defined taking into account that the error should already be presenting its asymptotic behaviour. The order of decay is computed by using the points corresponding to the two most refined grids.

Given the location of the solid points, in finite volume, when the parameter μ is equal to $\{0, 1, 2, 3\}$, the distance between the first solid point and its immediately previous grid point is given as a percentage of the grid size by $\{50\%; 75\%; 0\%; 25\%\}$ respectively, whereas the distance between the second solid point and its immediately previous grid point is $\{50\%; 25\%; 0\%; 75\%\}$. In finite differences, when μ is equal to $\{0, 1, 2, 3\}$, the same relative distances are $\{75\%; 0\%; 25\%; 50\%\}$ and $\{25\%; 0\%; 75\%; 50\%\}$ respectively.

When the relative distance is 0% (solid point coinciding with a centroid) or 50% (in the middle of two centroids), the results obtained with some discrete delta functions are considerably better. However, those locations are just particular cases and should not be taken as representative, since in immersed boundary methods the interface is assumed to fall anywhere relative to the grid. So, finite volume results should only be considered representative for $\mu \in \{1, 3\}$ and finite differences ones for $\mu \in \{0, 2\}$. In this chapter, finite volume results are presented for $\mu = 1$ and finite differences ones for $\mu = 0$.

4.2 Original Formulation

In this subchapter, the Immersed Layers Method is tested under a one-dimensional implementation, both in finite differences and finite volume. In the finite differences approach one considers the system of equations 2.12, whereas in the finite volume approach one considers system 2.16. In both systems, the one-dimensional discretisation of all the operators is taken and $\lambda = 1$. Starting with the finite differences implementation, the numerical solution and the error field when considering function F1 are displayed in figure 4.2, using δ_3^* and $n = 100$. The decays obtained for all the error quantities are shown in table 4.2.



(a) Numerical field.

(b) Error field.

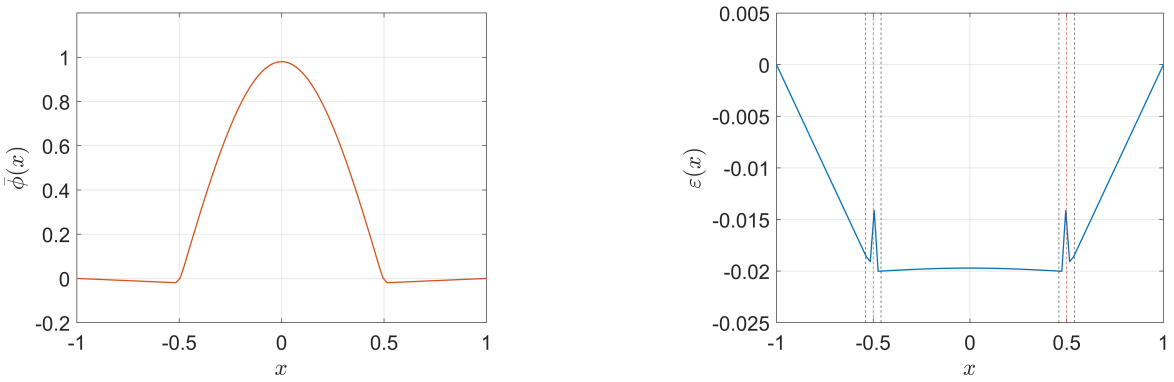
Figure 4.2: Results for function F1 under a finite differences approach, using δ_3^* and $n = 100$ cells.

Table 4.2: Error decay for function F1, in the original IL formulation, using finite differences ($\mu = 0$).

δ_h	$\bar{\varepsilon}$	$\|\varepsilon\|_2$	ε_{max}	$\bar{\varepsilon}^{Sup}$	$\|\varepsilon\|_2^{Sup}$	ε_{max}^{Sup}	$\bar{\varepsilon}^{Trans}$	$\|\varepsilon\|_2^{Trans}$	$\varepsilon_{max}^{Trans}$
δ_3^*	1.9965	1.9971	1.9892	1.9938	1.9925	1.9817	1.9953	1.9932	1.9892
δ_{2002}	1.9963	1.9979	1.9992	1.9983	1.9992	1.9992	1.9971	1.9956	1.9956
δ_{2103}	1.9968	1.9975	1.9968	1.9971	1.9958	1.9894	2.0007	1.9975	1.9968
δ_{4206}	2.0121	2.0018	2.0006	2.0228	2.0110	2.0006	2.0064	2.0015	2.0016

From figure 4.2, one concludes that the absolute values of the error field near the interface are comparable to the ones observed inside the solid body, due to the fact that function F1 is well behaved across the interface (continuous both in value and first derivative). Furthermore, according to table 4.2, the moment properties assumed during the derivation of the discrete delta functions are able to arise. As explained in appendix A.3, discrete delta functions δ_3^*, δ_{2002} and δ_{2103} were derived considering a moment order $m = 2$, whereas δ_{4206} presents a moment order $m = 4$. Thus, the good behaviour of the field across the interface allows δ_3^*, δ_{2002} and δ_{2103} to originate second-order decays in all the regions. However, δ_{4206} is incapable of manifesting its fourth-order potential, since all the operators appearing in the system of equations were discretised with second-order accuracy only.

When considering function F2 under finite differences, the numerical solution and error field are illustrated in figure 4.3 using δ_3^* and $n = 100$. The decays of all the error quantities appear in table 4.3.



(a) Numerical field.

(b) Error field.

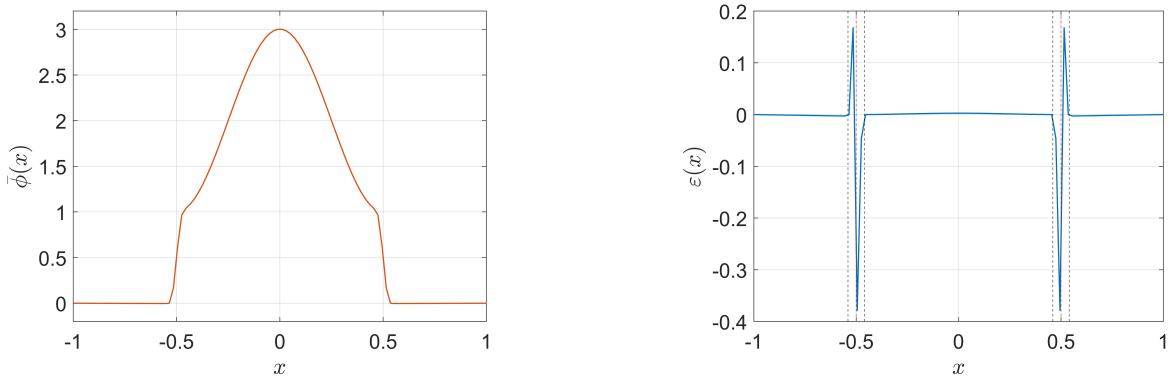
Figure 4.3: Results for function F2 under a finite differences approach, using δ_3^* and $n = 100$ cells.

Table 4.3: Error decay for function F2, in the original IL formulation, using finite differences ($\mu = 0$).

δ_h	$\bar{\varepsilon}$	$\|\varepsilon\|_2$	ε_{max}	$\bar{\varepsilon}^{Sup}$	$\|\varepsilon\|_2^{Sup}$	ε_{max}^{Sup}	$\bar{\varepsilon}^{Trans}$	$\|\varepsilon\|_2^{Trans}$	$\varepsilon_{max}^{Trans}$
δ_3^*	0.9977	0.9985	1.0022	0.9943	0.9963	1.0021	0.9973	0.9970	1.0022
δ_{2002}	0.9974	0.9981	1.0013	0.9950	0.9962	1.0012	0.9987	0.9987	1.0013
δ_{2103}	0.9978	0.9987	1.0024	0.9958	0.9975	1.0024	1.0004	1.0002	1.0024
δ_{4206}	0.9999	1.0013	1.0014	0.9902	0.9928	1.0011	0.9955	0.9957	1.0014

Once more, according to figure 4.3, the absolute values of the error field near the interface present a similar order of magnitude compared to the ones inside the solid body. So, the discontinuity in first derivative did not lead, in this case, to significant error peaks near the interface. However, according to the results of table 4.3, despite function F2 being continuous, the existence of a discontinuity in the first derivative across the interface was enough to prevent the properties of the discrete delta functions to manifest. All the error quantities presented a first-order decay in all the domain regions.

The numerical solution and error field for function F3, under finite differences, are presented in figure 4.4, using δ_3^* and $n = 100$ cells. The decays are shown in table 4.4, with “N.D.” standing for “No Decay”.



(a) Numerical field.

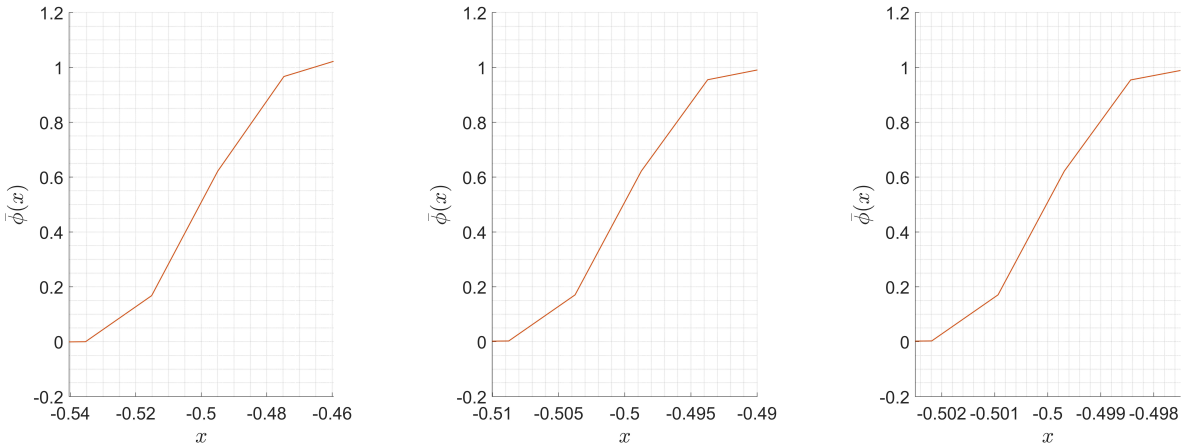
(b) Error field.

Figure 4.4: Results for function F3 under a finite differences approach, using δ_3^* and $n = 100$ cells.

Table 4.4: Error decay for function F3, in the original IL formulation, using finite differences ($\mu = 0$).

δ_h	$\bar{\varepsilon}$	$\|\varepsilon\|_2$	ε_{max}	$\bar{\varepsilon}^{Sup}$	$\|\varepsilon\|_2^{Sup}$	ε_{max}^{Sup}	$\bar{\varepsilon}^{Trans}$	$\|\varepsilon\|_2^{Trans}$	$\varepsilon_{max}^{Trans}$
δ_3^*	1.0177	0.4990	N.D.	1.9938	1.9925	1.9817	N.D.	N.D.	N.D.
δ_{2002}	1.0160	0.4990	N.D.	1.9983	1.9992	1.9992	N.D.	N.D.	N.D.
δ_{2103}	1.0189	0.4990	N.D.	1.9971	1.9958	1.9894	N.D.	N.D.	N.D.
δ_{4206}	1.0074	0.4990	N.D.	2.0228	2.0110	2.0006	N.D.	N.D.	N.D.

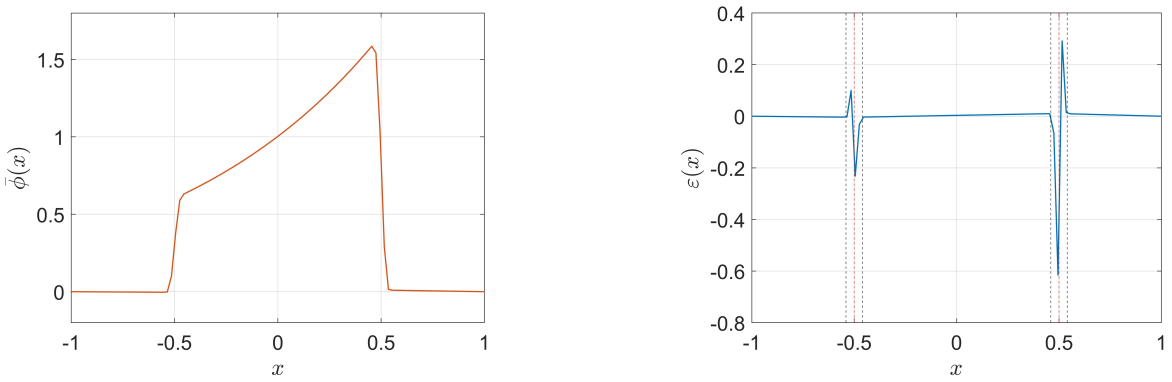
As one observes in figure 4.4, the introduction of a discontinuity in value across the interface led to significant error peaks in the transition region. Moreover, according to table 4.4, all the error quantities do not decay in the transition region. However, the continuity in derivative across the interface appears to be enough to hold a second-order decay in the superfluous region. Notice that, as one refines the grid, the discrete delta function always continues to act over its fixed characteristic length, wh, set at the time of its deduction. Thus, the transition region becomes thinner and thinner, but it will always look the same if one zooms in, performing a smeared transition between the left and the right values of the given discontinuity. This phenomenon, illustrated in figure 4.5, may be pointed as the main reason why the use of discrete delta functions is unable (by its own) to present an overall accuracy higher than first for discontinuous fields. As table 4.4 shows, the non-decay in the transition region makes the global mean error to present first-order accuracy. The L^2 norm reverts from a second-order superfluous decay to a global half-order behaviour. As a remark, article [15] reports decays of the L^2 norm half order below the ones verified for the mean error. Although the context is different (that article concerns a Stokes flow), it shows that half-order penalties in the L^2 norm decay are not an unprecedented phenomenon in immersed boundary formulations using discrete delta functions, when dealing with discontinuous fields.



(a) $n = 100$ (b) $n = 400$ (c) $n = 1600$

Figure 4.5: Transition region around the first solid point, $|x - x_{sp_1}| \leq (\frac{w}{2})h$, using δ_3^* for different grids.

The numerical solution and error field obtained with function F4 under finite differences, considering δ_3^* and $n = 100$ cells, are presented in figure 4.6. Table 4.5 displays the orders of decay that were observed for all the discrete delta functions in all the regions.



(a) Numerical field.

(b) Error field.

Figure 4.6: Results for function F4 under a finite differences approach, using δ_3^* and $n = 100$ cells.

Table 4.5: Error decay for function F4, in the original IL formulation, using finite differences ($\mu = 0$).

δ_h	$\bar{\varepsilon}$	$\|\varepsilon\|_2$	ε_{max}	$\bar{\varepsilon}^{Sup}$	$\|\varepsilon\|_2^{Sup}$	ε_{max}^{Sup}	$\bar{\varepsilon}^{Trans}$	$\|\varepsilon\|_2^{Trans}$	$\varepsilon_{max}^{Trans}$
δ_3^*	0.9933	0.4988	N.D.	0.9900	0.9909	0.9925	N.D.	N.D.	N.D.
δ_{2002}	0.9963	0.4984	N.D.	0.9945	0.9953	0.9961	N.D.	N.D.	N.D.
δ_{2103}	0.9941	0.4990	N.D.	0.9934	0.9941	0.9930	N.D.	N.D.	N.D.
δ_{4206}	0.9919	0.4988	N.D.	0.9808	0.9846	0.9861	N.D.	N.D.	N.D.

As in the previous case, due to the discontinuity in value, the error presents significant peaks near the interface and the error quantities do not decay in the transition region. Here the derivative across the interface is discontinuous as well, leading to first-order accuracy in the superfluous region. When the entire domain is considered, the mean error decays with first-order accuracy, whereas the L^2 norm presents a half-order decay. In article [1], Eldredge reported a first-order decay of the L^2 norm in the superfluous region for a two-dimensional field with discontinuity in value and normal derivative across the interface, when using δ_3^* in a finite differences approach. Note that it is exactly what one obtained in the cell of table 4.5 that is highlighted in blue. Even though the problem there is two-dimensional and here it is one-dimensional, one can say that the results seem to be in line with expectations. Further up in this work, in chapter 6, the exact same two-dimensional problem will be addressed.

The results of the one-dimensional finite volume implementation of the Immersed Layers Method in its original formulation are presented in tables 4.6 to 4.9, being identical to the finite differences ones. So, the derived finite volume approach behaves as expected. The first-order superfluous decay of the L^2 norm was again observed for function F4, in line with article [1]. This value is highlighted in table 4.9.

Table 4.6: Error decay for function F1, in the original IL formulation, using finite volume ($\mu = 1$).

δ_h	$\bar{\varepsilon}$	$\|\varepsilon\|_2$	ε_{max}	$\bar{\varepsilon}^{Sup}$	$\|\varepsilon\|_2^{Sup}$	ε_{max}^{Sup}	$\bar{\varepsilon}^{Trans}$	$\|\varepsilon\|_2^{Trans}$	$\varepsilon_{max}^{Trans}$
δ_3^*	2.0028	2.0037	2.0001.	1.9944	1.9945	1.9996	1.9997	2.0000	2.0001
δ_{2002}	1.9999	2.0002	2.0003	1.9994	2.0006	2.0003	1.9986	1.9986	1.9993
δ_{2103}	2.0026	2.0041	2.0002	1.9932	1.9928	1.9999	2.0000	2.0001	2.0002
δ_{4206}	2.0000	2.0006	2.0003	2.0005	2.0021	2.0003	1.9953	1.9952	1.9983

Table 4.7: Error decay for function F2, in the original IL formulation, using finite volume ($\mu = 1$).

δ_h	$\bar{\varepsilon}$	$\|\varepsilon\|_2$	ε_{max}	$\bar{\varepsilon}^{Sup}$	$\|\varepsilon\|_2^{Sup}$	ε_{max}^{Sup}	$\bar{\varepsilon}^{Trans}$	$\|\varepsilon\|_2^{Trans}$	$\varepsilon_{max}^{Trans}$
δ_3^*	0.9993	0.9996	1.0011	0.9976	0.9984	1.0011	0.9987	0.9985	1.0011
δ_{2002}	0.9992	0.9993	1.0007	0.9979	0.9984	1.0006	0.9993	0.9993	1.0007
δ_{2103}	0.9993	0.9996	1.0012	0.9974	0.9990	1.0012	1.0002	1.0001	1.0012
δ_{4206}	1.0004	1.0010	1.0007	0.9957	0.9968	1.0007	0.9978	0.9979	1.0007

Table 4.8: Error decay for function F3, in the original IL formulation, using finite volume ($\mu = 1$).

δ_h	$\bar{\varepsilon}$	$\|\varepsilon\|_2$	ε_{max}	$\bar{\varepsilon}^{Sup}$	$\|\varepsilon\|_2^{Sup}$	ε_{max}^{Sup}	$\bar{\varepsilon}^{Trans}$	$\|\varepsilon\|_2^{Trans}$	$\varepsilon_{max}^{Trans}$
δ_3^*	0.9188	0.50001	N.D.	1.9945	1.9944	1.9996	N.D.	N.D.	N.D.
δ_{2002}	1.0006	0.5000	N.D.	1.9994	2.0006	2.0002	N.D.	N.D.	N.D.
δ_{2103}	1.0016	0.5001	N.D.	1.9932	1.9938	1.9999	N.D.	N.D.	N.D.
δ_{4206}	1.0056	0.5000	N.D.	2.0005	2.0002	2.0002	N.D.	N.D.	N.D.

Table 4.9: Error decay for function F4, in the original IL formulation, using finite volume ($\mu = 1$).

δ_h	$\bar{\varepsilon}$	$\|\varepsilon\|_2$	ε_{max}	$\bar{\varepsilon}^{Sup}$	$\|\varepsilon\|_2^{Sup}$	ε_{max}^{Sup}	$\bar{\varepsilon}^{Trans}$	$\|\varepsilon\|_2^{Trans}$	$\varepsilon_{max}^{Trans}$
δ_3^*	0.9974	0.4998	N.D.	0.9952	0.9956	0.9957	N.D.	N.D.	N.D.
δ_{2002}	0.9990	0.4997	N.D.	0.9976	0.9979	0.9976	N.D.	N.D.	N.D.
δ_{2103}	0.9978	0.4999	N.D.	0.9968	0.9971	0.9958	N.D.	N.D.	N.D.
δ_{4206}	0.9967	0.4999	N.D.	0.9919	0.9926	0.9929	N.D.	N.D.	N.D.

4.3 Equation for the Normal Derivative Jump

In this subchapter, the improvement strategy concerning the additional equation expressing the normal derivative jump (shortly referred to as the σ-equation) is tested. From now on, all the results will concern function F4, since it is the most general function and the one with the worst results in the original formulation. Moreover, all the results will be obtained using the Poisson equation in a finite volume implementation, that is, using the first equation appearing in system 2.16. The second equation of system 2.16 will be discarded and the new equations of subchapter 3.2 will now be used. The decays obtained using the additional equation for the normal derivative jump with one superfluous point from each subdomain, derived in section 3.2.1, are presented in table 4.10. The decays considering the use of two superfluous points from each subdomain, derived in section 3.2.2, are shown in table 4.11.

Table 4.10: Error decay for function F4, using the σ-equation with one superfluous point ($\mu = 1$).

δ_h	$\bar{\varepsilon}$	$\|\varepsilon\|_2$	ε_{max}	$\bar{\varepsilon}^{Sup}$	$\|\varepsilon\|_2^{Sup}$	ε_{max}^{Sup}	$\bar{\varepsilon}^{Trans}$	$\|\varepsilon\|_2^{Trans}$	$\varepsilon_{max}^{Trans}$
δ_3^*	1.0003	0.5003	N.D.	1.9964	1.9964	1.9892	N.D.	N.D.	N.D.
δ_{2002}	1.0012	0.5000	N.D.	1.9979	1.9981	1.9937	N.D.	N.D.	N.D.
δ_{2103}	1.0031	0.5004	N.D.	1.9972	1.9973	1.9929	N.D.	N.D.	N.D.
δ_{4206}	1.0059	0.5002	N.D.	1.9951	1.9952	1.9945	N.D.	N.D.	N.D.

Table 4.11: Error decay for function F4, using the σ-equation with two superfluous points ($\mu = 1$).

δ_h	$\bar{\varepsilon}$	$\|\varepsilon\|_2$	ε_{max}	$\bar{\varepsilon}^{Sup}$	$\|\varepsilon\|_2^{Sup}$	ε_{max}^{Sup}	$\bar{\varepsilon}^{Trans}$	$\|\varepsilon\|_2^{Trans}$	$\varepsilon_{max}^{Trans}$
δ_3^*	1.0003	0.5003	N.D.	1.9938	1.9937	2.0092	N.D.	N.D.	N.D.
δ_{2002}	1.0001	0.5000	N.D.	1.9967	1.9969	2.0053	N.D.	N.D.	N.D.
δ_{2103}	1.0004	0.5004	N.D.	1.9948	1.9949	2.0106	N.D.	N.D.	N.D.
δ_{4206}	1.0000	0.5002	N.D.	2.2334	2.1124	2.0915	N.D.	N.D.	N.D.

When comparing the results presented in tables 4.10 and 4.11 with the ones in table 4.9, one concludes that the alternative equation, both considering one or two superfluous points, is able to achieve second-order decays in the superfluous region for all the error parameters. However, as expected for a discontinuous function (recall the phenomenon of figure 4.5), the numerical solution continues not decaying in the transition region. Consequently, the global decays continue looking like in table 4.9.

Note that the version using one superfluous point used unilateral first-order schemes to compute the first derivative at each side of the interface. However, second-order accuracy is still obtained in the superfluous region. This phenomenon is reported by Leveque and Li in article [9], concerning the derivation of the Immersed Interface Method. According to them, having a first-order error exclusively in a set of lower dimension does not prevent obtaining global second-order accuracy. In other words, a first-order error in a set of lower dimension does not contaminate a second-order error existing around that set. As previously referred, this method computes correction terms near the interface, so that undesirable terms in the truncation error vanish. Indeed, in the immediate vicinity of the interface, they only ensure that the truncation error is first-order and it did not prevent global second-order accuracy. Similarly, here, the equation for σ expresses the normal derivative jump for the solid points discretising the interface, that constitutes a lower dimension set. Notice that the normal derivative jump, σ, is a quantity only defined at the interface. So, following the reasoning of Leveque and Li [9], a first-order error exclusively at the interface would not be able to contaminate a second-order superfluous decay.

Besides evaluating the orders of decay, it is important to study the value of the error quantities, in order to compare the effect of the discrete delta function. As such, figure 4.7 shows the decay graphs of the superfluous error parameters with one superfluous point, whose data was presented in table 4.10. In figure 4.8, the same graphs are displayed for the case of considering two superfluous points.

As figure 4.7 suggests, the discrete delta function that leads to the lowest errors is δ_{2002}, with almost one order of magnitude difference in the L^2 norm and in the mean error, when compared to the worst one, δ_{4206}. The discrete delta functions δ_{2103} and δ_3^* present similar results, between the previous ones. In the strategy with two superfluous points, figure 4.8 shows that δ_{2103} and δ_3^* continue behaving similarly, but now they are the worst. The best discrete delta function is δ_{4206}, with more than one order of magnitude advantage from the worst ones and around one order of magnitude advantage from δ_{2002}.

Comparing both graphs for the most refined mesh, one observes that the best discrete delta function leads to error parameters around 10^{-6} in the strategy with one superfluous point, whereas the best discrete delta function in the case of two superfluous points achieves errors around 10^{-8}. As such, despite the same order of decay, the use of two superfluous points presents numerical advantage.

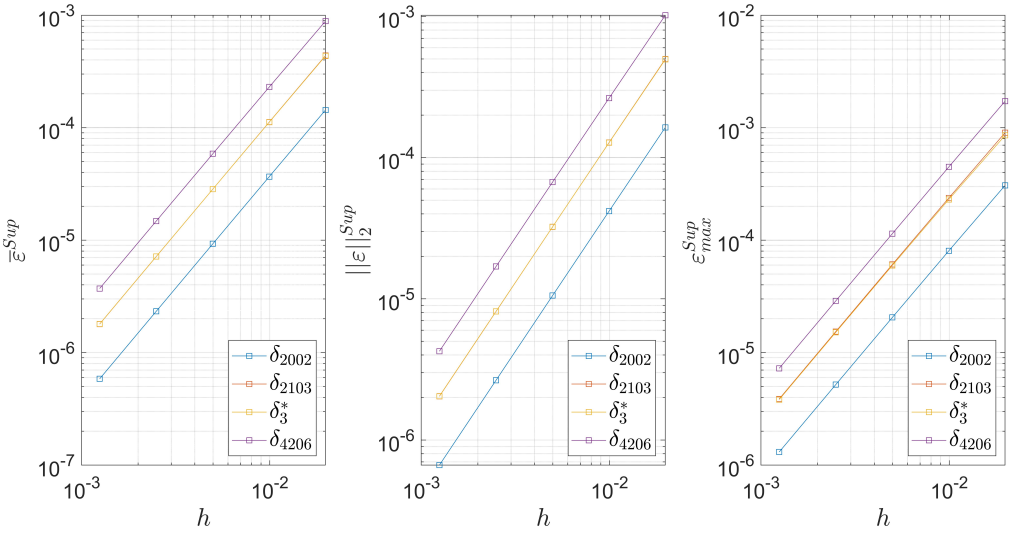


Figure 4.7: Superfluous decay using the σ-equation with one superfluous point.

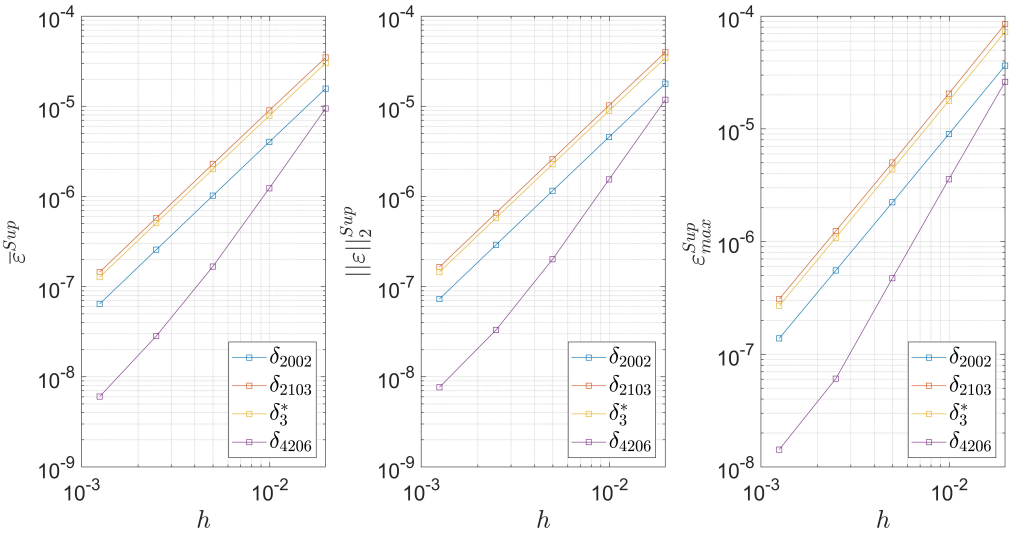


Figure 4.8: Superfluous decay using the σ-equation with two superfluous points.

4.4 Equation Imposing the Interface Value

In this subchapter, the improvement strategy concerning the additional equation expressing the field value at the interface (shortly named as the $\bar{\phi}$-equation) is tested. Again, the results are presented for function F4, using the Poisson equation under a finite volume formulation. The decays obtained for the strategy derived in section 3.3.1, with two superfluous points, are presented in table 4.12. The decays for the version with three superfluous points, derived in section 3.3.2, are shown in table 4.13.

Similarly to what happened with the additional equation expressing the normal derivative jump, this alternative additional equation with extrapolation of the subfields to the interface is able to achieve second-order results in the superfluous region for all discrete delta functions. Note that, as one will

Table 4.12: Error decay for function F4, with $\mu = 1$, using the $\bar{\phi}$-equation with two superfluous points.

δ_h	$\bar{\varepsilon}$	$\|\varepsilon\|_2$	ε_{max}	$\bar{\varepsilon}^{Sup}$	$\|\varepsilon\|_2^{Sup}$	ε_{max}^{Sup}	$\bar{\varepsilon}^{Trans}$	$\|\varepsilon\|_2^{Trans}$	$\varepsilon_{max}^{Trans}$
δ_3^*	1.0042	0.5002	N.D.	1.9961	1.9964	1.9944	N.D.	N.D.	N.D.
δ_{2002}	1.0019	0.5000	N.D.	1.9981	1.9979	1.9967	N.D.	N.D.	N.D.
δ_{2103}	1.0041	0.5003	N.D.	1.9968	1.9964	1.9944	N.D.	N.D.	N.D.
δ_{4206}	1.0071	0.5001	N.D.	1.9944	1.9940	1.9919	N.D.	N.D.	N.D.

Table 4.13: Error decay for function F4, with $\mu = 1$, using the $\bar{\phi}$-equation with three superfluous points.

δ_h	$\bar{\varepsilon}$	$\|\varepsilon\|_2$	ε_{max}	$\bar{\varepsilon}^{Sup}$	$\|\varepsilon\|_2^{Sup}$	ε_{max}^{Sup}	$\bar{\varepsilon}^{Trans}$	$\|\varepsilon\|_2^{Trans}$	$\varepsilon_{max}^{Trans}$
δ_3^*	1.0003	0.5003	N.D.	2.0052	2.0098	2.0444	N.D.	N.D.	N.D.
δ_{2002}	1.0001	0.5000	N.D.	2.0036	2.0063	2.0287	N.D.	N.D.	N.D.
δ_{2103}	1.0004	0.5004	N.D.	2.0046	2.0082	2.0385	N.D.	N.D.	N.D.
δ_{4206}	1.0000	0.5002	N.D.	2.6421	2.6208	2.6091	N.D.	N.D.	N.D.

see, the slightly higher than second-order results for δ_{4206} in table 4.13 are due to the fact that the asymptotic region has not yet been reached and the decay is yet adjusting towards second-order. Once more, the errors in the transition region do not decay, as it was already verified in the original formulation with function F4. Globally, the mean error decays with first-order and the L^2 norm presents the already detected half-order behaviour for discontinuous fields. In order to compare the effect of the discrete delta functions, the superfluous decays of both strategies are presented in figures 4.9 and 4.10.

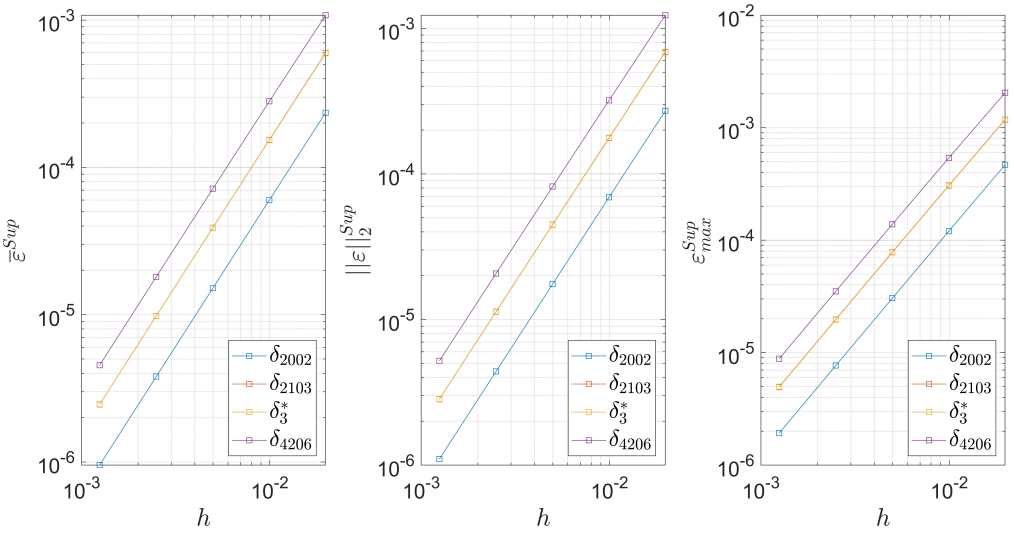


Figure 4.9: Superfluous decay using the $\bar{\phi}$-equation with two superfluous points.

As observed in figure 4.9, the discrete delta function that leads to the best results when using two superfluous points is δ_{2002}, whereas the highest error values are obtained when δ_{4206} is used. The discrete delta functions δ_{2103} and δ_3^* behave similarly, in between. These results from figure 4.9 are very similar to the ones in figure 4.7. In figure 4.9, concerning the use of three superfluous points, one

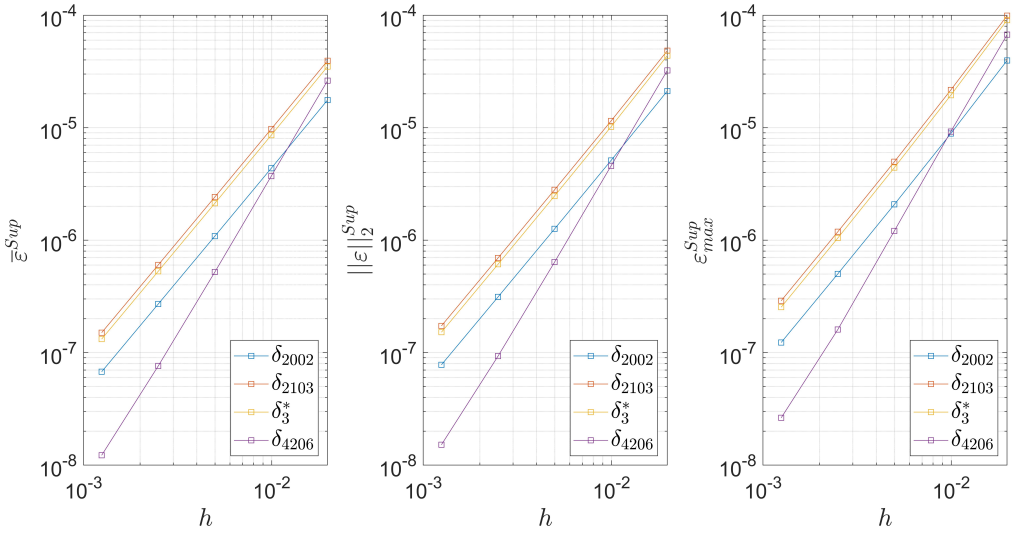


Figure 4.10: Superfluous decay using the $\bar{\phi}$-equation with three superfluous points.

concludes that the slightly higher than second-order decays from for δ_{4206} are explained by the fact that the numerical solution is yet adjusting towards the asymptotic region, contrary to what is verified for the other discrete delta functions, that are already there. For the coarsest mesh, δ_{2002} presents a lower error, but it is immediately overtaken by δ_{4206}. As such, for the most refined grids, these results are similar to the ones in figure 4.8, with δ_{2103} and δ_3^* being the worst discrete delta functions and δ_{4206} being the best.

4.5 Comparison between 1D Methods

Analysing figures 4.7 to 4.10, one may compare the results for a given discrete delta function, in order to determine what is the strategy that leads to the lowest superfluous error. To facilitate the analysis, figure 4.11 displays all the decays concerning δ_{2002} together, collapsing the data presented in figures 4.7 to 4.10. The value in parentheses appearing in the legend refers to the number of superfluous points.

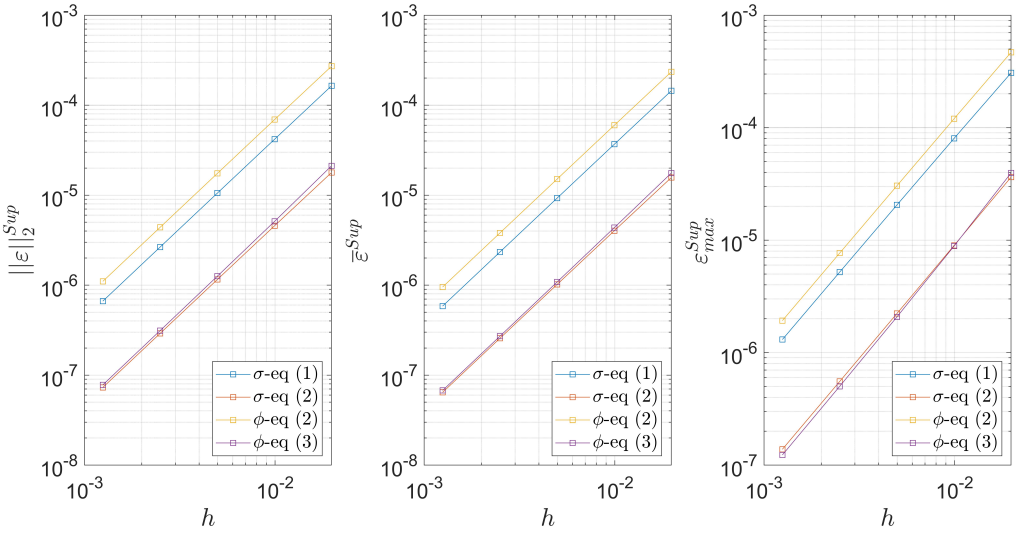


Figure 4.11: Gathering of all the superfluous decays with σ-equation and $\bar{\phi}$-equation strategies for δ_{2002}.

As figure 4.11 indicates, the strategy corresponding to the σ-equation using two superfluous points presents the lowest L^2 and mean error values, being the second best in terms of maximum error for the most refined grids and the best for the most coarse ones. The graphs concerning the other discrete delta functions, available in appendix C.1, reveal identical results for δ_3^* and δ_{2103}. In the case of discrete delta function δ_{4206}, the strategy corresponding to the σ-equation using two superfluous points is always the best. Therefore, ordering all the strategies from the worst to the best, that is, from the one with highest errors to the one with the lowest ones, overall one has: $\bar{\phi}$-equation (2), σ-equation (1), $\bar{\phi}$-equation (3), σ-equation (2). As such, from all the derived one-dimensional improvement strategies in chapter 3, one concludes that the σ-equation strategy with two superfluous points is the one presenting the best numerical results. Moreover, recalling figure 4.8, the discrete delta function that led to the lowest error values when conjugated with this strategy was δ_{4206}.

As already shown, all these alternative strategies present second-order accuracy in the superfluous region and the original formulation (both in finite differences and finite volume) is only first-order accurate. Therefore, all the alternative strategies are better than the original one-dimensional formulation.

4.6 Smearing Correction

In this subchapter, the three one-dimensional smearing corrective procedures presented in subchapter 3.4 are implemented and compared. The main objective is to find out if they are able to propagate the second-order decays achieved in the superfluous region to the transition region and, consequently, to the entire domain. To test the corrective procedures, one considers, as an example, the σ-equation with two superfluous points, since it was the strategy with the best numerical results. The decays obtained using corrective procedures C1S, C2 and C2S are presented in tables 4.14, 4.15 and 4.16, respectively.

Table 4.14: Error decay for function F4, with $\mu = 1$, using strategy σ-equation (2), with correction C1S.

δ_h	$\bar{\varepsilon}$	$\|\varepsilon\|_2$	ε_{max}	$\bar{\varepsilon}^{Sup}$	$\|\varepsilon\|_2^{Sup}$	ε_{max}^{Sup}	$\bar{\varepsilon}^{Trans}$	$\|\varepsilon\|_2^{Trans}$	$\varepsilon_{max}^{Trans}$
δ_3^*	2.0120	2.0639	1.9963	1.9938	1.9937	2.0092	1.9986	1.9975	1.9963
δ_{2002}	2.0007	2.0066	1.9983	1.9967	1.9969	2.0053	1.9993	1.9989	1.9983
δ_{2103}	2.0109	2.0513	1.9963	1.9948	1.9949	2.0106	1.9987	1.9974	1.9963
δ_{4206}	2.6650	2.4952	1.9960	2.2334	2.1124	2.0915	2.0001	1.9965	1.9960

Table 4.15: Error decay for function F4, with $\mu = 1$, using strategy σ-equation (2), with correction C2.

δ_h	$\bar{\varepsilon}$	$\|\varepsilon\|_2$	ε_{max}	$\bar{\varepsilon}^{Sup}$	$\|\varepsilon\|_2^{Sup}$	ε_{max}^{Sup}	$\bar{\varepsilon}^{Trans}$	$\|\varepsilon\|_2^{Trans}$	$\varepsilon_{max}^{Trans}$
δ_3^*	2.0794	2.3695	1.9972	1.9938	1.9937	2.0092	1.9990	1.9979	1.9972
δ_{2002}	2.0410	2.2648	1.9983	1.9967	1.9969	2.0053	1.9994	1.9987	1.9983
δ_{2103}	2.0693	2.3421	1.9972	1.9948	1.9949	2.0106	1.9989	1.9979	1.9972
δ_{4206}	2.8649	2.4970	1.9961	2.2334	2.1124	2.0915	1.9996	1.9971	1.9961

Table 4.16: Error decay for function F4, with $\mu = 1$, using strategy σ-equation (2), with correction C2S.

δ_h	$\bar{\varepsilon}$	$\|\varepsilon\|_2$	ε_{max}	$\bar{\varepsilon}^{Sup}$	$\|\varepsilon\|_2^{Sup}$	ε_{max}^{Sup}	$\bar{\varepsilon}^{Trans}$	$\|\varepsilon\|_2^{Trans}$	$\varepsilon_{max}^{Trans}$
δ_3^*	1.9917	1.9923	2.0092	1.9938	1.9937	2.0092	2.0000	2.0025	2.0113
δ_{2002}	1.9952	1.9969	2.0053	1.9967	1.9969	2.0053	2.0008	2.0038	2.0070
δ_{2103}	2.0693	2.3421	1.9972	1.9948	1.9949	2.0106	1.9989	1.9979	1.9972
δ_{4206}	2.2320	2.1118	2.0915	2.2334	2.1124	2.0915	2.9865	2.9867	2.9854

As concluded from tables 4.14 to 4.16, all the derived one-dimensional corrective procedures are able to propagate second-order superfluous decays to the entire computational domain. Notice that the superfluous decays are identical to the ones of table 4.11, since the corrective procedures only act on transition values. As previously referred, contrary to what happens with the other discrete delta functions, the superfluous values concerning δ_{4206} are still adjusting towards the asymptotic behaviour and, consequently, the corrective procedures are acting over a superfluous field that is yet characterised by a decay of order higher than second. As such, some entries of tables 4.14 to 4.16 concerning the transition region and the entire domain for δ_{4206} display higher than second-order values as well.

In order to compare the corrective procedures between themselves, one should fix the discrete delta function and focus on the error parameters that each procedure originates. As an example, consider the decays obtained with δ_{4206} for all the corrective procedures, shown in figure 4.12, in which C0 stands for not using any correction. The graphs for the other discrete delta functions are available in appendix C.2.

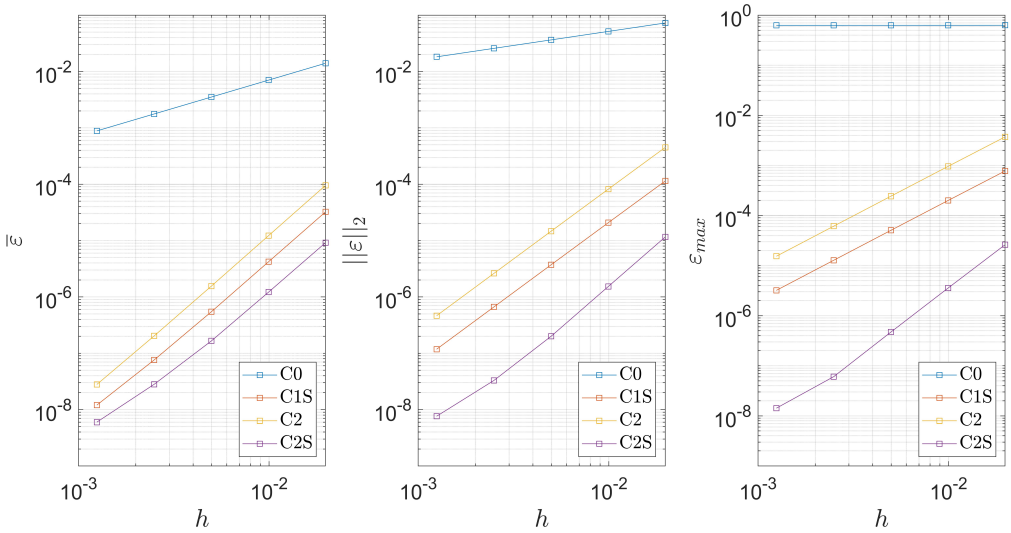


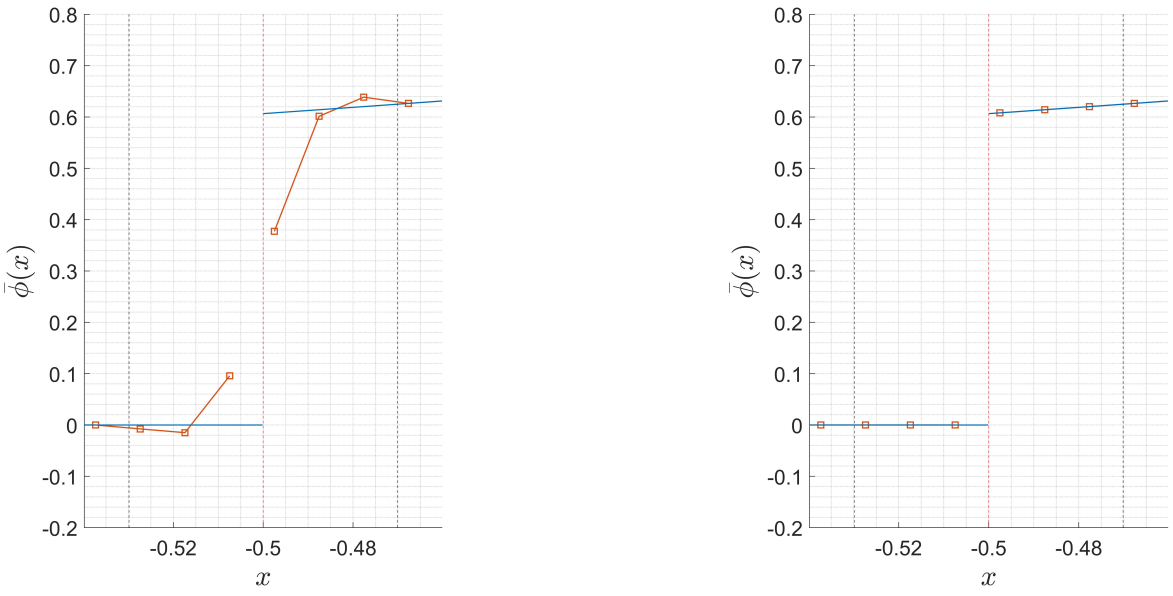
Figure 4.12: Gathering of all the global decays using different corrective procedures, for δ_{4206}.

According to figure 4.12, the corrective procedure C2S, using the solid point and two superfluous points, is the one leading to the lowest error in the corrected transition region. As explained in section 3.4.3, although all the corrective procedures act on the same superfluous field, strategy C2S propagates the superfluous values to the transition region with a formulation that is third-order accurate, whereas strategies C1S and C2 perform that propagation with a second-order accurate formulation. However,

despite the smallest error due to the third-order propagation, the second-order nature of the superfluous region field will never allow higher orders of accuracy to manifest in the transition region (and consequently in the entire domain), unless the superfluous decay is not yet stabilised. Indeed, this is what is happening when combining corrective strategy C2S with discrete delta function δ_{4206}, whose superfluous results were still adjusting towards second-order (recall the graph presented in figure 4.8).

As a final remark, besides C1S and C2 being both second-order accurate corrective procedures, C2 reveals to be worst for all the error parameters. This result can be explained by the fact that, while procedure C2 uses two superfluous points whose values arose from the numerical simulation, procedure C1S uses one superfluous point together with one solid point whose analytical value is known. The graphs available in appendix C.2 for the other discrete delta functions lead to the same qualitative conclusion, with C2 being the procedure with the highest errors and C2S presenting the lowest ones.

In figure 4.13, the recovery of a sharp interface is exemplified around solid point x_{sp_1}, considering the use of σ-equation strategy with two superfluous points and discrete delta function δ_{4206}. The transition region is indicated by vertical dark dashed lines and the location of the interface appears as a vertical red dashed line. Since function F4 is a discontinuous field, the discrete delta function leads to a smeared interface, as represented in figure 4.13a. Once one of the derived corrective procedures is applied, the interface sharpness is recovered. To obtain figure 4.13b, procedure C2S was used as an example. Notice that the superfluous values remain untouched and only transition points are corrected.



(a) Smeared interface before correction.

(b) Sharp interface after correction C2S.

Figure 4.13: Interface before and after correction, with strategy σ-equation (2), δ_{4206} and $n = 201$. The numerical solutions are represented in orange and the analytical function appears represented in blue.

Chapter 5

2D Improvement Strategies

In this chapter, several improvement strategies are presented concerning a two-dimensional problem. In chapter 3, two lines of thought were followed to close the system of equations: deducing an expression to the normal derivative jump across the interface, or deducing an equation to impose the value at the interface. As seen in chapter 4 concerning the one-dimensional results, the addition of an equation to model the normal derivative jump revealed to be more precise in the one-dimensional tests, achieving similar orders of decay, but presenting a lower error. Thus, taking into account the one-dimensional results, the two-dimensional improvement strategy followed in this work focused on deriving additional equations to express σ, instead of equations aiming to impose the field value at the interface.

This chapter begins with a geometrical description of a two-dimensional problem, in subchapter 5.1. Then, several strategies to express the single-layer potential strength are derived. In subchapter 5.2, a direct method is deduced. In subchapter 5.3, a least squares strategy is employed, with two possible approaches to deal with the solid point in the least squares polynomial. In subchapter 5.4, a two-dimensional method faithfully based in the one-dimensional line of thought is derived. Finally, in subchapter 5.5, two-dimensional corrective procedures are presented, to replace the numerical transition values (affected by the smearing phenomenon) by new estimates based on the superfluous results.

5.1 Problem Description

In two dimensions, one considers a square domain $\Omega = \{(x, y) \in \mathbb{R}^2 : x \in [-\frac{L}{2}; \frac{L}{2}] \wedge y \in [-\frac{L}{2}; \frac{L}{2}]\}$, for a given length $L \in \mathbb{R}$. The solid body constitutes an interior subdomain Ω^-, surrounded by a fluid exterior subdomain Ω^+. The solid-fluid interface Γ is discretised into p solid points spaced by a distance δs, measured along the surface. A general two-dimensional problem is illustrated in figure 5.1.

The coordinate system (x, y) is characterised by the x-axis pointing to the right and the y-axis pointing upwards. Since the proposed improvement strategy relies on deriving expressions to the normal derivative jump at each solid point, it is useful to work in local coordinate systems, in which the normal direction is readily available. In that regard, a local coordinate system (η, ξ) is represented in figure 5.1. The η-axis is tangent to the interface and advances counterclockwise, whereas the ξ-axis is normal to

the interface and points according to the exterior unit normal.

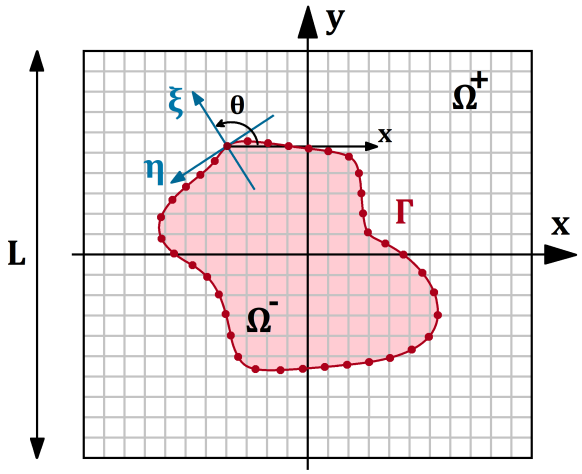


Figure 5.1: Illustration of the local coordinate system centred in a given solid point $(x_{sp}, y_{sp}) \in \Gamma$.

The transformation of coordinates from the global referential (x, y) to the local referential centred in a given solid point $(x_{sp}, y_{sp}) \in \Gamma$ is performed according to equations 5.1. These expressions and this nomenclature are similar to what is presented in the formulation of the IIM, by Leveque and Li [9]. The angle θ depends on the solid point around which the local referential is considered, being measured counterclockwise between the Cartesian x-axis and the normal ξ-axis, as exemplified in figure 5.1.

$$\begin{cases} \xi = (x - x_{sp})\cos(\theta) + (y - y_{sp})\sin(\theta) \\ \eta = -(x - x_{sp})\sin(\theta) + (y - y_{sp})\cos(\theta) \end{cases} \tag{5.1}$$

The computational domain is discretised into a Cartesian grid with a total of $N = n^2$ square cells, leading to a grid size $h = L/n$ in finite volume and $h = L/(n-1)$ in finite differences. Notice that the domain discretisation presented in figure 5.1 corresponds to a finite volume reasoning, with domain boundaries coinciding with cell faces. In order to determine the number p of equally spaced solid points, the relative distance between them was targeted as $\delta s/h = 1$. However, most likely, this exact value will be impossible to reconcile with the assumption of equally spaced solid points. In two dimensions, given the perimeter of the interface, P, the number of solid points was computed by applying relation 5.2 with the target ratio $(\delta s/h)_{target} = 1$. In general, the fraction in equation 5.2 is not a natural number. As such, the floor operation was applied, causing the true relative distance to be different from the target one.

$$p = \left\lfloor \frac{P}{\left(\frac{\delta s}{h}\right)_{target} h} \right\rfloor = \left\lfloor \frac{P}{h} \right\rfloor \tag{5.2}$$

The true relative distance between solid points, $(\delta s/h)_{real}$, may be computed with the number of solid points that resulted from equation 5.2, by simply applying the expression of equation 5.3. The more refined the grid, the lower will be the error between the real relative distance and the target one.

$$\left(\frac{\delta s}{h}\right)_{real} = \frac{(P/p)}{h} \tag{5.3}$$

5.2 Direct Method

The Direct Method (DM) strategy reconstructs a local polynomial around each solid point, that is later used to express the jump in normal derivative across the interface, σ. Once the number of terms desired to incorporate the polynomial is set, an equal number of neighbouring points is used to perform the reconstruction. Thus, the reconstruction of the polynomial is made *directly*, meaning that the number of unknown coefficients in the polynomial expansion, n_C, is equal to the number of sample points used to compute them, n_s. The polynomial expansion, expressed in local coordinates with a general number of n_C terms is presented in equation 5.4, where variables $m_\eta, m_\xi \in \mathbb{N}$ stand for the exponents in the last considered term to which η and ξ are raised. Notice that one needs to perform a polynomial expansion for each side of the interface, $+$ and $-$, to obtain the normal derivative coming from each subdomain.

$$\phi^{\pm}(\eta,\xi) = C_0^{\pm} + C_1^{\pm}\eta + C_2^{\pm}\xi + C_3^{\pm}\eta\xi + ... + C_{n_c-1}^{\pm}\eta^{m_\eta}\xi^{m_\xi} \tag{5.4}$$

Having n_C coefficients in the polynomial expansion, one selects $n_s = n_C$ points available in the domain (grid points or solid points) to compute them. Expressing polynomial 5.4 for each of the sample points, one reaches the systems of equations 5.5 (one for each subdomain, $+$ and $-$).

$$\begin{pmatrix} 1 & 0 & 0 & \dots & 0 \\ 1 & \eta_1^{\pm} & \xi_1^{\pm} & \dots & (\eta_1^{\pm})^{m_\eta}(\xi_1^{\pm})^{m_\xi} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \eta_{n_C-1}^{\pm} & \xi_{n_C-1}^{\pm} & \dots & (\eta_{n_C-1}^{\pm})^{m_\eta}(\xi_{n_C-1}^{\pm})^{m_\xi} \end{pmatrix}_{n_C\times n_C} \begin{pmatrix} C_0^{\pm} \\ C_1^{\pm} \\ \vdots \\ C_{n_C-1}^{\pm} \end{pmatrix}_{n_C\times 1} = \begin{pmatrix} \phi_0^{\pm} \\ \phi_1^{\pm} \\ \vdots \\ \phi_{n_C-1}^{\pm} \end{pmatrix}_{n_C\times 1} \tag{5.5}$$

As indicated by the first line of the square matrix, the first point included in the process (to both $+$ and $-$ expansions) is the solid point in which the local coordinate system is centred, $\phi_0^{\pm}$. Designating the entries of the inverse of the square matrix appearing in equation 5.5 by $(m_{i,j})$, one is able to express each coefficient in terms of the sample values $\phi_0^{\pm}, ..., \phi_{n_C-1}^{\pm}$, according to equation 5.6.

$$C_i^{\pm} = \sum_{j=1}^{n_C} m_{(i+1),j}^{\pm}\phi_{j-1}^{\pm} \quad , \quad i = 0, 1, ..., n_C - 1 \tag{5.6}$$

Having the polynomial expansion coefficients properly expressed in terms of grid and/or solid points, the computation of the normal derivative jump is readily performed, as presented in equation 5.7. Keep in mind that the coordinates of the solid point under treatment are $(\eta_0, \xi_0) = (0, 0)$.

$$\sigma = (\nabla\phi_{sp}^{+} - \nabla\phi_{sp}^{-}) \cdot \mathfrak{n} = \left.\frac{\partial\phi^{+}}{\partial\xi}\right|_{(0,0)} - \left.\frac{\partial\phi^{-}}{\partial\xi}\right|_{(0,0)} = C_2^{+} - C_2^{-} \tag{5.7}$$

To proceed with the derivation of the additional equation for σ, further assumptions concerning the sample values $\phi_1^{\pm}, ..., \phi_{n_C-1}^{\pm}$ must be made. As known from numerical methods based on polynomial interpolation, to compute a first derivative with second-order accuracy along a given direction, one needs, in the most general scenario, three points along that direction (so that a quadratic polynomial may be

constructed). Thus, to compute both the normal and the tangential first derivatives with second-order accuracy, one needs to have a 3×3 block of points, with three normal levels and three tangential levels represented. Considering a general order of accuracy ρ in the computation of the first derivative, table 5.1 summarises the $(\rho+1)^2$ required terms. For example, in order to achieve first-order accuracy in the computation of the first derivative, one needs the four polynomial terms in yellow, implying the use of a 2×2 block of points. For second-order accuracy in the computation of the first derivative, the orange terms should be included as well, implying the use of a 3×3 block. For third-order accuracy, the inclusion of the red terms leads to a 4×4 block, and so on.

Table 5.1: Terms to achieve an order of accuracy ρ in the first derivative, with a direct method.

1	ξ	ξ^2	ξ^3	$\dots$	ξ^ρ
η	$\eta\xi$	$\eta\xi^2$	$\eta\xi^3$	$\dots$	$\eta\xi^\rho$
η^2	$\eta^2\xi$	$\eta^2\xi^2$	$\eta^2\xi^3$	$\dots$	$\eta^2\xi^\rho$
η^3	$\eta^3\xi$	$\eta^3\xi^2$	$\eta^3\xi^3$	$\dots$	$\eta^3\xi^\rho$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
η^ρ	$\eta^\rho\xi$	$\eta^\rho\xi^2$	$\eta^\rho\xi^3$	$\dots$	$\eta^\rho\xi^\rho$

In this work, points $\phi_0^\pm$ were set as the solid point under study, points $\phi_1^\pm$ as the previous solid point, and points $\phi_2^\pm$ as the next one. This choice already defines the first normal level with three tangential levels represented in it. The second and third normal levels were defined as being composed by the closest three points belonging to the first and second layers of superfluous points, respectively. A superfluous point is said to belong to the first superfluous layer if its distance to the interface, d_Γ, is such that $\frac{w}{2}h \leq d_\Gamma < (\frac{w}{2}+1)h$. Points in the second superfluous layer verify $(\frac{w}{2}+1)h \leq d_\Gamma < (\frac{w}{2}+2)h$. The layout of two 3×3 block stencils is exemplified in figure 5.2, for the exterior subdomain. As in the one-dimensional case, any point from the transition region should be invoked to integrate the blocks.

(a) Example 1. (b) Example 2.

Figure 5.2: Exterior blocks used in the Direct Method. The solid points appear in blue, the orange points come from the first superfluous layer and yellow points come from the second superfluous layer.

Notice that, as figure 5.2 suggests, the stencils do not exactly look like 3×3 blocks. This is inevitable, since the solid points are characterised by the tangential and normal directions, but the superfluous points obey to the Cartesian layout. In figure 5.2a, one sees three well defined normal levels and a small misalignment between the tangential position of corresponding points in different normal levels. In figure

5.2b, however, there is a notorious misalignment concerning both tangential and normal levels.

As one already defined that $\phi_0^{\pm}$, $\phi_1^{\pm}$, $\phi_2^{\pm}$ are solid points, the expression for the normal derivative jump takes the form presented in equation 5.8. In the particular case of a 3×3 block, one has $n_C = 9$ points, of which $\sqrt{n_C} = 3$ are solid point values composing the first normal level. Since the solid point values are known, those terms are put in the right-hand side, whereas unknown quantities are kept on the left. This way, the filling of matrices $\mathbf{M_C}$, $\mathbf{M_D}$ and right-hand side vector $\mathbf{v_b}$ becomes immediate.

$$\sum_{j=\sqrt{n_C}+1}^{n_C} m_{3,j}^{+}\phi_{j-1}^{+} - \sum_{j=\sqrt{n_C}+1}^{n_C} m_{3,j}^{-}\phi_{j-1}^{-} + (-1)\sigma = -\left(\sum_{j=1}^{\sqrt{n_C}} m_{3,j}^{+}\phi_{j-1}^{+} - \sum_{j=1}^{\sqrt{n_C}} m_{3,j}^{-}\phi_{j-1}^{-} \right) \tag{5.8}$$

5.3 Least Squares Method

As displayed in figure 5.2, a possible drawback of the Direct Method may arise due to the geometrical imperfection inherently observed in the stencil blocks. In that regard, a Least Squares Method was also implemented, in which the n_C unknown polynomial coefficients are computed by using a sample of n_s points, with $n_s > n_C$. Therefore, with this strategy, one may have a cloud of sample points encompassing the region that would be occupied by a hypothetical 3×3 perfect block.

Since $n_s > n_C$, computing the coefficients from the set of sample points is an overdetermined problem, with more equations than unknowns, and the n_C coefficients are unable to exactly match the n_s restrictions. In other words, the computed coefficients are a compromise solution between all the constraints, and will always lead to a so-called fitting error. That is, when substituting the obtained coefficients in the polynomial, one will not recover exactly the sample values, but approximations.

One advantage of the least squares formulation consists in the need of a lower number of terms in the polynomial expansion when compared with a direct method, to achieve the same order of accuracy. This conclusion may be taken from the work of Diogo [16]. The needed terms to achieve a given order of accuracy ρ in the computation of the first derivative with a least squares formulation are summarised in table 5.2. This table constitutes an adaptation of a similar one in article [16], due to nomenclature differences and since there the objective is the computation of function values instead of first derivatives.

Table 5.2: Terms to achieve an accuracy ρ in the first derivative with least squares, adapted from [16].

1	ξ	ξ^2	ξ^3	$\dots$	ξ^ρ
η	$\eta\xi$	$\eta\xi^2$	$\eta\xi^3$	$\dots$	$\eta\xi^\rho$
η^2	$\eta^2\xi$	$\eta^2\xi^2$	$\eta^2\xi^3$	$\dots$	$\eta^2\xi^\rho$
η^3	$\eta^3\xi$	$\eta^3\xi^2$	$\eta^3\xi^3$	$\dots$	$\eta^3\xi^\rho$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
η^ρ	$\eta^\rho\xi$	$\eta^\rho\xi^2$	$\eta^\rho\xi^3$	$\dots$	$\eta^\rho\xi^\rho$

For example, to present first-order accuracy in the computation of a first derivative, a least squares formulation only needs to work with polynomials relying on the terms in yellow. To achieve second-order accuracy in the first derivative, the formulation should also include the orange terms, and so on.

As such, from table 5.2, one concludes that a given order of accuracy ρ is achieved by a least squares polynomial containing $n_C = (\rho+1)(\rho+2)/2$ terms. Since the least squares formulation constitutes, by definition, an overdetermined problem, one needs to ensure that the sample of n_s points selected in the implementation is such that $n_s > n_C$, being their selection discussed in section 5.3.3. All this information concerning the Least Squares Method is summarised in table 5.3, side by side with the Direct Method.

Table 5.3: Comparison between the Direct Method and the Least Squares Method.

Order of Accuracy in	Direct Method		Least Squares Method	
the First Derivative (ρ)	$n_C = (\rho+1)^2$	n_s	$n_C = \frac{(\rho+1)(\rho+2)}{2}$	n_s
1	4	4	3	> 3
2	9	9	6	> 6
3	16	16	10	> 10

As referred, given the overdetermination, the coefficients arising from a least squares formulation are unable to exactly match the n_s constraints. Particularly, if the solid point itself is used in the sample, its value in the least squares polynomial will also be associated to a fitting error. In order to find out if this is problematic, one derives two different strategies. In section 5.3.1, one proceeds working with the polynomial of equation 5.4, without any special treatment concerning the fitting error. In section 5.3.2, a second strategy is derived, by imposing the solid point value directly on the least squares polynomial.

5.3.1 Strategy with No Polynomial Imposition of the Solid Point Value

In this strategy, the polynomial of equation 5.4 is taken as the starting point, containing the n_C terms arising from table 5.2 (for second-order accuracy, one has the yellow and the orange ones). Given a sample of $n_s > n_C$ points, the relation between polynomial coefficients and sample values is expressed, in local coordinates, according to equation 5.9. Notice that, in general, the number of sample points used in the interior subdomain, n_s^-, does not need to match the corresponding exterior number, n_s^+.

$$\begin{pmatrix} 1 & 0 & 0 & \dots & 0 \\ 1 & \eta_1^{\pm} & \xi_1^{\pm} & \dots & (\eta_1^{\pm})^{m_\eta}(\xi_1^{\pm})^{m_\xi} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \eta_{n_s^{\pm}-1}^{\pm} & \xi_{n_s^{\pm}-1}^{\pm} & \dots & (\eta_{n_s^{\pm}-1}^{\pm})^{m_\eta}(\xi_{n_s^{\pm}-1}^{\pm})^{m_\xi} \end{pmatrix}_{n_s^{\pm} \times n_c} \begin{pmatrix} C_0^{\pm} \\ C_1^{\pm} \\ \vdots \\ C_{n_c-1}^{\pm} \end{pmatrix} = \begin{pmatrix} \phi_0^{\pm} \\ \phi_1^{\pm} \\ \vdots \\ \phi_{n_s^{\pm}-1}^{\pm} \end{pmatrix} \tag{5.9}$$

For the sake of compactness, one will name the rectangular matrix in equation 5.9 as $\mathbf{D}$ (as its entries rely on normal and tangential distance values between sample points and the solid point under study), the coefficients vector by $\mathbf{c}$, and the vector of sample points by $\boldsymbol{\phi_{sa}}$. So, one has $\mathbf{Dc} = \boldsymbol{\phi_{sa}}$.

Since the computation of the coefficients from the set of sample points constitutes an overdetermined problem, when substituted into equation 5.9, the coefficients will lead to a given right-hand side $\boldsymbol{\phi_{sa}^{LS}}$, different from $\boldsymbol{\phi_{sa}}$. In other words, the computed coefficients are a compromise solution, corresponding to the minimisation of a given residual function. Usually, the residual function incorporates a diagonal

weight function matrix $\mathbf{W_{LS}}$, assigning different weights to each sample point. The least squares residual function is given by $\mathbf{r}^T\mathbf{W_{LS}}\mathbf{r}$, with $\mathbf{r}$ being the residual vector $\mathbf{r} = \boldsymbol{\phi_{sa}} - \boldsymbol{\phi_{sa}^{LS}}$, as derived in the work of Diogo [16]. Thus, the least squares problem corresponds to perform the minimisation of equation 5.10.

$$\min\left(\mathbf{r}^T\mathbf{W_{LS}}\mathbf{r}\right) = \min\left[(\boldsymbol{\phi_{sa}} - \boldsymbol{\phi_{sa}^{LS}})^T\mathbf{W_{LS}}(\boldsymbol{\phi_{sa}} - \boldsymbol{\phi_{sa}^{LS}})\right] \tag{5.10}$$

Following the work of Vasconcelos [17], the weights appearing in the diagonal of the weight function matrix are given by equation 5.11, in which the shape factor κ is taken as the classical value $\kappa = 2$. Note that not considering weights is equivalent to assume an identity weight function matrix, $(\mathbf{I})_{n_s \times n_s}$.

$$(w_{LS})_{i,i} = \frac{1}{(\sqrt{(\eta_i - \eta_0)^2 + (\xi_i - \xi_0)^2})^\kappa} = (\eta_i^2 + \xi_i^2)^{-\kappa/2} \tag{5.11}$$

The computation of the coefficients according to the a least squares procedure is presented in the deduction of equation 5.12 , being introduced a pseudoinverse matrix $\mathbf{P_I} = (\mathbf{D}^T\mathbf{W_{LS}}\mathbf{D})^{-1}(\mathbf{D}^T\mathbf{W_{LS}})$.

$$\begin{aligned} &(\mathbf{D}^T\mathbf{W_{LS}})\mathbf{D}\mathbf{c} = (\mathbf{D}^T\mathbf{W_{LS}})\boldsymbol{\phi_{sa}} \\ \Leftrightarrow\ &\mathbf{c} = (\mathbf{D}^T\mathbf{W_{LS}}\mathbf{D})^{-1}(\mathbf{D}^T\mathbf{W_{LS}})\boldsymbol{\phi_{sa}} \end{aligned} \tag{5.12}$$

Therefore, from equation 5.12, each coefficient is expressed in terms of sample points as presented in equation 5.13, by means of the entries of the pseudoinverse matrix $(\mathbf{P_I})_{n_C \times n_s}$, here named as $(p_{i,j})$.

$$C_i^{\pm} = \sum_{j=1}^{n_s^{\pm}} p_{i+1,j}^{\pm}\phi_{j-1}^{\pm} \tag{5.13}$$

Having the coefficients computed, one is now able to express the normal derivative jump, following the expression of equation 5.7, that remains valid here. This step is performed in equation 5.14.

$$\sigma = \sum_{j=1}^{n_s^+} p_{3,j}^+\phi_{j-1}^+ - \sum_{j=1}^{n_s^-} p_{3,j}^-\phi_{j-1}^- \tag{5.14}$$

The values of the solid points included in the sample are known. Besides $\phi_0^{\pm}$ being the solid point under study, if for example $\phi_1^{\pm}$ and $\phi_2^{\pm}$ are defined to be the previous and next solid points respectively, the final equation to be incorporated in the system will look like equation 5.15.

$$\sum_{j=4}^{n_s^+} p_{3,j}^+\phi_{j-1}^+ - \sum_{j=4}^{n_s^-} p_{3,j}^-\phi_{j-1}^- + (-1)\sigma = -\left(\sum_{j=1}^{3} p_{3,j}^+\phi_{j-1}^+ - \sum_{j=1}^{3} p_{3,j}^-\phi_{j-1}^-\right) \tag{5.15}$$

Note that one could use a different number of solid points besides the one under study, or even none. What is important is to ensure that one has enough normal and tangential levels represented inside the cloud of sample points. The selection of the sample cloud is discussed in section 5.3.3. To obtain the final equation, one simply needs to put the terms lying on solid point values in the right-hand side, while leaving the single-layer potential strength and the superfluous points in the left-hand side as unknowns.

5.3.2 Strategy with Polynomial Imposition of the Solid Point Value

In order to find out if the existence of a fitting error in the value predicted by the least squares polynomial to the solid point under study is problematic, an alternative strategy is here derived. Notice that the first equation in the matrix system of equation 5.9 states that $\phi^{\pm}(0,0) = C_0^{\pm} = \phi_0^{\pm}$, that is a known quantity. Instead of letting this constraint enter the least squares formulation (that will not be able to respect it exactly due to the overdetermination), one may simply reduce the system of equations by passing $C_0^{\pm}$ to the right-hand side of each equation. This step, that is presented in the system 5.16, corresponds to work with a polynomial of relative values, $\phi^{\pm}(\eta,\xi) - \phi_0^{\pm}$. When comparing with equation 5.9, the new rectangular matrix in equation 5.16 has one less column and one less line.

$$\underbrace{\begin{pmatrix} \eta_1^{\pm} & \xi_1^{\pm} & \cdots & (\eta_1^{\pm})^{m_\eta}(\xi_1^{\pm})^{m_\xi} \\ \vdots & \vdots & \ddots & \vdots \\ \eta_{n_s^{\pm}-1}^{\pm} & \xi_{n_s^{\pm}-1}^{\pm} & \cdots & (\eta_{n_s^{\pm}-1}^{\pm})^{m_\eta}(\xi_{n_s^{\pm}-1}^{\pm})^{m_\xi} \end{pmatrix}}_{(n_s^{\pm}-1)\times(n_C-1)} \begin{pmatrix} C_1^{\pm} \\ \vdots \\ C_{n_C-1}^{\pm} \end{pmatrix} = \begin{pmatrix} \phi_1^{\pm} - \phi_0^{\pm} \\ \vdots \\ \phi_{n_s^{\pm}-1}^{\pm} - \phi_0^{\pm} \end{pmatrix} \tag{5.16}$$

Naming again the rectangular matrix in equation 5.16 as $\mathbf{D}$, the coefficients vector by $\mathbf{c}$, and the vector of sample points by $\boldsymbol{\phi_{sa}}$, the deduction is almost identical to the one derived in the previous section. Minor differences arise in the coefficients numbering, due to the removal of coefficients $C_0^{\pm}$ from the rectangular matrices, and the sample values are now relative to the imposed solid point value. As such, in this second strategy, the expression for each coefficient is presented in equation 5.17.

$$C_i^{\pm} = \sum_{j=1}^{n_s^{\pm}-1} p_{i,j}^{\pm}\left(\phi_j^{\pm} - \phi_0^{\pm}\right) \tag{5.17}$$

The computation of the normal derivative jump in this second strategy continues to follow the general expression provided in equation 5.7. However, when applying that equation, the previously referred minor adjustments will lead to a slightly different expression for σ, as presented in equation 5.18.

$$\sigma = \sum_{j=1}^{n_s^{+}-1} p_{2,j}^{+}\left(\phi_j^{+} - \phi_0^{+}\right) - \sum_{j=1}^{n_s^{-}-1} p_{2,j}^{-}\left(\phi_j^{-} - \phi_0^{-}\right) \tag{5.18}$$

Consider the same example that was explored for the first strategy without imposition. If one considers $\phi_1^{\pm}$ to be the previous solid points and $\phi_2^{\pm}$ to be the next solid points, the final rearranged equation that is added to the system in this second strategy is presented in 5.19.

$$\begin{aligned} \sum_{j=3}^{n_s^{+}-1} p_{2,j}^{+}\phi_j^{+} - \sum_{j=3}^{n_s^{-}-1} p_{2,j}^{-}\phi_j^{-} + (-1)\sigma = &- \left(\sum_{j=1}^{2} p_{2,j}^{+}\left(\phi_j^{+} - \phi_0^{+}\right) - \sum_{j=1}^{2} p_{2,j}^{-}\left(\phi_j^{-} - \phi_0^{-}\right)\right) + \\ &+ \left(\sum_{j=3}^{n_s^{+}-1} p_{2,j}^{+}\phi_0^{+} - \sum_{j=3}^{n_s^{-}-1} p_{2,j}^{-}\phi_0^{-}\right) \end{aligned} \tag{5.19}$$

5.3.3 Selection of the Sample Points

In this section, the selection of sample points to be used in the least squares procedure is addressed. Given the existence of the transition region, from which no grid point should be used, the cloud enclosing the sample points should be further elongated in the normal direction. As such, instead of a circular cloud, a natural choice is to define an elliptical cloud from which the sample points will be selected. The semiaxes of the elliptical cloud are named as s_η and s_ξ along the tangential and normal directions, respectively. Figure 5.3 exemplifies an elliptical cloud with semiaxes $s_\eta = 2h$ and $s_\xi = (\frac{w}{2} + 3)h$.

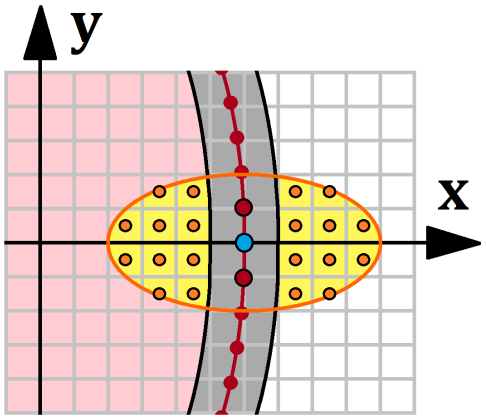


Figure 5.3: Elliptical cloud with semiaxes $s_\eta = 2h$ and $s_\xi = (\frac{w}{2} + 3)h$, for $w = 2$.

The equation describing an ellipse expressed in local coordinates (η, ξ) with semiaxes s_η and s_ξ along the tangential and normal directions, respectively, is presented in equation 5.20.

$$\left(\frac{\eta}{s_\eta}\right)^2 + \left(\frac{\xi}{s_\xi}\right)^2 = 1 \tag{5.20}$$

The elliptical cloud of figure 5.3 encompasses three solid points (the one for which the σ equation is being derived and its previous and next solid points), as well as ten superfluous points from each subdomain. So, in this example, the sample of points to be used in the least squares local polynomial expansion $\phi^+(\eta, \xi)$ would be composed of the ten superfluous points located in subdomain Ω^+, together with the exterior subfield information at the three solid points (with the central one being imposed, or not, depending on the followed strategy). The sample to be used in the local polynomial expansion $\phi^-(\eta, \xi)$ would contain the ten superfluous points located in subdomain Ω^- and the interior subfield information at the three solid points. Summarising, each sample would contain a total of $n_s = 13$ points in this example. In general, the interior and exterior samples may present a different number of points.

As a final remark, despite the lower number of terms in the least squares formulation when compared to the direct approach, the elliptical cloud should be representative of a sufficient number of normal and tangential levels, so that the desired order of accuracy is indeed achieved. In this work, this concern will be accomplished by ensuring that the elliptical cloud encompasses the direct method stencil.

5.4 1D-Inspired Method

In this strategy, the objective is to replicate as faithfully as possible the one-dimensional procedure, whose results achieved second-order accuracy. As such, this method will be named as 1D-Inspired Method (1DIM). The unknown σ is computed by expressing the normal first derivatives of each interface side by means of points strategically placed along the normal direction. Since in general the computation of a first derivative with second-order accuracy needs three points, two points along the normal direction are considered in each side of the interface, besides the solid point, as illustrated in figure 5.4. Since the desired locations are not available in the grid, the desired points (in blue) will need to be constructed using neighbouring grid points (in orange). Figure 5.4 exemplifies with 2×2 blocks of superfluous points.

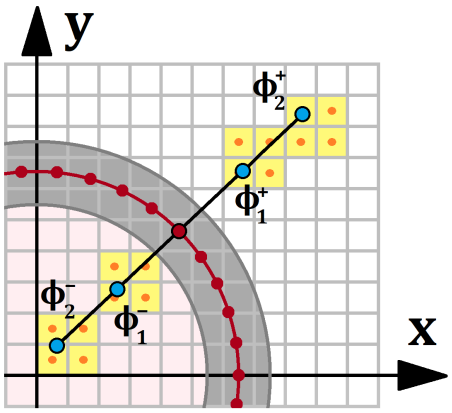
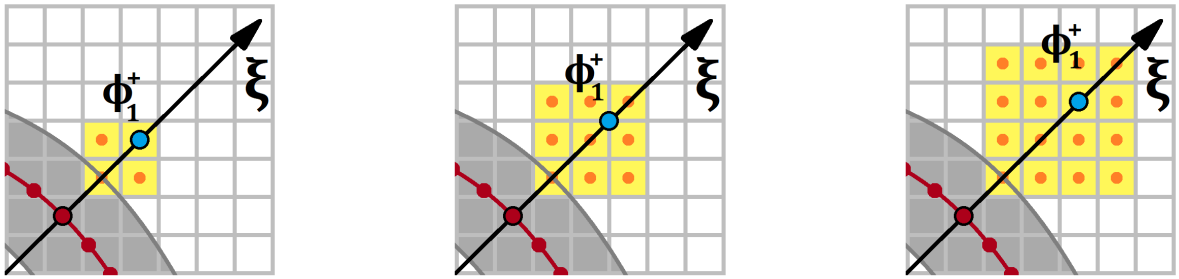


Figure 5.4: Illustration of a 1D-inspired stencil, exemplifying with 2×2 superfluous blocks.

The first step consists of determining the location of points ϕ_1^+, ϕ_2^+, ϕ_1^-, ϕ_2^- along the normal direction. Since these points will invoke a neighbouring block of grid points, the main concern should be that any point inside of the block has its centroid inside the transition region affected by smearing (represented in grey). In other words, all the points invoked to contribute should be superfluous points.

Consider that the position of ϕ_1^+ is known. If the square block presents an odd number of cells along each side, the closest grid point to ϕ_1^+ is found, defining the central point of the block. If the block has an even number of cells along each side, the four closest points surrounding ϕ_1^+ are identified and the square is defined. To compute the minimum distance at which point ϕ_1^+ should be from the solid point, one must consider the worst case scenarios of figure 5.5, for 2×2, 3×3 and 4×4 superfluous blocks.



(a) 2×2 Block. (b) 3×3 Block. (c) 4×4 Block.

Figure 5.5: Superfluous blocks used to determine the minimum distance of point ϕ_1^+ from the solid point.

If ϕ_1^+ is equidistant from several grid points, it becomes indifferent and equally valid to choose between several distinct stencils. Since the determination of the closest points is implemented iteratively, the choice may be left to the program execution order. Consider, for example, figure 5.5b. From all the four equidistant grid neighbours of ϕ_1^+, the block arising from the choice of the southwest one is drawn, simply to compute a worst case scenario distance at which ϕ_1^+ should be located from the solid point. The choice of any of its four equidistant closest neighbours is equally valid during the execution. The minimum distances arising from the worst case scenarios of figure 5.5 are presented in equations 5.21, being the expressions easily generalised to any $k \times k$ superfluous block of grid points, with $k \in \mathbb{N}$.

$$\begin{cases} \text{Block } 2\times 2\text{:} \quad \xi_1^+ = ||\mathbf{x}_1^+ - \mathbf{x}_{sp}|| \geq \left(\frac{w}{2} + 2\frac{\sqrt{2}}{2}\right)h \\ \text{Block } 3\times 3\text{:} \quad \xi_1^+ = ||\mathbf{x}_1^+ - \mathbf{x}_{sp}|| \geq \left(\frac{w}{2} + 3\frac{\sqrt{2}}{2}\right)h \\ \text{Block } 4\times 4\text{:} \quad \xi_1^+ = ||\mathbf{x}_1^+ - \mathbf{x}_{sp}|| \geq \left(\frac{w}{2} + 4\frac{\sqrt{2}}{2}\right)h \\ \qquad\qquad \vdots \\ \text{Block } k\times k\text{:} \quad \xi_1^+ = ||\mathbf{x}_1^+ - \mathbf{x}_{sp}|| \geq \left(\frac{w}{2} + k\frac{\sqrt{2}}{2}\right)h \end{cases} \tag{5.21}$$

According to the desired block dimensions and the respective criterion in equations 5.21, the location of point ϕ_1^+ along the normal axis, ξ_1^+, may be set. The location of point ϕ_2^+ is considered to be $\xi_2^+ = 2\xi_1^+$, so that point ϕ_1^+ is equidistant from ϕ_{sp}^+ and ϕ_2^+. The normal positions of the interior points, ϕ_1^- and ϕ_2^-, are simply taken as the symmetric of the exterior ones, that is, $\xi_1^- = -\xi_1^+$ and $\xi_2^- = -\xi_2^+$.

Having the position of the points along the normal direction already defined, one is able to derive the expression for the single-layer potential strength by using them. In that regard, one begins by writing the polynomial expansions of equation 5.22, one for each side of the interface, with general C coefficients.

$$\phi^{\pm}(\xi) = C_0^{\pm} + C_1^{\pm}\xi + C_2^{\pm}\xi^2 \tag{5.22}$$

The polynomial coefficients of equation 5.22 are computed, for each side, using the solid point value and the two points along the normal direction that one just defined, as presented in equation 5.23.

$$\begin{pmatrix} 1 & 0 & 0 \\ 1 & \xi_1^{\pm} & (\xi_1^{\pm})^2 \\ 1 & \xi_2^{\pm} & (\xi_2^{\pm})^2 \end{pmatrix} \begin{pmatrix} C_0^{\pm} \\ C_1^{\pm} \\ C_2^{\pm} \end{pmatrix} = \begin{pmatrix} \phi_{sp}^{\pm} \\ \phi_1^{\pm} \\ \phi_2^{\pm} \end{pmatrix} \tag{5.23}$$

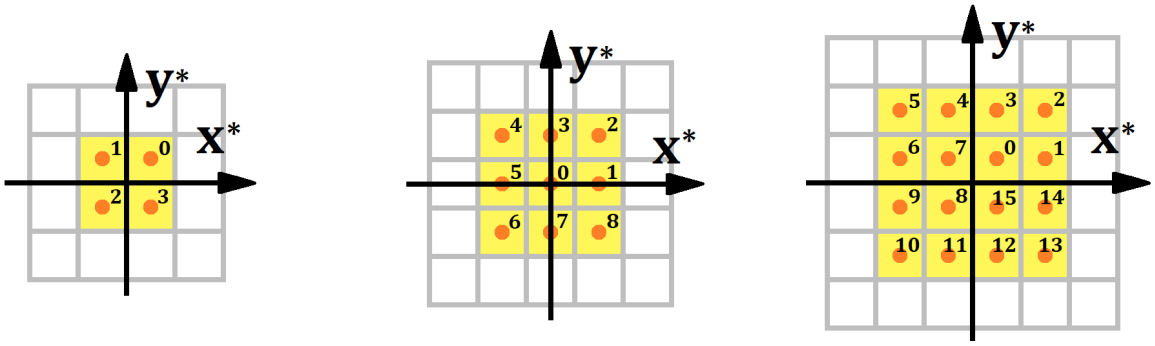
Inverting the square matrix in equation 5.23 and naming its entries as (m_{ij}), equation 5.24 arises.

$$\begin{pmatrix} C_0^{\pm} \\ C_1^{\pm} \\ C_2^{\pm} \end{pmatrix} = \begin{pmatrix} m_{11}^{\pm} & m_{12}^{\pm} & m_{13}^{\pm} \\ m_{21}^{\pm} & m_{22}^{\pm} & m_{23}^{\pm} \\ m_{31}^{\pm} & m_{32}^{\pm} & m_{33}^{\pm} \end{pmatrix} \begin{pmatrix} \phi_{sp}^{\pm} \\ \phi_1^{\pm} \\ \phi_2^{\pm} \end{pmatrix} \tag{5.24}$$

Having the polynomial coefficients computed, the single-layer potential strength is expressed in terms of the normal stencil (solid point values $\phi_{sp}^{\pm}$ together with ϕ_1^+, ϕ_2^+, ϕ_1^-, ϕ_2^-), as presented in equation 5.25.

$$\sigma = \left.\frac{\partial \phi^+}{\partial \xi}\right|_{\xi=0} - \left.\frac{\partial \phi^-}{\partial \xi}\right|_{\xi=0} = C_1^+ - C_1^-$$
$$\Rightarrow \sigma = \left(m_{21}^+ \phi_{sp}^+ + m_{22}^+ \phi_1^+ + m_{23}^+ \phi_2^+\right) - \left(m_{21}^- \phi_{sp}^- + m_{22}^- \phi_1^- + m_{23}^- \phi_2^-\right) \tag{5.25}$$

At this point, the challenge lies in expressing ϕ_1^+, ϕ_2^+, ϕ_1^-, ϕ_2^- in terms of neighbouring superfluous grid points. As an example, one illustrates the procedure for a 2×2 superfluous block. To express points ϕ_1^+, ϕ_2^+, ϕ_1^-, ϕ_2^- as a linear combination of the grid points inside their respective block, one considers a Cartesian local coordinate system (x^*, y^*), with origin at the geometric centre of the superfluous block, as displayed in figure 5.6. Axis x^* and y^* are aligned with x and y, respectively.



(a) 2×2 Block. (b) 3×3 Block. (c) 4×4 Block.

Figure 5.6: Cartesian local coordinate system (x^*, y^*), with origin at the geometric centre of the blocks.

First, a polynomial expressed in local coordinates (x^*, y^*) is constructed, with the same number of coefficients as the number of points that compose the superfluous block. Equation 5.26 applies for the case concerning a 2×2 superfluous block of grid points. To avoid confusion with the coefficients concerning the polynomial expansion along the normal direction, here a star superscript $(*)$ is used.

$$\bar{\phi}(x^*, y^*) = C_0^* + C_1^* x^* + C_2^* y^* + C_3^* x^* y^* \tag{5.26}$$

Taking into account the numbering strategy of figure 5.6a, one arrives to equation 5.27. Note that, once the desired block dimension is defined in the beginning of the simulation, all the superfluous blocks will be identical. As such, the square matrix in equation 5.27, containing the local distances of all the points in a superfluous block, is a constant throughout the execution and it is only inverted once.

$$\begin{pmatrix} 1 & h/2 & h/2 & (h/2)^2 \\ 1 & -h/2 & h/2 & -(h/2)^2 \\ 1 & -h/2 & -h/2 & (h/2)^2 \\ 1 & h/2 & -h/2 & -(h/2)^2 \end{pmatrix} \begin{pmatrix} C_0^* \\ C_1^* \\ C_2^* \\ C_3^* \end{pmatrix} = \begin{pmatrix} \bar{\phi}_0 \\ \bar{\phi}_1 \\ \bar{\phi}_2 \\ \bar{\phi}_3 \end{pmatrix} \tag{5.27}$$

Inverting the square matrix in equation 5.27 and naming its entries as (m_{ij}^*), one is able to compute the polynomial coefficients C^* as a linear combination of the superfluous points that compose the block.

The expression for the coefficients in the 2×2 superfluous block is presented in equation 5.28.

$$C_i^* = \sum_{j=1}^{4} m_{i+1,j}^* \bar{\phi}_{j-1} \quad , \quad i = 0, 1, 2, 3 \tag{5.28}$$

Having the C^* coefficients expressed, the local polynomial is constructed as performed in equation 5.29, wherein an auxiliary variable $\gamma_j^*(x^*, y^*) = m_{1,j}^* + m_{2,j}^* x^* + m_{3,j}^* y^* + m_{4,j}^* x^* y^*$ is introduced.

$$\begin{aligned}
&\bar{\phi}(x^*, y^*) = \left(\sum_{j=1}^{4} m_{1,j}^* \bar{\phi}_{j-1}\right) + \left(\sum_{j=1}^{4} m_{2,j}^* \bar{\phi}_{j-1}\right) x^* + \left(\sum_{j=1}^{4} m_{3,j}^* \bar{\phi}_{j-1}\right) y^* + \left(\sum_{j=1}^{4} m_{4,j}^* \bar{\phi}_{j-1}\right) x^* y^* \\
\Leftrightarrow\ &\bar{\phi}(x^*, y^*) = \sum_{j=1}^{4} \left(m_{1,j}^* + m_{2,j}^* x^* + m_{3,j}^* y^* + m_{4,j}^* x^* y^*\right) \bar{\phi}_{j-1} \\
\Leftrightarrow\ &\bar{\phi}(x^*, y^*) = \sum_{j=1}^{4} \gamma_j^* \ \bar{\phi}_{j-1}
\end{aligned} \tag{5.29}$$

For example, to express point ϕ_1^+ as a linear combination of the points of the corresponding superfluous block, one only needs to substitute its local coordinates (x^*, y^*) in equation 5.29 and take into account that the superfluous values $\bar{\phi}_{j-1}$ are numbered according to figure 5.6a.

Applying equation 5.29 to all four points along the normal direction with their respective superfluous blocks, one is finally able to express the normal derivative jump, as written in equation 5.30.

$$\begin{aligned}
\sigma = {} & m_{21}^+ \phi_{sp}^+ + m_{22}^+ \left(\sum_{j=1}^{4} \gamma_j^* \ \bar{\phi}_{j-1}\right)_{1^+} + m_{23}^+ \left(\sum_{j=1}^{4} \gamma_j^* \ \bar{\phi}_{j-1}\right)_{2^+} + \\
& + (-m_{21}^-) \phi_{sp}^- + (m_{22}^-) \left(\sum_{j=1}^{4} \gamma_j^* \ \bar{\phi}_{j-1}\right)_{1^-} + (m_{23}^-) \left(\sum_{j=1}^{4} \gamma_j^* \ \bar{\phi}_{j-1}\right)_{2^-}
\end{aligned} \tag{5.30}$$

Rearranging equation 5.30, one arrives to equation 5.31, wherein the unknown superfluous points and normal derivative jump were put in the left-hand side, whereas the information concerning the solid point was put in the right. With this, filling matrices $\mathbf{M_C}$, $\mathbf{M_D}$ and vector $\mathbf{v_b}$ is straightforward.

$$\begin{aligned}
& m_{22}^+ \left(\sum_{j=1}^{4} \gamma_j^* \ \bar{\phi}_{j-1}\right)_{1^+} + m_{23}^+ \left(\sum_{j=1}^{4} \gamma_j^* \ \bar{\phi}_{j-1}\right)_{2^+} + (m_{22}^-) \left(\sum_{j=1}^{4} \gamma_j^* \ \bar{\phi}_{j-1}\right)_{1^-} + \\
& + (m_{23}^-) \left(\sum_{j=1}^{4} \gamma_j^* \ \bar{\phi}_{j-1}\right)_{2^-} + (-1)\sigma = -\left(m_{21}^+ \phi_{sp}^+ - m_{21}^- \phi_{sp}^-\right)
\end{aligned} \tag{5.31}$$

5.5 Smearing Correction

In this subchapter, two distinct strategies are presented to perform the smearing correction in the transition region. The first one, presented in section 5.5.1, corresponds to the two-dimensional generalisation of the correction strategy implemented in the one-dimensional case that obtained the best results. The second one, presented in section 5.5.2, consists of correcting the values inside the transition region by using independent corrections along the each Cartesian direction.

5.5.1 Correction with a Direct Method

In this correction procedure, the best one-dimensional approach, that made use of the solid point and two superfluous points, is generalised. The one-dimensional implementation consisted of generating a quadratic polynomial with those three points, using a direct method. As such, the natural two-dimensional generalisation of that correction procedure is to consider a direct method acting on a 3×3 block of points, with one of the normal levels corresponding to three solid points.

Notice that using a 3×3 block of points is exactly the same that is being used to compute the normal derivative jump, if the direct method is employed. Since in the corrective step one is dealing with function values and not first derivatives, performing the correction with a direct method involving a 3×3 block constitutes indeed a third-order accurate method. However, as it was referred in the end of section 3.4.3 concerning the one-dimensional case, this third-order accuracy should never lead to third-order accurate transition corrected values, since it is expected to act over second-order superfluous results. The important is to ensure that the corrective step is performed with, at least, second-order accuracy. As seen in the one-dimensional results, the third-order correction revealed to lead to a lower absolute error in the transition region, but the global decay continued to occur with second-order accuracy.

In the end of the execution, for each point inside the transition region, one determines its closest solid point, around which a polynomial expansion is performed, so that a new field value is computed for the transition point. The polynomial used in this strategy is similar to the one in equation 5.4, taking $n_C = 9$ and selecting the first nine terms from table 5.1, that is, the yellow and orange terms. The polynomials used in this strategy (one for each interface side, $+$ and $-$) are written in equation 5.32.

$$\phi^{\pm}(\eta,\xi) = C_0^{\pm} + C_1^{\pm}\eta + C_2^{\pm}\xi + C_3^{\pm}\eta\xi + C_4^{\pm}\eta^2 + C_5^{\pm}\eta^2\xi + C_6^{\pm}\xi^2 + C_7^{\pm}\eta\xi^2 + C_8^{\pm}\eta^2\xi^2 \tag{5.32}$$

The sample of points to compose the 3×3 block is selected in the same way that in the direct method, as illustrated in figure 5.2, being the sample points named as $\phi_0^{\pm}$, $\phi_1^{\pm}$, ..., $\phi_8^{\pm}$. Expressing the value of polynomial 5.32 at each of the sample points, one reaches equation 5.33. Once more, for both sides of the interface, $\phi_0^{\pm}$ were conventioned to be solid point values of the central solid point, whereas $\phi_1^{\pm}$ and $\phi_2^{\pm}$ correspond to the previous and next solid point values, respectively.

$$\begin{pmatrix} 1 & 0 & 0 & 0 & \dots & 0 \\ 1 & \eta_1^{\pm} & \xi_1^{\pm} & \eta_1^{\pm}\xi_1^{\pm} & \dots & (\eta_1^{\pm})^2(\xi_1^{\pm})^2 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \eta_8^{\pm} & \xi_8^{\pm} & \eta_8^{\pm}\xi_8^{\pm} & \dots & (\eta_8^{\pm})^2(\xi_8^{\pm})^2 \end{pmatrix} \begin{pmatrix} C_0^{\pm} \\ C_1^{\pm} \\ \vdots \\ C_8^{\pm} \end{pmatrix} = \begin{pmatrix} \phi_0^{\pm} \\ \phi_1^{\pm} \\ \vdots \\ \phi_8^{\pm} \end{pmatrix} \tag{5.33}$$

Inverting the square matrix in equation 5.33, one arrives to the values of constants $C_i^{\pm}$. Notice that the corrective step is performed after the execution, when the values of $\phi_3^{\pm}$, $\phi_1^{\pm}$, ..., $\phi_8^{\pm}$ are already determined. Therefore, constants $C_i^{\pm}$ are completely computed and they are given a numeric value. As such, the polynomial of equation 5.32 is totally defined and the correction of the transition point is simply performed by substituting its local coordinates (η,ξ) in the polynomial expression.

5.5.2 Correction along Each Cartesian Direction

In this procedure, the correction of points in the transition region is made by using simple extrapolations along each Cartesian direction, as exemplified in figure 5.7, in the case of the exterior subdomain. For each point inside the transition region, the closest superfluous points along x and along y belonging to the same subdomain are identified. If the superfluous point identified along x is closer than the one along y (case in blue), the correction is computed with a simple extrapolation along x using two superfluous points. If the superfluous point identified along y is the closest one (case in orange), the correction is computed with a similar extrapolation along y. If both the closest x and y points are located at the same distance from the transition point (case in green), both extrapolations are computed and the correction value is assumed to be their average. If a transition point occurs to fall exactly at the interface, this procedure is performed for both subdomains and the final correction value is taken as their mean.

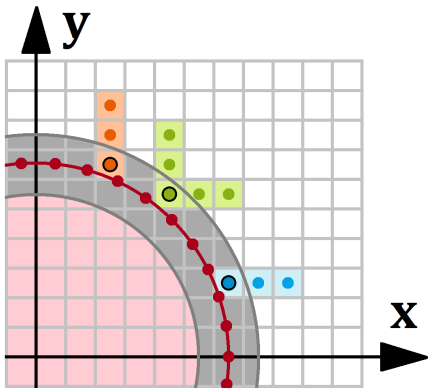


Figure 5.7: Illustration of the Cartesian correction along each direction, for the exterior subdomain.

Considering the blue case, the second superfluous point to be used in the extrapolation is considered to be the one immediately in the right, that is, the second closest point along x. Similarly, in the orange case, the second point that is invoked in the extrapolation is the second closest along y, that corresponds in this case to the one immediately above. In the green case, an identical reasoning is applied to identify the second points entering the extrapolation along x and the extrapolation along y.

Notice that all the points are aligned, the distance between the two superfluous points is equal to the grid size and the distance between the transition point and the closest superfluous point is a multiple of the grid size. As such, the extrapolation formula will always present the same structure. Naming the closest and second closest superfluous points along a given direction as $\bar{\phi}_1^{sup}$ and $\bar{\phi}_2^{sup}$ and naming the number of grid size units separating the transition point and the closest superfluous points by ν, equations 5.34 arise. These expressions are identical to the general formula in equation 3.26.

$$\bar{\phi}_x^{new} = (\nu_x + 1)\bar{\phi}_{1_x}^{sup} - \nu_x \bar{\phi}_{2_x}^{sup} \tag{5.34a}$$

$$\bar{\phi}_y^{new} = (\nu_y + 1)\bar{\phi}_{1_y}^{sup} - \nu_y \bar{\phi}_{2_y}^{sup} \tag{5.34b}$$

Summarising, the blue correction is performed using equation 5.34a, the orange correction uses equation 5.34b and the green correction applies both equations taking the average of their results.

Chapter 6

2D Results

In this chapter, firstly, one presents the results concerning the original two-dimensional formulation of the IL Method, under finite differences and finite volume. With this, one aims to replicate the finite differences results presented by Eldredge in article [1], as well as validate the behaviour of the derived finite volume approach. Then, the numerical results obtained by applying the two-dimensional improvement strategies deduced in chapter 5 are presented. Once more, all the additional second equations that were derived are used together with the masked Poisson equation in a finite volume framework.

In subchapter 6.1, general considerations are made concerning the geometry, error quantities to be tracked and the discontinuous function with discontinuous normal derivative that will be used to perform the numerical tests. In subchapter 6.2, the original formulation of the IL Method is studied, under both finite differences and finite volume frameworks. In subchapter 6.3, the Direct Method derived in subchapter 5.2 is implemented. In subchapter 6.4, the Least Squares Method presented in subchapter 5.3 is studied, imposing or not the value of the solid point. Then, in subchapter 6.5, the 1D-Inspired Method derived in subchapter 5.4 is implemented. In subchapter 6.6, the best 2D strategies are compared. Finally, in subchapter 6.7, the 2D corrective procedures are applied, in order to determine whether they are able to recover interface sharpness and to propagate superfluous decays to the transition region.

6.1 General Considerations

In order to study a two-dimensional problem, the geometry must be set. The square computational domain was considered to have a side length $L = 2$. The solid body was particularised into a circle centred at the origin of the Cartesian referential, with radius $R = 0.5$. This geometry is similar to the one considered in article [1], so that the numerical results may be compared. As such, the transition region, $\mathbb{T}$, over which the discrete delta functions act, is defined according to equation 6.1.

$$\mathbb{T} = \{\mathbf{x} \in \Omega : R - \frac{w}{2}h < \|\mathbf{x}\| < R + \frac{w}{2}h\} \tag{6.1}$$

The superfluous region, that is out of the direct influence of the discrete delta functions is given by the union of the interior and exterior subregions defined in equation 6.2, that is $\mathbb{S} = \mathbb{S}^- \cup \mathbb{S}^+$ respectively.

$$\begin{cases} \mathbb{S}^- = \{\mathbf{x} \in \Omega : \|\mathbf{x}\| \leq R - \frac{w}{2}h\} \\ \mathbb{S}^+ = \{\mathbf{x} \in \Omega : \|\mathbf{x}\| \geq R + \frac{w}{2}h\} \end{cases} \tag{6.2}$$

The discontinuous function with discontinuous normal derivative across the interface that will be studied throughout this chapter is also the same as in article [1]. This two-dimensional function will be shortly referred to as function F5. Its analytical expression is provided in equation 6.3 and its graphical representation is displayed in figure 6.1. As a remark, notice that this function presents a null masked Laplacian, $\overline{\nabla^2\phi}$, since both interior and exterior Laplacian terms are zero, $\nabla^2\phi^+ = 0 = \nabla^2\phi^-$. Therefore, there is no concern about computing the masked Laplacian term during the execution, that would need to invoke the computation of the Heaviside functions as well, keeping the right-hand side $\mathbf{v_a}$ simpler.

$$\bar{\phi}(\mathbf{x}) = \begin{cases} e^x \cos(y) & , \quad \mathbf{x} \in \Omega^- \\ 0 & , \quad \mathbf{x} \in \Omega^+ \end{cases} \tag{6.3}$$

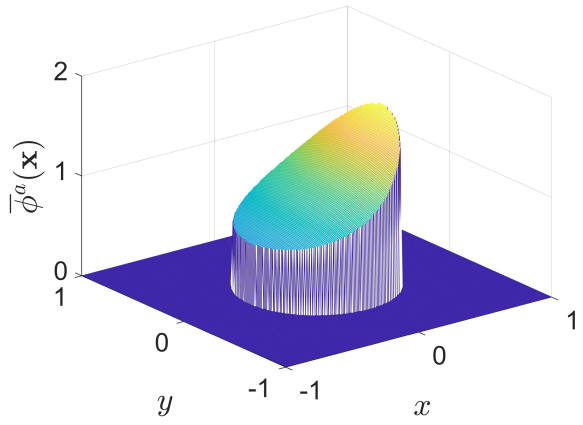


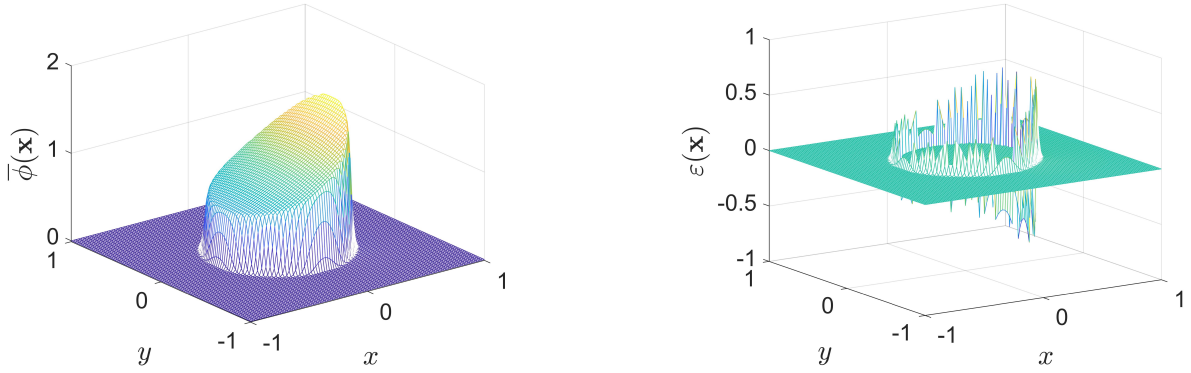
Figure 6.1: Graphical representation of function F5, used throughout this chapter.

The error field continues to be defined as in equation 4.1, with all the variables being now vectors $N \times 1$ with $N = n^2$. Similarly, the definitions of the error norms in equations 4.2a, 4.2b and 4.2c still hold, with N properly computed. The expression for the order of decay, in equation 4.3, continues valid.

Notice that, in this two-dimensional problem with a circular body, there is no need to care about the representativeness of the numbers of cells that are used to evaluate the decays, since the p solid points along the circular surface will fall anywhere relative to the grid. However, this also prevents choosing cells where the interface falls in the same relative position to evaluate the decay, as one did in the one-dimensional case. As such, when computing the order of decay, one may be catching some unmeaningful oscillatory behaviour and not the straight line trend around which it occurs (as one will see, this occurs in the maximum superfluous error decay of some strategies). To ensure that the straight line trend is caught, one must choose points in the decay graph sufficiently distant from each other. For that reason, the decays computed throughout this chapter use the least and the most refined grids of the set $N \in \{50^2; 100^2; 200^2\}$. The grid in between is used to check if there is indeed a straight line trend.

6.2 Original Formulation

In this subchapter, the original formulation of the Immersed Layers Method is implemented, first in finite differences and then in finite volume. Starting with the finite differences framework, the numerical solution and the error field obtained for function F5 are exemplified in figure 6.2, when considering $N = 100^2$ cells and using the discrete delta function δ_3^*. The decays verified for all the discrete delta functions, for all the error parameters in each region, are presented in table 6.1.



(a) Numerical field.

(b) Error field.

Figure 6.2: Results for function F5 under a finite differences approach, using δ_3^* and $N = 100^2$ cells.

Table 6.1: Error decay for function F5, in the original IL formulation, using finite differences.

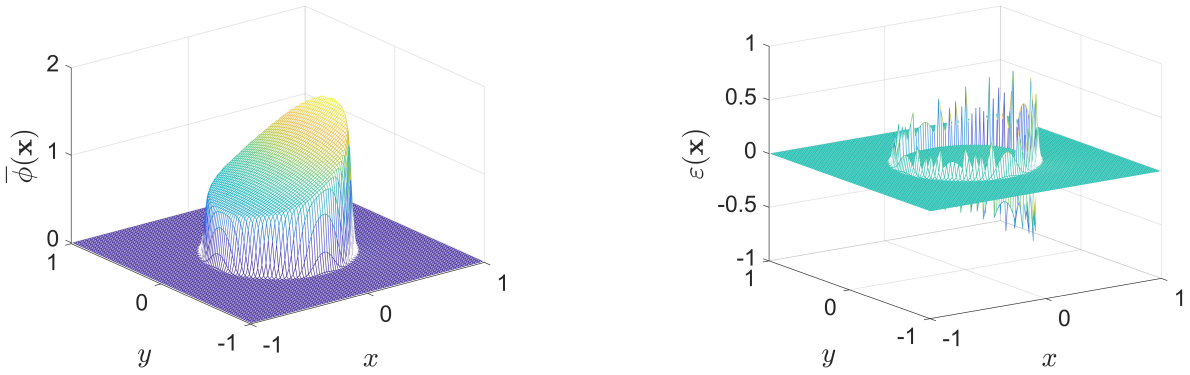
δ_h	$\bar{\varepsilon}$	$\|\varepsilon\|_2$	ε_{max}	$\bar{\varepsilon}^{Sup}$	$\|\varepsilon\|_2^{Sup}$	ε_{max}^{Sup}	$\bar{\varepsilon}^{Trans}$	$\|\varepsilon\|_2^{Trans}$	$\varepsilon_{max}^{Trans}$
δ_3^*	0.9452	0.4769	N.D.	0.9128	0.9257	0.9209	N.D.	N.D.	N.D.
δ_{2002}	0.9859	0.4903	N.D.	0.9925	0.9721	0.3453	N.D.	N.D.	N.D.
δ_{2103}	0.9998	0.4774	N.D.	1.1460	1.0648	0.2919	N.D.	N.D.	N.D.
δ_{4206}	0.9905	0.4899	N.D.	1.0446	1.0173	0.7524	N.D.	N.D.	N.D.

As figure 6.2 shows, the numerical interface is now smeared instead of sharp, as it was already expected to occur. The error field presents significant peaks near the interface, as already occurred in the one-dimensional results. As table 6.1 shows, similarly to what Eldredge [1] presented, one obtained first-order accuracy in the superfluous decay of the L^2 error norm, when using δ_3^*. Note that Eldredge only analyses the region corresponding to what one defined as $\mathbb{S}^+$, that is, the region of the exterior subdomain that is not directly affected by the discrete delta function, since the focus is the computation of the fluid field. However, to keep the transition region $\mathbb{T}$ isolated, in this work one always studies the decays in $\mathbb{T}$ and in $\mathbb{S} = \Omega \setminus \mathbb{T}$. Since $\mathbb{S}^-$ has the same nature of $\mathbb{S}^+$, being out of the direct effect of the discrete delta functions, the decays in $\mathbb{S}^-$, in $\mathbb{S}^+$ and in $\mathbb{S} = \mathbb{S}^- \cup \mathbb{S}^+$ were naturally found to be identical.

Concerning the global error parameters, the results are identical to the one-dimensional ones obtained with a discontinuous function with discontinuous normal derivative: the mean error decays with first-order accuracy, the L^2 norm decays with half order and the maximum error does not decay. In the transition region, a non-decay is also detected. However, in the superfluous region some differences

arise concerning the maximum error decay, with δ_{4206} presenting a slightly lower than first-order decay (0.7524) but with δ_{2002} and δ_{2103} presenting a significant penalisation (0.3453 and 0.2919, respectively). The discrete delta function δ_3^* preserves the first-order decay in the superfluous maximum error in 2D.

The numerical solution and error field obtained using a finite volume framework is displayed in figure 6.3, considering $N = 100^2$ cells and the discrete delta function δ_3^*. The decays resulting from the use of all the discrete delta functions, for all the error parameters in each region, are presented in table 6.2.



(a) Numerical field. (b) Error field.

Figure 6.3: Results for function F5 under a finite volume approach, using δ_3^* and $N = 100^2$ cells.

Table 6.2: Error decay for function F5, in the original IL formulation, using finite volume.

δ_h	$\bar{\varepsilon}$	$\|\varepsilon\|_2$	ε_{max}	$\bar{\varepsilon}^{Sup}$	$\|\varepsilon\|_2^{Sup}$	ε_{max}^{Sup}	$\bar{\varepsilon}^{Trans}$	$\|\varepsilon\|_2^{Trans}$	$\varepsilon_{max}^{Trans}$
δ_3^*	0.9841	0.5254	N.D.	0.9345	0.9338	0.8581	N.D.	N.D.	N.D.
δ_{2002}	0.9683	0.5212	N.D.	0.9350	0.9112	0.1647	N.D.	N.D.	N.D.
δ_{2103}	0.9862	0.5256	N.D.	0.9592	0.9452	0.3645	N.D.	N.D.	N.D.
δ_{4206}	0.9844	0.5159	N.D.	0.9434	0.9317	0.5081	N.D.	N.D.	N.D.

The finite volume numerical solution and error field are visually identical to the ones obtained in finite differences. Concerning table 6.2, the decays in the entire domain and in the transition region are similar to the ones of table 6.1. However, the penalisation verified in the superfluous maximum decay is higher for δ_{2002} and δ_{4206} (with decays 0.1647 and 0.5081, respectively). The penalisation verified in δ_{2103} is less accentuated than it was in the finite differences framework, but the decay value is still one of the lowest (0.3645). Discrete delta function δ_3^* presents a slight penalisation as well, but not comparable to the other ones (0.8581). Indeed, this severe penalisation in the maximum superfluous decay of δ_{2002} and δ_{2103} is already a symptom of the bad behaviour that these discrete delta functions will display in all the upcoming improvements strategies. The results presented in tables 6.1 and 6.2 reveal that, in the original two-dimensional formulation, discrete delta function δ_3^* is preferable to others.

When comparing the error values obtained with δ_3^*, one concludes that there is no significant difference between the finite differences version (IL-FD) and the finite volume (IL-FV) one. To prove it, figure 6.4 presents the superfluous decay graphs obtained with δ_3^* for both frameworks, where it is visible that they are practically superimposed. Considering the intermediate grid, for example, the mean value and the L^2 norm of the superfluous error differ about 1% and the maximum superfluous error differs 5%.

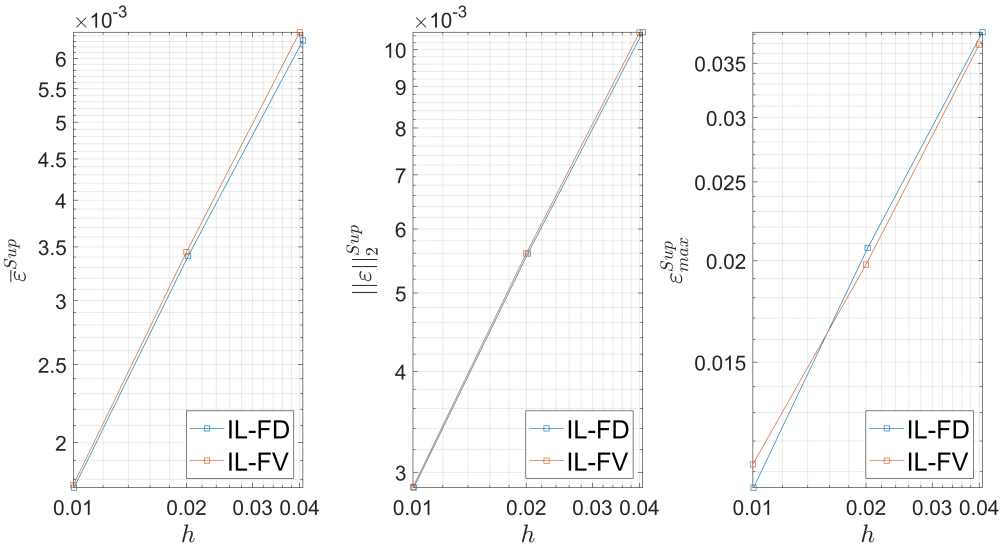


Figure 6.4: Superfluous comparison between the IL-FD and the IL-FV for discrete delta function δ_3^*.

Therefore, from now on, when comparing the superfluous decays of a new derived strategy with the original method, only one of these formulations needs to be represented, since the other one is identical.

6.3 Direct Method

In this subchapter, the results concerning the implementation of the Direct Method derived in subchapter 5.2 are presented. Since one is pursuing second-order accuracy in the computation of the first derivatives, the natural choice would be to use a 3×3 block of points, as explained during the derivation of the method. However, since in the one-dimensional results the use of a first-order unilateral derivative expression did not prevent obtaining second-order superfluous results, one will test a 3×2 block of points as well. The first normal layer is composed of the solid point under study, the previous and the next one, whereas the second normal layer corresponds to the three closest points belonging to the first superfluous layer (recall figure 5.2). The results concerning the decay of the several error quantities for the Direct Method are presented in table 6.3. For the sake of compactness, the version using 3×2 blocks is coded as DM-32, whereas the version using 3×3 blocks is coded as DM-33.

Table 6.3: Decays for function F5, with the Direct Method for 3×2 and 3×3 blocks.

		$\bar{\varepsilon}$	$\|\varepsilon\|_2$	ε_{max}	$\bar{\varepsilon}^{Sup}$	$\|\varepsilon\|_2^{Sup}$	ε_{max}^{Sup}	$\bar{\varepsilon}^{Trans}$	$\|\varepsilon\|_2^{Trans}$	$\varepsilon_{max}^{Trans}$
DM-32	δ_3^*	1.0195	0.5145	N.D.	1.8625	1.8154	0.9521	N.D.	N.D.	N.D.
	δ_{2002}	0.9502	0.5197	N.D.	0.3482	0.4517	N.D.	N.D.	N.D.	N.D.
	δ_{2103}	1.0000	0.5172	N.D.	0.6462	0.6484	N.D.	N.D.	N.D.	N.D.
	δ_{4206}	1.0176	0.5165	N.D.	1.7519	1.5627	0.7658	N.D.	N.D.	N.D.
DM-33	δ_3^*	1.0134	0.5149	N.D.	1.8348	1.6889	0.3650	N.D.	N.D.	N.D.
	δ_{2002}	0.9585	0.5190	N.D.	0.4156	0.4793	N.D.	N.D.	N.D.	N.D.
	δ_{2103}	1.0416	0.5178	N.D.	1.6121	0.9798	N.D.	N.D.	N.D.	N.D.
	δ_{4206}	1.0055	0.5170	N.D.	1.1392	0.9834	0.2984	N.D.	N.D.	N.D.

Globally, table 6.3 shows that all the discrete delta functions present a first-order decay of the mean error and a half-order decay of the L^2 norm. Transition error quantities do not decay.

Consider now the superfluous decays of δ_3^* and δ_{4206}. Function δ_3^* presents the best superfluous results (highlighted in green), occurring for the 3×2 block of points. Indeed, the superfluous results of this discrete delta function correspond to a first-order decay of the maximum error (similarly to what occurred in the original IL formulation), but decays of order around 1.8 for the mean value and L^2 norm of the superfluous error. So, the results highlighted in green for δ_3^* using 3×2 blocks present clear advantage, in terms of order of decay, when compared to the original IL formulation. Concerning δ_{4206}, advantage is also found when compared to the original formulation, with the maximum superfluous error presenting a similar decay, but the superfluous mean value and L^2 norm decaying above 1.5.

In the original IL formulation, delta functions δ_{2002} and δ_{2103} presented a very low decay in the maximum superfluous error. Now, in the Direct Method, these functions lead to a non-decay of that parameter, as shown in table 6.3. Remind that one is looking for good superfluous decays, to lately extend them to the transition region. Until now, the 2D results always revealed a poor performance of discrete delta functions δ_{2002} and δ_{2103}. Indeed, the non-decay of the maximum superfluous error (or a decay of just a few decimals) was also found to occur for these two discrete delta functions in the upcoming methods. So, henceforward, discrete delta functions δ_{2002} and δ_{2103} are dropped from the 2D analysis.

Excluding the problematic delta functions δ_{2002} and δ_{2103}, one should note that δ_3^* and δ_{4206} present worst results for 3×3 blocks than for 3×2 blocks. As explained in subchapter 5.2, the polynomial terms used in the Direct Method are computed by selecting the same number of sample points, whose geometrical layout must be related with the terms selected to incorporate the polynomial, in order to achieve a given order of accuracy. On the other hand, the first normal level (corresponding to the solid points) obeys to the interface orientation, whereas the other normal levels (coming from the superfluous region) are disposed according to the Cartesian coordinate system. As such, it is not possible to obtain perfect 3×2 or 3×3 blocks of points. Since the objective is to compute a normal derivative, that is a quantity along one of the axis of the local coordinate system, the solid points present an adequate layout, faithfully translating three distinct and equally spaced tangential levels. Moreover, note that the information coming from solid sample points is analytical, whereas the computation of the superfluous sample points comes from the numerical solving process. With all this said, given that 3×2 blocks are subjected to a stronger influence of their solid points, one may identify two reasons why their results are better: strongly dependence on points that present an adequate layout (the solid points); and less exposure to field values arising from the numerical process, relying more strongly on analytical information.

Besides comparing the orders of decay of the several error quantities with the original IL formulation, it is also important to analyse their value. The comparison between the original IL formulation implemented under a FV framework is compared with the DM with 3×2 blocks in figure 6.5, using δ_3^*.

As presented in figure 6.5, the best results achieved with the Direct Method, using 3×2 blocks and δ_3^*, not only present better orders of decay than the original IL formulation (recall tables 6.2 and 6.3), but also a smaller error. For the most refined mesh, the mean value and L^2 norm of the superfluous error are around two orders of magnitude lower. The difference in the maximum superfluous error is around

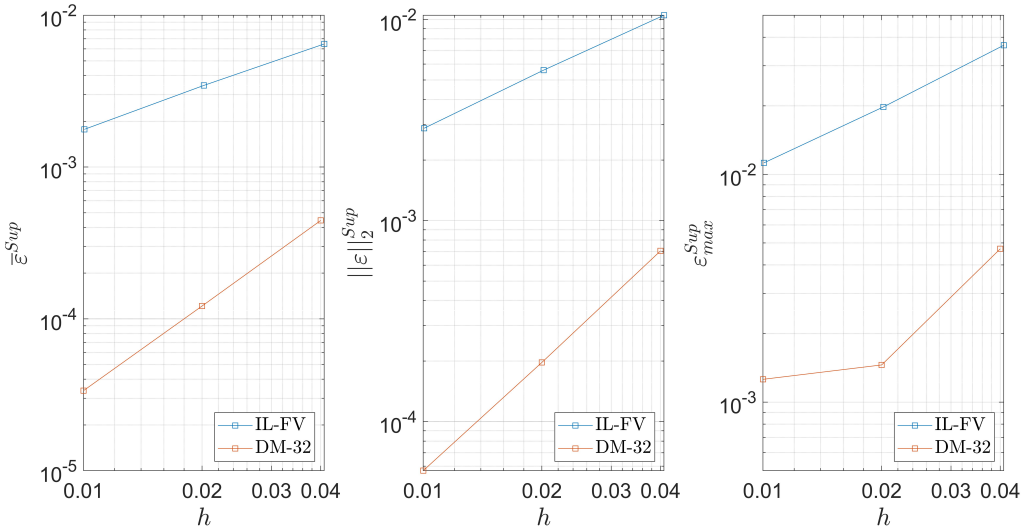


Figure 6.5: Superfluous comparison between IL-FV and method DM-32, for δ_3^*.

one order of magnitude. So, method DM-32 is better than the original formulation (IL-FD and IL-FV).

Again, using a first-order accurate computation of the normal derivative jump did not revert the superfluous results to first-order accuracy, in line with what Leveque and Li [9] state, regarding the inability of first-order decays in a set of lower dimension to contaminate second-order decays. However, an exact second-order accuracy of 2.0 was not achieved in this 2D implementation, what can be explained by the use of discrete delta functions to perform the regularisation operation in the masked Poisson equation.

In article [18], a very recent work by Cola *et al.*, a simple Poisson problem was studied, with just one singularity in the domain. In that problem, no interpolation operation relying on discrete delta functions was present and a discrete delta function was only used to regularise the only singularity appearing in the domain. In 1D, they were able to achieve second-order decays for some discrete delta functions. However, in 2D, despite the huge simplicity of the problem (just one singularity and not an interface covered of them), the results reverted to first-order accuracy for all the discrete delta functions, what they referred to as "a problem that has been known in literature for decades". As such, although the additional second equation one adds to the IL method may be theoretically strong enough to lead to second-order accuracy, one cannot forget the well-known and reported problems caused by the use of discrete delta functions in 2D regularisations. So, given this problem "known for decades", any two-dimensional accuracy improvement over first-order should be interpreted as a very positive result.

6.4 Least Squares Method

In this section, the Least Squares Method, derived in subchapter 5.3, is implemented and studied, as an alternative to the Direct Method. Similarly to what was verified in the original IL formulation and in the Direct Method, the error quantities obtained with the Least Squares Method do not decay in the transition region, nor the global maximum error. Moreover, the global mean error always decays with first-order accuracy and the global L^2 norm displays a half-order decay. As in the other methods, only

superfluous decays varied with the parameters to be set. Thus, only superfluous decays are presented.

Notice that the LSM has several degrees of freedom to define. First, one must decide the number of solid points to be used in the formulation. Here, two implementations are considered: using the solid point under study alone (LSM1), or using it together with the previous and next one (LSM3). After that, one may impose the value at the solid point under study directly on the least squares polynomial or not; then, one must decide whether to use weights or not; finally, one must set the elliptical cloud dimensions.

Besides all these parameters, one also has the number of coefficients used in the LS polynomial. In section 6.4.1, three terms are used in the LS polynomial (the yellow ones from table 5.2), whereas in section 6.4.2 six terms are included (the yellow and orange ones from table 5.2). Keep in mind that the exclusive use of the yellow terms corresponds theoretically to compute the first derivatives with first-order accuracy. However, the 1D results proved numerically that the statement made by Leveque and Li [9] indeed verifies (first-order exclusively occurring in a set of lower dimension is not able to contaminate a second-order decay). Moreover, the results for the Direct Method also showed that a reversion to first-order does not occur (being 3×2 blocks even better than the formal second-order formulation with a 3×3 blocks, in terms of order of decay). That is the reason why one tests the yellow terms alone.

6.4.1 Using Three Polynomial Terms

The superfluous decays concerning the implementation considering the solid point under study alone (LSM1) are presented in table 6.4, whereas the ones obtained when also using its previous and next neighbours (LSM3) are shown in table 6.5. The cases wherein the assignment of weights was turned off are identified with W0, whereas cases using weights are identified with W1. When the imposition of the solid point under study was turned off the results appear identified with I0, or with I1 if the imposition was used. Several elliptical clouds with semiaxes $s_\eta = Zh$ and $s_\xi = (\frac{w}{2} + Z)h$ were considered, $Z \in \mathbb{N}$.

Table 6.4: Superfluous decays using method LSM1 with three terms, for $s_\eta = Zh$, $s_\xi = (\frac{w}{2} + Z)h$.

			$Z=3$			$Z=4$			$Z=5$		
			$\bar{\varepsilon}^{Sup}$	$\|\varepsilon\|_2^{Sup}$	ε_{max}^{Sup}	$\bar{\varepsilon}^{Sup}$	$\|\varepsilon\|_2^{Sup}$	ε_{max}^{Sup}	$\bar{\varepsilon}^{Sup}$	$\|\varepsilon\|_2^{Sup}$	ε_{max}^{Sup}
I0	W0	δ_3^*	1.6617	1.6412	1.4702	1.4462	1.4229	1.3282	1.2126	1.1919	1.1136
		δ_{4206}	1.5295	1.4860	1.0238	1.3741	1.3460	1.0293	1.1768	1.1522	0.9353
	W1	δ_3^*	1.6905	1.6689	1.4167	1.4889	1.4629	1.3517	1.2387	1.2141	1.1153
		δ_{4206}	1.5729	1.5168	1.0057	1.4440	1.4115	1.0420	1.2466	1.2211	0.9857
I1	W0	δ_3^*	1.8360	1.8103	1.18514	1.8023	1.7780	1.2926	1.7845	1.7638	1.4094
		δ_{4206}	1.6464	1.5528	0.9002	1.6339	1.5596	0.9627	1.6286	1.5658	1.0207
	W1	δ_3^*	1.8644	1.8338	1.1193	1.8439	1.8161	1.1969	1.8359	1.8123	1.2852
		δ_{4206}	1.6662	1.5553	0.8633	1.6640	1.5700	0.9112	1.6651	1.5834	0.9608

According to tables 6.4 and 6.5, the difference between assigning weights or not is reduced, being the decays with the use of weights slightly higher for the mean and the L^2 norm of the superfluous error, but slightly lower when considering the maximum superfluous error. On the other hand, the polynomial imposition of the solid point under study appears to have a more significant impact, being the results

Table 6.5: Superfluous decays using method LSM3 with three terms, for $s_\eta = Zh$, $s_\xi = (\frac{w}{2} + Z)h$.

			$Z=3$			$Z=4$			$Z=5$		
			$\bar{\varepsilon}^{Sup}$	$\|\varepsilon\|_2^{Sup}$	ε_{max}^{Sup}	$\bar{\varepsilon}^{Sup}$	$\|\varepsilon\|_2^{Sup}$	ε_{max}^{Sup}	$\bar{\varepsilon}^{Sup}$	$\|\varepsilon\|_2^{Sup}$	ε_{max}^{Sup}
I0	W0	δ_3^*	1.7588	1.7380	1.3476	1.6200	1.5976	1.4842	1.4736	1.4515	1.3690
		δ_{4206}	1.6012	1.5365	0.9770	1.5168	1.4774	1.0382	1.4050	1.3744	1.0351
	W1	δ_3^*	1.8550	1.8283	1.1698	1.8213	1.7969	1.2612	1.7994	1.7781	1.3637
		δ_{4206}	1.6737	1.5710	0.8932	1.6663	1.5798	0.9398	1.5847	1.6570	0.9956
I1	W0	δ_3^*	1.8367	1.8110	1.1856	1.8024	1.7781	1.2930	1.7846	1.7639	1.4094
		δ_{4206}	1.6474	1.5541	0.9016	1.6345	1.5601	0.9628	1.6288	1.5660	1.0209
	W1	δ_3^*	1.8680	1.8376	1.1217	1.8455	1.8180	1.2005	1.8378	1.8139	1.2859
		δ_{4206}	1.6750	1.5641	0.8708	1.6731	1.5777	0.9148	1.6710	1.5893	0.9664

with imposition better for the mean and the L^2 norm of the superfluous error, but slightly lower when considering the maximum superfluous error. As such, the maximum superfluous error appears to behave oppositely from the mean and the L^2 norm, concerning the effect of the use of weights and imposition. Regarding the increase in the elliptical cloud dimensions, most decays were found to reduce, though in table 6.5 some exceptions are verified for the superfluous maximum error.

The results achieving the highest decays in each table appear highlighted in green. Like in the Direct Method, the best results occur for discrete delta function δ_3^*, with the highest mean and L^2 superfluous decays occurring for I1 and W1 in the method LSM3, considering an elliptical cloud characterised by $Z = 3$. Although the maximum superfluous decay is lower in this case than when using δ_3^*, I0 and W0, it is still higher than first-order, existing still an advantage compared to the original formulation. Note that the values highlighted in green are very similar to the ones obtained in the best Direct Method case, with a slight advantage for these ones obtained with LSM3 using δ_3^*, I1, W1 and an elliptical cloud characterised by $Z = 3$. Notice that some other interesting results arise. Instead of looking for the highest decays with the single concern that the superfluous maximum error is still higher than first-order, if one was looking for a more homogeneous decay of all the error parameters, several cases would arise as a good compromise with all the quantities decaying with order around 1.5 (like the cases highlighted in grey). The numerical results appear to show that higher orders in the superfluous mean and L^2 norm are achieved at the expense of diminishing the decay obtained in the superfluous maximum error.

The comparison between the error arising from the original formulation of the IL method and the best results occurring for LSM3 with I1 and W1 and using δ_3^* is presented in figure 6.6. For the sake of compactness in the legend of figures, the case with the best results in this least squares method that uses three polynomial coefficients (the yellow coefficients of table 5.2) is coded as LSM-Y, standing for method LSM3 with I1, W1 and considering an elliptical cloud characterised by $Z = 3$. Later on, a comparison between the best results of all the two-dimensional strategies will be provided.

As figure 6.6 suggests, not only the orders of decay observed for method LSM-Y are higher than the ones verified for the original IL method (recall tables 6.2 and 6.5), but also the error values are indeed lower. As an example, for the most refined mesh, the mean value and the L^2 norm of the superfluous error are almost two orders of magnitude lower when using strategy LSM-Y. The difference

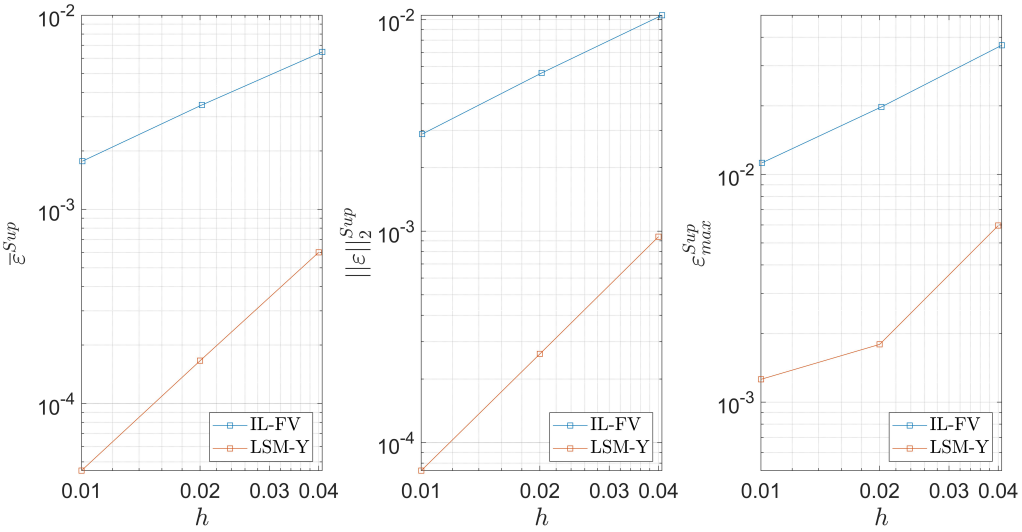


Figure 6.6: Superfluous comparison between IL-FV and method LSM-Y, for δ_3^*.

in the maximum superfluous error is around one order of magnitude. Thus, the differences are significant and one can state that method LSM-Y is better than the original formulation (IL-FD and IL-FV).

6.4.2 Using Six Polynomial Terms

In this section, the six terms theoretically needed to achieve second-order results are used (the ones in yellow and the ones in orange from table 5.2). The superfluous decays concerning the implementation with the solid point alone are presented in table 6.6, whereas the ones obtained when also using its previous and next neighbours are shown in table 6.7. The nomenclature used to identify the use of weights and imposition is similar to what was explained in the previous section.

Table 6.6: Superfluous decays using method LSM1 with six terms, for $s_\eta = Zh$, $s_\xi = (\frac{w}{2} + Z)h$.

			$Z=3$			$Z=4$			$Z=5$		
			$\bar{\varepsilon}^{Sup}$	$\|\varepsilon\|_2^{Sup}$	ε_{max}^{Sup}	$\bar{\varepsilon}^{Sup}$	$\|\varepsilon\|_2^{Sup}$	ε_{max}^{Sup}	$\bar{\varepsilon}^{Sup}$	$\|\varepsilon\|_2^{Sup}$	ε_{max}^{Sup}
I0	W0	δ_3^*	1.8682	1.7327	0.3949	1.9019	1.7637	0.4213	1.8389	1.7622	0.4694
		δ_{4206}	1.4927	1.1311	0.4275	1.6838	1.1696	0.4592	1.6621	1.1913	0.4605
	W1	δ_3^*	1.8431	1.7076	0.3940	1.8958	1.7552	0.4077	1.6978	1.6853	0.4457
		δ_{4206}	1.4715	1.1267	0.4317	1.6463	1.1582	0.4540	1.6377	1.1763	0.4415
I1	W0	δ_3^*	1.8497	1.7142	0.3977	1.8849	1.7503	0.4125	1.9088	1.7772	0.4317
		δ_{4206}	1.5047	1.1311	0.4324	1.6766	1.1572	0.4589	1.6978	1.1679	0.4631
	W1	δ_3^*	1.8074	1.6732	0.4024	1.8510	1.7159	0.4103	1.8786	1.7458	0.4215
		δ_{4206}	1.4777	1.1268	0.4365	1.6413	1.1502	0.4597	1.6608	1.1565	0.4617

According to tables 6.6 and 6.7, the strategy using six terms in the least squares polynomial reveals different behaviours from the one with three terms. For example, the superfluous mean and L^2 norm decays appear to decrease when weights are used. Regarding the imposition, it is not that easy to set a general rule that is always (or almost always) valid, since there would be a lot of exceptions. Overall, one also finds the highest mean and L^2 norm decays for the largest elliptical clouds, contrary to what was

Table 6.7: Superfluous decays using method LSM3 with six terms, for $s_\eta = Zh$, $s_\xi = (\frac{w}{2} + Z)h$.

			$Z=3$			$Z=4$			$Z=5$		
			$\bar{\varepsilon}^{Sup}$	$\|\varepsilon\|_2^{Sup}$	ε_{max}^{Sup}	$\bar{\varepsilon}^{Sup}$	$\|\varepsilon\|_2^{Sup}$	ε_{max}^{Sup}	$\bar{\varepsilon}^{Sup}$	$\|\varepsilon\|_2^{Sup}$	ε_{max}^{Sup}
I0	W0	δ_3^*	1.8610	1.7289	0.4089	1.9024	1.7701	0.4240	1.9294	1.8006	0.4564
		δ_{4206}	1.4817	1.1400	0.4389	1.6783	1.1677	0.4639	1.6834	1.1833	0.4657
	W1	δ_3^*	1.8085	1.6811	0.4218	1.8581	1.7286	0.4223	1.8868	1.7594	0.4329
		δ_{4206}	1.4615	1.1439	0.4500	1.6403	1.1597	0.4672	1.6579	1.1644	0.4654
I1	W0	δ_3^*	1.8500	1.7159	0.4024	1.8849	1.7507	0.4129	1.9090	1.7778	0.4324
		δ_{4206}	1.4812	1.1336	0.4357	1.6785	1.1575	0.4596	1.6975	1.1682	0.4631
	W1	δ_3^*	1.8063	1.6795	0.4243	1.8534	1.7223	0.4182	1.8796	1.7492	0.4257
		δ_{4206}	1.4579	1.1449	0.4510	1.6402	1.1585	0.4665	1.6597	1.1602	0.4632

verified using three polynomial terms. The best results from each table are again highlighted in green. Although the highest mean and L^2 norm decay values are higher in this strategy with six polynomial terms, the maximum superfluous error is not able display first-order accuracy. Since the objective of this work is to improve the original IL method, strategies in which some relevant error parameter performs worse must not be regarded. More precisely, one must not admit lower than first-order decays of the maximum superfluous error. Indeed, notice that none of the simulations with six terms led to maximum superfluous decays higher then half order. As such, the results from the strategy using only three polynomial terms should be considered better than the ones arising from the use of six terms.

Even so, one represents in figure 6.7 the comparison between the original IL method in finite volume and the best strategy with six terms, that is, LSM3 with W0 and I0, considering elliptical clouds with $Z = 5$ and using δ_3^*. For compactness, method LSM3 with W0, I0 and $Z = 5$ is coded as LSM-O, as it is the best method with six terms (using the orange terms together with the yellow ones).

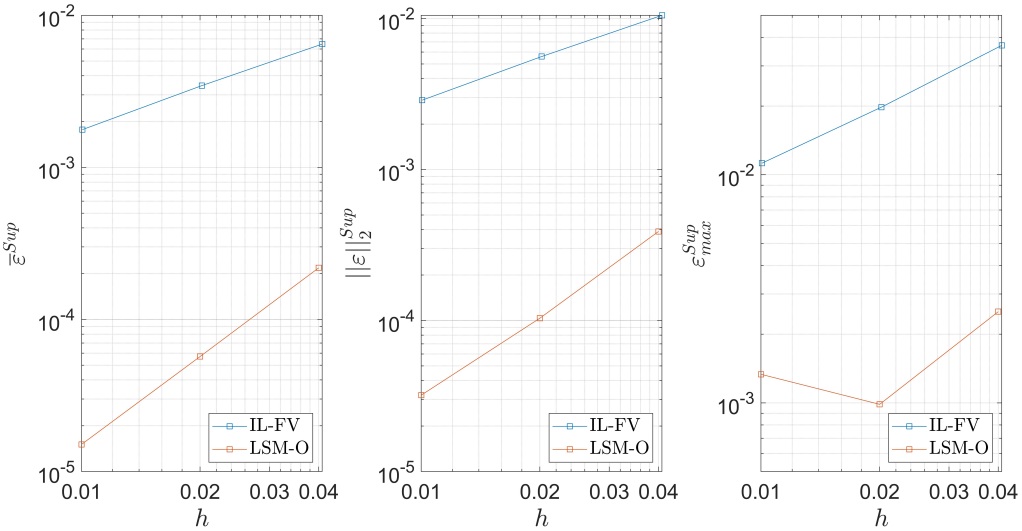


Figure 6.7: Superfluous comparison between IL-FV and method LSM-O, for δ_3^*.

As shown in figure 6.7, all the superfluous error quantities verified with strategy LSM-O using δ_3^* are lower than the ones verified in the original formulation. For the finest mesh, the mean value and L^2

norm of the superfluous error present two orders of magnitude of advantage, whereas the maximum superfluous error presents one order of magnitude of advantage. However, despite the oscillation, the maximum superfluous error decays with less than first-order, what does not constitute an improvement to the original formulation. As finest grids are considered, if these decay values continue holding, the original formulation is expected to eventually surpass method LSM-O. Therefore, strategy LSM-Y must be considered the best one coming from the implementation of the least squares formulation.

6.5 1D-Inspired Method

In this subchapter, the 1D-Inspired Method derived in subchapter 5.4, intended to replicate as faithfully as possible the one-dimensional reasoning, is studied. During the derivation of this method, two parameters were let free: the location of the first points along the normal direction and the dimension of the superfluous blocks. Given the restrictions of equation 5.21, one considered distances $\xi_1^+ = (\frac{w}{2}+1.5)h$ for 2×2 blocks, $\xi_1^+ = (\frac{w}{2}+2.5)h$ for 3×3 blocks and $\xi_1^+ = (\frac{w}{2}+3.0)h$ for 4×4 superfluous blocks.

The decays obtained in the superfluous region are presented in table 6.8. Given the geometry of the problem, the use of 4×4 blocks cannot be combined with δ_{4206} in the grid with $N = 50^2$ cells, since the most exterior block would not be fully contained in the computational domain. So, exceptionally, the decays of those three entries of table 6.8 were computed considering $N \in \{75^2; 100^2; 200^2\}$.

Table 6.8: Superfluous decays using the 1D-Inspired Method, considering $Z \times Z$ superfluous blocks.

	$Z=2$			$Z=3$			$Z=4$		
	$\bar{\varepsilon}^{Sup}$	$\|\varepsilon\|_2^{Sup}$	ε_{max}^{Sup}	$\bar{\varepsilon}^{Sup}$	$\|\varepsilon\|_2^{Sup}$	ε_{max}^{Sup}	$\bar{\varepsilon}^{Sup}$	$\|\varepsilon\|_2^{Sup}$	ε_{max}^{Sup}
δ_3^*	1.9407	1.8079	0.4490	1.9861	1.8702	0.5407	2.0107	1.9010	0.5891
δ_{4206}	1.7698	1.2110	0.4791	1.8072	1.2638	0.5117	1.7982	1.1292	0.4644

As presented in table 6.8, the discrete delta function δ_3^* is the one presenting the best results, reaching a true second-order decay for the superfluous mean error (2.0107) and a very close to second-order decay of the superfluous L^2 norm (1.9010), for 4×4 superfluous blocks. However, the superfluous maximum error presents a decay that is lower than first-order (0.5891), what is worse than the original IL formulation. Notice that the results improve with the increase of the dimension of the superfluous block from 2×2 to 3×3, but practically stabilise from 3×3 to 4×4.

Besides comparing the orders of decay of this best result of this strategy with the original IL formulation, one must compare the error values, as presented in figure 6.8, using δ_3^*. For compactness, code 1DIM-4 stands for the use of a 1DIM with 4×4 blocks.

Again, the improved method presents an advantage of around two orders of magnitude for the mean value and L^2 norm of the superfluous error and one order of magnitude for the maximum value. However, once more, given the higher decay of the maximum superfluous error in the original formulation, this advantage of one order magnitude will decrease (and eventually vanish) as one considers finest and finest grids.

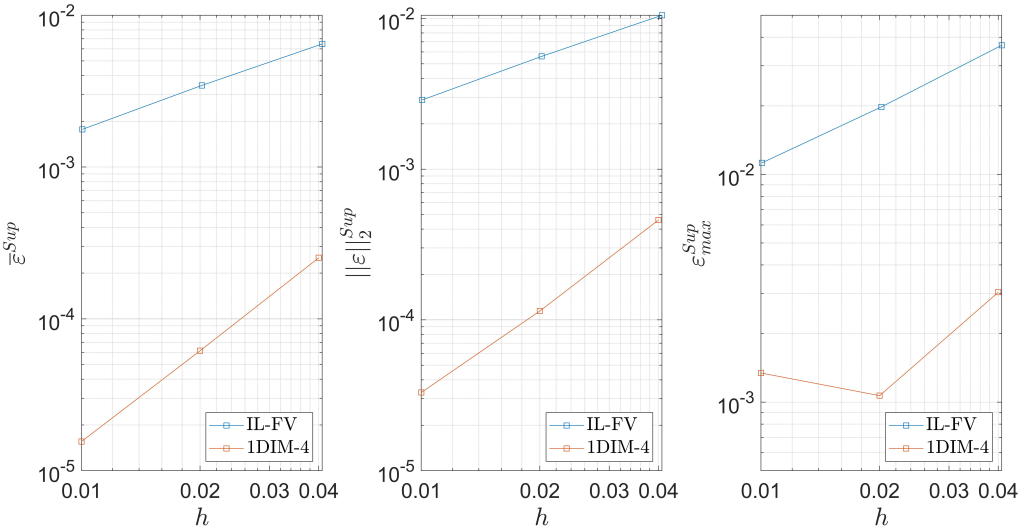


Figure 6.8: Superfluous comparison between IL-FV and method 1DIM for 4×4 blocks and δ_3^*.

6.6 Comparison between 2D Methods

In order to come up with the best two-dimensional strategy for the alternative equation to be used with the masked Poisson equation, one must compare all the results together. In that regard, figure 6.9 presents the superimposition of the results obtained for all the best implementations of each strategy (DM-32, LSM-Y, LSM-O, 1DIM-4) as well as the original IL formulation represented by the version IL-FV.

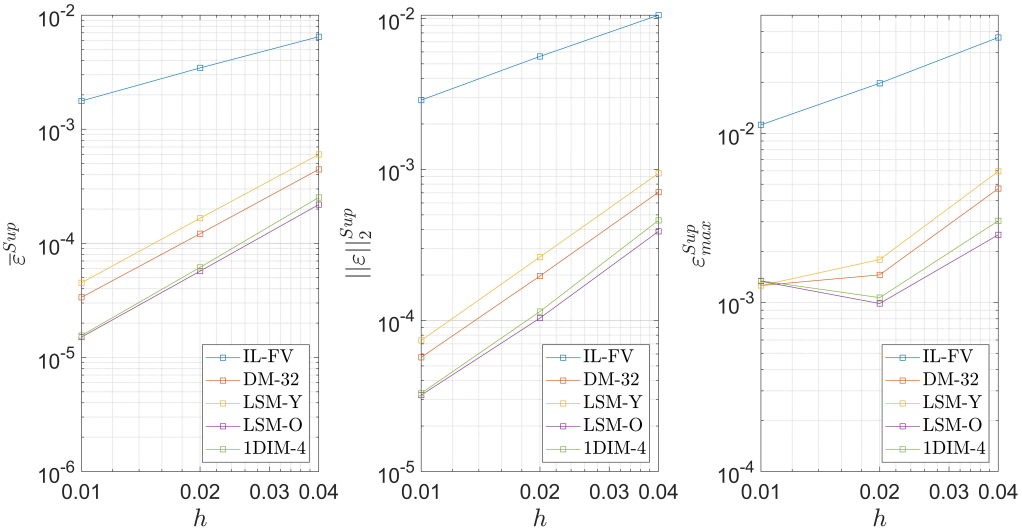


Figure 6.9: Comparison between the original IL formulation and the best methods, with δ_3^*.

As figure 6.9 shows, the strategy with lowest mean value and L^2 norm of the superfluous error is LSM-O, followed by 1DIM-4, DM-32, LSM-Y and finally the original formulation. The same order is true for the maximum superfluous error except for the most refined mesh, from which the order between the four alternative strategies inverts. This inversion is due to the fact that LSM-Y is the strategy with the highest decay of the maximum superfluous error, being expected to surpass the others. The superfluous

decays of all these strategies are gathered in table 6.9 for easier comparison. Although the graphs of versions IL-FV and IL-FD are practically superimposed, table 6.9 recovers the decays of IL-FD.

Table 6.9: Superfluous comparison between the original IL formulation and the best methods, with δ_3^*.

Strategy	$\bar{\varepsilon}^{Sup}$	$\|\varepsilon\|_2^{Sup}$	ε_{max}^{Sup}
IL-FD	0.9128	0.9257	0.9209
IL-FV	0.9345	0.9338	0.8581
DM-32	1.8625	1.8154	0.9521
LSM-Y	1.8680	1.8376	1.1217
LSM-O	1.9294	1.8006	0.4564
1DIM-4	2.0107	1.9010	0.5891

The highest superfluous decays (the closest ones to second order) appear to be achieved at the expense of letting the maximum superfluous error to decay in a poor way, as occurs for methods 1DIM-4 and LSM-O. On the other hand, strategy LSM-Y is able to keep a good decay of this error quantity, while performing close to second-order accuracy in the mean value and L^2 norm (with values higher than 1.8).

In table 6.9, the decays concerning the original formulation (IL-FD) used by Eldredge in article [1] are highlighted in blue. Strategies LSM-O and 1DIM-4 lead to the highest superfluous decays of the mean value and L^2 norm of the superfluous error, but present a decay of the maximum superfluous error that is lower than the original IL-FD method. So, focusing exclusively in the orders of decay computed for these meshes, strategies LSM-O and 1DIM-4 are disregarded. Strategies DM-32 and LSM-Y present first-order decay of the maximum superfluous error (similarly to IL-FD), but are able to achieve an order around 1.8 for the mean value and L^2 norm of the superfluous error, much higher than the original formulation. So, both DM-32 and LSM-Y reveal to be better than IL-FD, with strategy LSM-Y performing slightly better than DM-32. With this said, focusing exclusively in the orders of decay computed for these meshes, LSM-Y is elected as the strategy leading to the best superfluous decays.

Although the order of decay that the error quantities experience is an important parameter to track, the main goal in any simulation is to reach a numerical solution that presents the lowest error possible. Therefore, taking into account figure 6.9, if one intends to work exclusively in this range of refinements, LSM-O would result as a natural choice, despite its low order of decay in the maximum superfluous error.

6.7 Smearing Correction

In this subchapter, the two-dimensional smearing corrective procedures derived in subchapter 5.5 are implemented and compared. The main objective is to find out if they are able to propagate the achieved superfluous decays to the transition region and, consequently, to the entire domain. To test the corrective procedures, one considers, as an example, the least squares strategy LSM-Y, using δ_3^*, since it was the two-dimensional strategy presenting the best superfluous decays overall. The decays obtained for all the error quantities in all the domain regions are shown in table 6.10. For the sake of compactness, the polynomial strategy derived in subchapter 5.5.1 appears in the tables and legends coded as C-Poly, whereas the Cartesian strategy derived in subchapter 5.5.2 is coded as C-Cart.

Table 6.10: Comparison between the two-dimensional correction strategies for method LSM-Y with δ_3^*.

Correction	$\bar{\varepsilon}$	$\|\varepsilon\|_2$	ε_{max}	$\bar{\varepsilon}^{Sup}$	$\|\varepsilon\|_2^{Sup}$	ε_{max}^{Sup}	$\bar{\varepsilon}^{Trans}$	$\|\varepsilon\|_2^{Trans}$	$\varepsilon_{max}^{Trans}$
C-Poly	2.0787	2.0388	0.8219	1.8680	1.8376	1.1217	1.7809	1.6942	0.8219
C-Cart	2.1131	2.0986	1.3739	1.8680	1.8376	1.1217	1.7649	1.7248	1.3739

Notice that the superfluous decays in table 6.10 are identical to the ones in table 6.5 for the method LSM-Y, since the corrective procedures only act in the transition region. In terms of the orders of decay that were obtained by the correction in the transition region and in the entire domain, procedure C-Cart led to slightly better results. However, as already identified, the maximum error in the superfluous region presents some oscillations in the decay, contrary to what happens with the mean value and the L^2 norm. As such, the corrective procedures are acting over a superfluous field that is not strictly stabilised. Therefore, as it was already verified in the one-dimensional results (recall the last lines of tables 4.14 to 4.16), the decays obtained in the corrected transition region, and consequently in the entire domain, may occur to be higher than the decays verified in the superfluous region from which they were computed. Naturally this is just a numerical phenomenon due to the fact that the superfluous decays are not fully stabilised, with the maximum error oscillating around the straight line tendency.

Since the orders of decay achieved by both corrections are similar and both are affected by the fact that the maximum error in the superfluous region oscillates, one should compare the value of the global error quantities, in order to assess if any of the procedures presents advantage over the other. Figure 6.10 presents the global decay graphs for strategy LSM-Y, when corrected with the polynomial and the Cartesian procedures. The code C0 stands for the case when no correction is applied to the results.

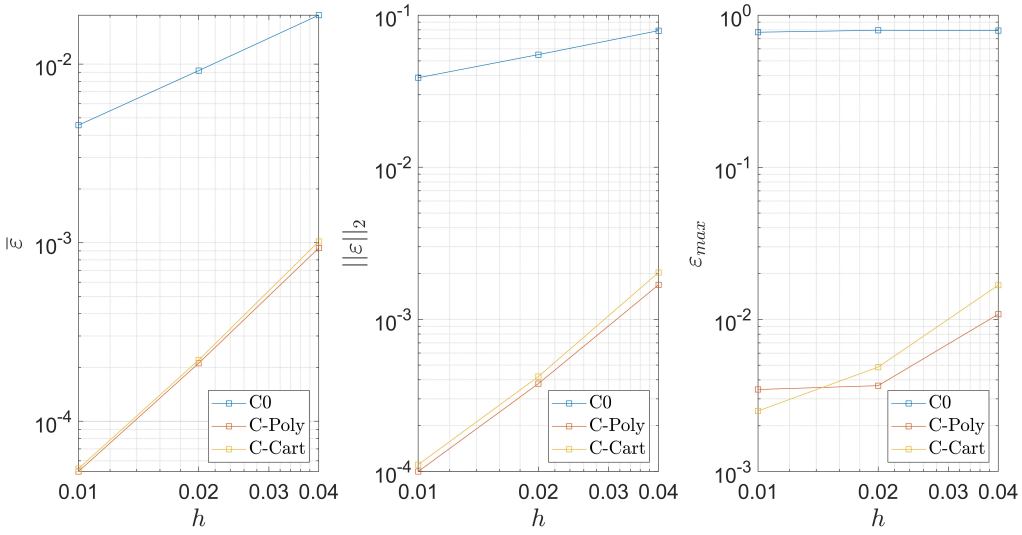
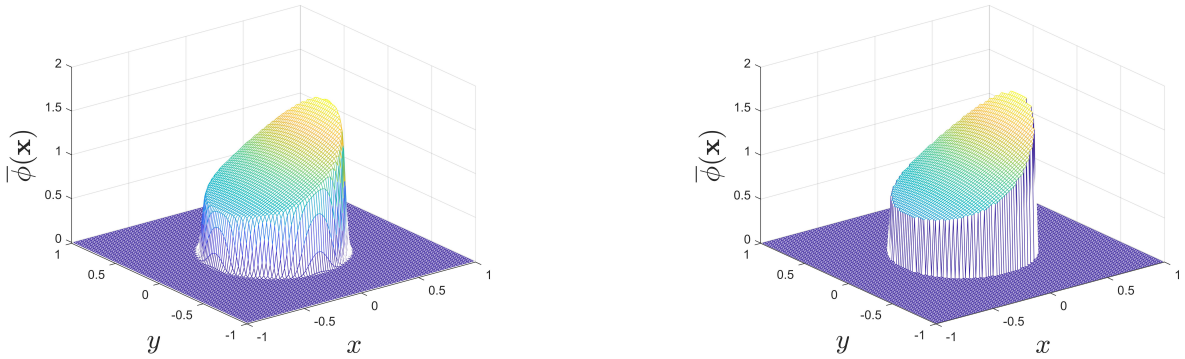


Figure 6.10: Comparison between corrective procedures C-Poly and C-Cart, on strategy LSM-Y with δ_3^*.

As previously seen, without correction, the global decay of the mean error is first-order accurate, the L^2 norm decays with half order and the maximum error does not decay (lines represented in blue in figure 6.10). On the other hand, as table 6.10 showed, both corrections lead to second-order global decay of the mean error and the L^2 norm and a decay around first-order for the maximum error in the

entire domain. When comparing the value of the error quantities, the lines representing both corrections are very close to each other, with correction C-Poly slightly better in the mean value and the L^2 norm. As an example, considering the central grid, the relative difference between both corrections is around 4% in the mean error and around 10% in the L^2 norm. However, notice that C-Cart displays slightly higher decays than C-Poly for the mean value and the L^2 norm. As such, it those decays continue to hold, the results for C-Cart are expected to surpass C-Poly as one continues to refine. Regarding the maximum error, finest grids would need to be analysed as well, in order to see whether C-Cart keeps the lowest value or if both correction procedures keep oscillating and changing their relative position.

In order to exemplify the two-dimensional sharpness recovery, the numerical solution for strategy LSM-Y is presented in figure 6.11, before and after corrective procedure C-Poly, using δ_3^* and $N = 100^2$.

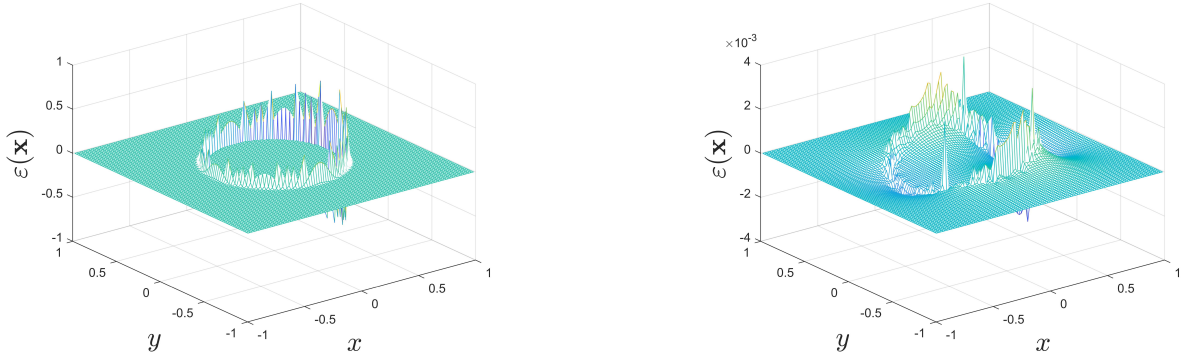


(a) Before correction.

(b) After corrective procedure C-Poly.

Figure 6.11: Numerical result obtained by strategy LSM-Y before and after correction C-Poly, using δ_3^*.

As expected, the numerical field in figure 6.11a presents a smeared interface due to the effect of the discrete delta functions in the vicinity of the interface, whereas its corrected version in figure 6.11b displays the desired sharp interface. Figure 6.12 shows the error field before and after the correction.



(a) Before correction.

(b) After corrective procedure C-Poly.

Figure 6.12: Error field obtained by strategy LSM-Y before and after correction C-Poly, using δ_3^*.

The results presented by correction C-Cart are identical, with no visual difference from the ones obtained with correction C-Poly. As such, one may conclude that both two-dimensional corrective procedures perform as expected, being able to recover interface sharpness and, as previously seen in table 6.10, to propagate superfluous decays to the transition region and, consequently, to the entire domain.

Chapter 7

Conclusions

7.1 Achievements

In this work, several improvement strategies were derived, to increase the accuracy of the Immersed Layers Method, when solving the Poisson equation. Given the treatment of the normal derivative jump as unknown, Eldredge [1] closed the system by setting an additional equation relying on a field interpolation performed using a discrete delta function. As such, during this dissertation, alternative equations to close the system were derived, implemented and compared. Moreover, the additional equations were combined with the masked Poisson equation under a finite volume framework, in alternative to what was done in article [1] that used finite differences.

In the one-dimensional case, the derived strategies led to second-order decays in the superfluous region. Depending on the number of points used in each strategy, some implementations revealed lower error than others, but all of them were second-order accurate outside of the transition region. At that point, having a superfluous field decaying with second-order, one derived several final corrective procedures to apply in the numerical solution coming from the system of equations. These procedures acted exclusively on transition values, that are inherently affected by the smearing when the field is not smooth across the interface. All the one-dimensional procedures were able to recover the interface sharpness and to propagate the superfluous decays to the transition region and, consequently, to the entire domain. As such, the objective of obtaining second-order results with Immersed Layers in one-dimensional problems was completely achieved for all the error quantities, in the entire domain.

In the two-dimensional case, several strategies were derived and tested, but only a few implementations led to more advantageous results when compared to the original Immersed Layers Method. When dealing with non-smooth fields, the two-dimensional reversion to first-order accurate results when using discrete delta functions is a problem “known in literature for decades”, as stated by Cola *et al.* in their recent article [18]. As such, obtaining second-order decays of all the error quantities in a two-dimensional problem is still an open question in the literature when using discrete delta functions. Nowadays, any accuracy improvement over first-order should be seen as a very positive result. Although the second equation using the discrete delta function to perform an interpolation operation was substituted, the regularisation operations appearing in the masked Poisson equation continue to be present. Therefore, even

when using second equations that are formally second-order accurate, the undesirable two-dimensional effect of the discrete delta functions that is present in the first equation will inevitably lead to accuracy penalties. Even so, in this work one was able to obtain second-order decays of the mean value and L^2 norm of the superfluous error, as in the 1D-Inspired Method with 4×4 blocks. However, at the same time, strong penalties affected the maximum superfluous error, that presented a decay of around half order (worse than the first order verified in the original formulation and thus unacceptable).

Overall, the best two-dimensional decays in which any error quantity presented penalties, when compared to the original method, was found to occur using method LSM-Y together with δ_3^*. In this method, the mean value and L^2 norm of the superfluous error decays with an order slightly higher than 1.8, whereas the maximum superfluous error decays with order 1.1. Regarding the 2D corrective procedures, two versions were successfully implemented, propagating the good superfluous results of strategy LSM-Y to the transition region and, consequently, to the entire domain. Concerning the discrete delta functions that were tested, all of them behaved well in the one-dimensional case, allowing global second-order results for all the derived strategies. However, in two dimensions, the results obtained using δ_{2002} and δ_{2103}, the discrete delta functions with lowest support width, presented strong penalties in the error decay values, even when used in the original formulation of the Immersed Layers Method. Discrete delta functions δ_{4206} and δ_3^* performed much better, with δ_3^* presenting the best results.

7.2 Future Work

Although the one-dimensional problem was completely solved with second-order accuracy for all error quantities in the entire domain, the two-dimensional results were not able to achieve that ideal level of accuracy. Even with the superfluous mean value and L^2 norm presenting second-order accuracy for some cases, the decay of the superfluous maximum error revealed to be very difficult to improve. As such, future work could still be developed, to improve the two-dimensional results even further.

As an example, a second equation relying on imposing the field value at the interface could be tried. Recall that this strategy was not generalised to the two-dimensional case, since the one-dimensional results showed that the equation expressing the normal derivative jump presented a lower error.

Moreover, several additional studies may be performed with the implemented methods. Giving some examples, one could test the use of a different number of solid points, choose the superfluous points to use in the Least Squares Method in a different way, increase the number of points used along the normal direction in the 1D-Inspired Method or set a different location of the first point along the normal direction. Besides this, new discrete delta functions may also be searched or even derived.

However, with the current methods, one could follow the line of thought constructed by Eldredge in article [1] and replicate the improvements strategies derived here for the masked Poisson equation to the masked convection-diffusion equation and, later on, to the masked Navier-Stokes equations.

In the long term, after the problem concerning second-order decay in two-dimensional problems is mastered, higher-order accuracy could be pursued. However, taking into account the current state of art for the accuracy of continuous forcing methods with non-smooth fields, this objective is highly ambitious.

Bibliography

[1] J. Eldredge. A method of immersed layers on Cartesian grids, with application to incompressible flows. *Journal of Computational Physics*, 448:110716, Jan. 2022. doi:10.1016/j.jcp.2021.110716.

[2] C. Peskin. Flow pattern around heart valves: A numerical method. *Journal of Computational Physics*, 10(2):252–271, Oct. 1972. doi:10.1016/0021-9991(72)90065-4.

[3] C. Peskin. Numerical analysis of blood flow in the heart. *Journal of Computational Physics*, 25(3): 220—-252, Nov. 1977. doi:10.1016/0021-9991(77)90100-0.

[4] C. Peskin and M. Lai. An Immersed Boundary Method with Formal Second-Order Accuracy and Reduced Numerical Viscosity. *Journal of Computational Physics*, 160(2):705–719, May 2000. doi:10.1006/jcph.2000.6483.

[5] C. Peskin. The Immersed Boundary Method. *Acta Numerica*, 11:479–517, Jan. 2002. doi:10.1017/S0962492902000077.

[6] R. Mittal and G. Iccarino. Immersed Boundary Methods. *Annual Review of Fluid Mechanics*, 37: 239–261, 2005. doi:10.1146/annurev.fluid.37.061903.175743.

[7] F. Sotiropoulos and X. Yang. Immersed boundary methods for simulating fluid-structure interaction. *Progress in Aerospace Sciences*, 65:1–21, Feb. 2014. doi:10.1016/j.paerosci.2013.09.003.

[8] C. Peskin and B. Griffith. On the order of accuracy of the immersed boundary method: Higher order convergence rates for sufficiently smooth problems. *Journal of Computational Physics*, 208 (1):75–105, Sept. 2005. doi:10.1016/j.jcp.2005.02.011.

[9] R. Leveque and Z. Li. The Immersed Interface Method for elliptic equations with discontinuous coefficients and singular sources. *SIAM Journal on Numerical Analysis*, 31(4):1019–1044, Aug. 1994. doi:10.1137/0731054.

[10] S. Xu and Z. Wang. An immersed interface method for simulating the interaction of a fluid with moving boundaries. *Journal of Computational Physics*, 216(2):454–493, Aug. 2006. doi:10.1016/j.jcp.2005.12.016.

[11] Z. Li and M. Lai. The Immersed Interface Method for the Navier–Stokes Equations with Singular Forces. *Journal of Computational Physics*, 171(2):822–842, Aug. 2001. doi:10.1006/jcph.2001.6813.

[12] R. Balam and M. Zapata. A fourth-order compact implicit immersed interface method for 2D Poisson interface problems. *Computers and Mathematics with Applications*, 119:257–277, Aug. 2022. doi:10.1016/j.camwa.2022.06.011.

[13] K. Taira and T. Colonius. The immersed boundary method: A projection approach. *Journal of Computational Physics*, 225(2):2118—-2137, Aug. 2007. doi:10.1016/j.jcp.2007.03.005.

[14] B. Leathers. *The Immersed Boundary Double Layer (IBDL) Method*. PhD thesis, University of California, 2022.

[15] Y. Mori. Convergence Proof of the Velocity Field for a Stokes Flow Immersed Boundary Method. *Communications on Pure and Applied Mathematics*, 61(9):1213–1263, Sept. 2008. doi:10.1002/cpa.20233.

[16] F. Diogo. A Very High-Order Finite Volume Technique for Convection-Diffusion Problems on Unstructured Grids. Master's thesis, Instituto Superior Técnico, 2019.

[17] A. Vasconcelos. A Very High-Order Finite Volume Method Based on Weighted Least Squares for the Solution of Poisson Equation on Unstructured Grids. Master's thesis, Instituto Superior Técnico, 2017.

[18] V. Cola, S. Cuomo, and G. Severino. Remarks on the numerical approximation of Dirac delta functions. *Results in Applied Mathematics*, 12:100200, Nov. 2021. doi:10.1016/j.rinam.2021.100200.

[19] Y. Liu and Y. Mori. Properties of discrete delta functions and local convergence of the Immersed Boundary Method. *SIAM Journal on Numerical Analysis*, 50(6):2986–3015, Aug. 2012. doi:10.1137/110836699.

[20] X. Yang, X. Zhang, Z. Li, and G. He. A smoothing technique for discrete delta functions with application to immersed boundary method in moving boundary simulations. *Journal of Computational Physics*, 228(20):7821–7836, Nov. 2009. doi:10.1016/j.jcp.2009.07.023.

[21] F. Moukalled, L. Mangani, and M. Darwish. *The Finite Volume Method in Computational Fluid Dynamics*. Springer, 1^{st} edition, 2016. ISBN:978-3-319-16873-9.

Appendix A

Generalised Functions

In this appendix, some properties of the generalised functions used in the Immersed Layers Method are presented, closely following the deductions performed in article [1] describing the method. Appendix A.1 concerns the Heaviside function, whereas appendix A.2 addresses the Dirac delta function. In appendix A.2.1 the unitary integral of the Dirac delta function is proved. In appendix A.2.2, equations 2.5 and 2.6 concerning respectively the immersion and restriction operations are derived.

A.1 Heaviside Function

Consider a given real function $f(x)$ with $x \in \mathbb{R}$. Given the definition of Heaviside function in equation 2.2, the integral of f over a semi-infinite interval may be converted into an integral of infinite range, according to equation A.1, since the Heaviside function will cancel out the integral in the interval that is complementary to the original one. Relation A.1 will allow the derivation of some important properties concerning the Dirac delta function, namely the proof of its unitary integral property.

$$\int_{-\infty}^{x} f(y)dy = \int_{-\infty}^{+\infty} H(x-y)f(y)dy \tag{A.1}$$

A.2 Dirac Delta Function

A.2.1 Unitary Integral

Differentiating equation A.1 with respect to x, one arrives to equation A.2, that is valid for any real function $f(x)$ with $x \in \mathbb{R}$. Notice that, by the chain rule, one may write $\frac{\partial}{\partial x} = \frac{\partial}{\partial(x-y)}\frac{\partial(x-y)}{\partial x} = \frac{\partial}{\partial(x-y)}$.

$$f(x) = \int_{-\infty}^{+\infty} \frac{\partial H(x-y)}{\partial x} f(y)dy = \int_{-\infty}^{+\infty} \delta(x-y)f(y)dy \tag{A.2}$$

Particularising function f to be equal to 1, one proves that the integral of the Dirac delta function over $\mathbb{R}$ is unitary, as presented in equation A.3. Since f is unitary (a constant) and the integration is being performed over the entire real line, the integral becomes translation invariant.

$$1 = \int_{-\infty}^{+\infty} \delta(x-y)\cdot 1\; dy \quad \Rightarrow \quad \int_{-\infty}^{+\infty} \delta(y)dy = 1 \tag{A.3}$$

Following the definition of the multidimensional Dirac delta function in equation 2.4, result A.3 implies the same property in the multidimensional space, $\mathbb{R}^\lambda$. Once more, the result is translation invariant, that is, the integral remains unitary when the integrand has the form $\delta(\mathbf{x}-\mathbf{x}_0)$, for any vector $\mathbf{x}_0 \in \mathbb{R}^\lambda$.

$$\int_{\mathbb{R}^\lambda} \delta(\mathbf{x})d\mathbf{x} = 1 \tag{A.4}$$

A.2.2 Immersion and Restriction Operations

Remembering that the interface Γ is considered to be parameterised by surface coordinate(s) $\boldsymbol{\zeta}$ with interface points being given by $\mathbf{x} = \mathbf{X}(\boldsymbol{\zeta})$, one may write equation A.5 for a general function $f(\mathbf{x})$ defined over the entire multidimensional space. This equation holds, since only values of f at the interface are actually invoked in the integral over $\mathbb{R}^\lambda$, due to the Dirac delta function.

$$\int_\Gamma f(\mathbf{X}(\boldsymbol{\zeta}))dS(\boldsymbol{\zeta}) = \int_{\mathbb{R}^\lambda} f(\mathbf{x})\delta(\chi(\mathbf{x}))d\mathbf{x} \tag{A.5}$$

As stated in article [1], equation A.6 may be written, being the validity of this relation proved by integrating it over $\mathbb{R}^\lambda$ and using equations A.4 and A.5 to manipulate it until an equality is reached.

$$f(\mathbf{x})\delta(\chi(\mathbf{x})) = \int_\Gamma f(\mathbf{X}(\boldsymbol{\zeta}))\delta(\mathbf{x}-\mathbf{X}(\boldsymbol{\zeta}))dS(\boldsymbol{\zeta}) \tag{A.6}$$

The demonstration procedure suggested in article [1] is presented in expressions A.7 to A.10. From step A.7 to step A.8, one used equation A.5 on the left-hand side and changed the order of integration in the right-hand side. From step A.8 to step A.9, the only term depending on $\mathbf{x}$ was isolated inside the integral with respect to that variable. From step A.9 to step A.10, property A.4 was employed.

$$\int_{\mathbb{R}^\lambda} f(\mathbf{x})\delta(\chi(\mathbf{x}))d\mathbf{x} = \int_{\mathbb{R}^\lambda}\int_\Gamma f(\mathbf{X}(\boldsymbol{\zeta}))\delta(\mathbf{x}-\mathbf{X}(\boldsymbol{\zeta}))\; dS(\boldsymbol{\zeta})\; d\mathbf{x} \tag{A.7}$$

$$\Leftrightarrow \int_\Gamma f(\mathbf{X}(\boldsymbol{\zeta}))\; dS(\boldsymbol{\zeta}) = \int_\Gamma\int_{\mathbb{R}^\lambda} f(\mathbf{X}(\boldsymbol{\zeta}))\delta(\mathbf{x}-\mathbf{X}(\boldsymbol{\zeta}))\; d\mathbf{x}\; dS(\boldsymbol{\zeta}) \tag{A.8}$$

$$\Leftrightarrow \int_\Gamma f(\mathbf{X}(\boldsymbol{\zeta}))\; dS(\boldsymbol{\zeta}) = \int_\Gamma f(\mathbf{X}(\boldsymbol{\zeta}))\left(\int_{\mathbb{R}^\lambda} \delta(\mathbf{x}-\mathbf{X}(\boldsymbol{\zeta}))\; d\mathbf{x}\right) dS(\boldsymbol{\zeta}) \tag{A.9}$$

$$\Leftrightarrow \int_\Gamma f(\mathbf{X}(\boldsymbol{\zeta}))\; dS(\boldsymbol{\zeta}) = \int_\Gamma f(\mathbf{X}(\boldsymbol{\zeta}))\; dS(\boldsymbol{\zeta}) \tag{A.10}$$

The right-hand side of equation A.6 only uses function values at the interface $f(\mathbf{X}(\boldsymbol{\zeta}))$. Moreover, the presence of the Dirac delta function in the left side of the equation implies that only interface values of f will actually be invoked too. Thus, relation A.6 holds, as well, for any function exclusively defined at the interface, $F(\boldsymbol{\zeta})$. Replacing $f(\mathbf{x})$ by $F(\boldsymbol{\zeta})$, one arrives to equation 2.5, that one was willing to derive.

On the other hand, the restriction operation of given function $g(\mathbf{x})$ with values in the multidimensional space $\mathbb{R}^\lambda$ simply arises from expressing property A.2 for the multidimensional case, as in equation A.11.

$$\int_{\mathbb{R}^\lambda} g(\mathbf{x})\delta(\mathbf{X}(\boldsymbol{\zeta}) - \mathbf{x})d\mathbf{x} = g(\mathbf{X}(\boldsymbol{\zeta})) \tag{A.11}$$

Using the notation of article [1] to represent the restriction operation of the given function $g(\mathbf{x})$ to the interface, $g(\mathbf{x})\delta^T\big(\chi(\mathbf{x})\big)$, one arrives to equation 2.6, as one was willing to derive.

A.3 Discrete Delta Function

Since it is not possible to numerically deal with singularities, the Dirac delta function is converted into a so-called discrete delta function. In a multidimensional space $\mathbb{R}^\lambda$, a discrete delta function is constructed multiplying real functions $\varphi(r)$ with $r \in \mathbb{R}$, as presented in equation 2.18. This function $\varphi(r)$ must verify several properties, so that the discrete delta function properly mimics the Dirac delta function. According to article [19], usual functions $\varphi(r)$ verify the following conditions:

- $\varphi(r)$ is continuous $\forall r \in \mathbb{R}$;
- φ has a compact support: $\varphi(r) = 0\ , |r| \geq \frac{w}{2}$, with w being the support width;
- φ satisfies the sum of squares condition: $\sum_{l\in\mathbb{Z}} \big[\varphi(l-r)\big]^2 = K\ ,\ \forall r \in \mathbb{R}$, for some $K \in \mathbb{R}$;
- φ has a given moment order m, for some $m \in \mathbb{N}_0$;
  Function φ is said to satisfy a given moment condition of order j if it verifies relation A.12.

$$\begin{cases} \sum_{k\in\mathbb{Z}} \varphi(k-r) = 1 & ,\ \text{if } j = 0 \\ \sum_{k\in\mathbb{Z}} (k-r)^j \varphi(k-r) = 0 & ,\ \text{if } j > 0 \end{cases} \tag{A.12}$$

  Function φ is said to have moment order m if it satisfies moment conditions up to order $m-1$.

- φ has a given smoothing order s, for some $s \in \mathbb{N}_0$;
  Function φ is said to have a given smoothing order $s \geq 1$, if there is a function $\psi(r)$ such that relation A.13 holds. Otherwise, φ is said to have $s = 0$, if it has a compact support.

$$\varphi(r) = \frac{1}{2^s} \sum_{l=0}^{s} \left(\frac{s!}{l!(s-l)!} \right) \psi(r-l) \tag{A.13}$$

As suggested, several parameters should be set when deriving a discrete delta function, namely the support width w, the moment order m and the smoothing order s. By imposing all these parameters, constant K will be a result from imposing the sum of squares condition. An example of the derivation of a discrete delta function may be found in article [5], wherein w, m and s are imposed, leading to the arising of constant K from the imposition of the sum of squares condition. As such, the discretisation of the Dirac delta function is not unique and multiple combinations of parameters may be set. However, as mentioned in that same article, not all the combinations of parameters are possible.

Notice that, in several works, the last property concerning the smoothing order appears replaced by a simpler formulation, namely in the article of Peskin, [5], setting a so-called even-odd condition, in

which $\sum_{i\ even}\varphi(r-i)=\sum_{i\ odd}\varphi(r-i)=\frac{1}{2}$. Article [19] proves that imposing the even-odd condition is equivalent to set a unitary smoothing order, $s=1$. Thus, the formulation with a smoothing order s is more general. Using the even-odd condition corresponds to be already setting this parameter as $s=1$.

As demonstrated in article [19], the moment order is related with the accuracy of interpolation operations performed with discrete delta functions, whereas the smoothing order is responsible for the suppression of high frequency errors. Since, in article [19], the sum of squares condition was not observed to have a relevant impact on the results, some discrete delta functions derived there do not verify it. More information about the importance of the sum of squares condition may be found in [5].

Similarly to what is defined in article [19], a discrete delta function is said to be of class (m,s,τ,w) if it is characterised by a moment order m, a smoothing order s and a support width w. The parameter τ assumes 1 if the sum of squares condition is verified and assumes 0 otherwise. Under this classification, that will be represented in subscript of each function, article [19] presents φ_{2002}, φ_{2103}, φ_{4206}, among many others. The expressions of these functions are presented in equations A.14, A.15 and A.16.

$$\varphi_{2002}(r)=\begin{cases}1-|r| & ,\ |r|\leq 1\\ 0 & ,\ |r|>1\end{cases} \tag{A.14}$$

$$\varphi_{2103}(r)=\begin{cases}0.5 & ,\ |r|\leq 0.5\\ \frac{3}{4}-\frac{1}{2}|r| & ,\ 0.5<|r|\leq 1.5\\ 0 & ,\ |r|>1.5\end{cases} \tag{A.15}$$

$$\varphi_{4206}(r)=\begin{cases}\frac{5}{8}-\frac{5}{24}|r|-\frac{1}{4}|r|^2+\frac{1}{12}|r|^3 & ,\ |r|\leq 1\\ \frac{9}{16}-\frac{11}{48}|r|-\frac{1}{8}|r|^2+\frac{1}{24}|r|^3 & ,\ 1<|r|\leq 2\\ \frac{13}{16}-\frac{49}{48}|r|+\frac{3}{8}|r|^2-\frac{1}{24}|r|^3 & ,\ 2<|r|\leq 3\\ 0 & ,\ |r|>3\end{cases} \tag{A.16}$$

The discrete delta function used by Eldredge in article [1] is constructed by means of a given φ_3^* function, that is derived in article [20]. This function results from applying a smoothing procedure (represented by the star symbol) to an initial function φ_3. This function φ_3 was obtained applying a different procedure from the ones in article [19] and, as such, one should not describe this function under that classification. Instead of setting m, s, w and imposing (or not) the sum of squares condition, from which a constant K results (or not); in this case, a constant K was imposed together with values for m and for w, setting no restriction for the smoothing order. The values considered to derive φ_3 were $K=\frac{1}{2}$, $m=2$ and $w=3$. The expression of function φ_3 is presented in equation A.17.

$$\varphi_3(r)=\begin{cases}\frac{1}{3}\left(1+\sqrt{-3r^2+1}\right) & ,\ |r|\leq 0.5\\ \frac{1}{6}\left(5-3|r|-\sqrt{-3(1-|r|)^2+1}\right) & ,\ 0.5<|r|\leq 1.5\\ 0 & ,\ |r|>1.5\end{cases} \tag{A.17}$$

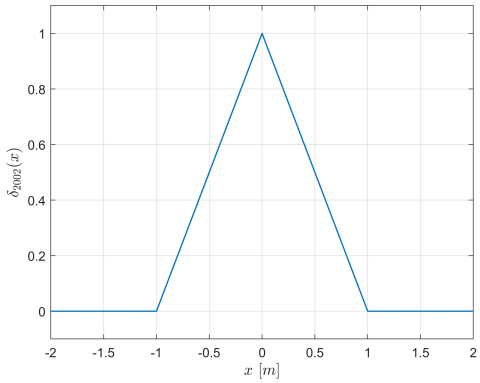
Finally, the authors of article [20] subjected function φ_3 to the smoothing procedure of equation A.18, from which its smoothed version φ_3^* arose. The objective of this smoothing procedure is to obtain discrete delta functions that lead to reduced non-physical oscillations when dealing with moving immersed boundary problems. Notice that, since smoothed versions φ^* are constructed by integrating the original functions φ, they present one higher derivative being continuous through each two neighbouring segments of the function. This property is what gives rise to the expression *smoothed*.

$$\varphi^*(r) = \int_{r-0.5}^{r+0.5} \varphi(r')dr' \tag{A.18}$$

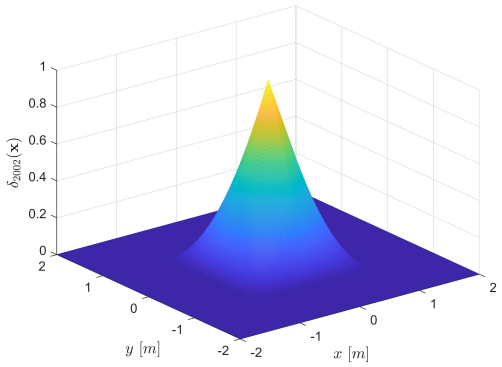
The final expression of the smoothed function φ_3^* is presented in equation A.19, from article [20].

$$\varphi_3^*(r) = \begin{cases} \frac{17}{8} + \frac{\sqrt{3}\pi}{108} + \frac{|r|}{4} - \frac{r^2}{4} + \frac{(1-2|r|)\sqrt{-12r^2+12|r|+1}}{16} - \frac{\sqrt{3}}{12}\arcsin\left(\frac{\sqrt{3}(2|r|-1)}{2}\right) & ,|r| \leq 1 \\ \frac{55}{48} - \frac{\sqrt{3}\pi}{108} - \frac{13|r|}{12} + \frac{r^2}{4} + \frac{(2|r|-3)\sqrt{-12r^2+36|r|-23}}{48} + \frac{\sqrt{3}}{36}\arcsin\left(\frac{\sqrt{3}(2|r|-3)}{2}\right) & ,1 < |r| \leq 2 \\ 0 & ,|r| > 2 \end{cases} \tag{A.19}$$

The graphics of all the discrete delta functions considered in this work (δ_{2002}, δ_{2103}, δ_{4206}, δ_3^*) are provided in figures A.1, A.2, A.3 and A.4, respectively, implemented in one and two dimensions.

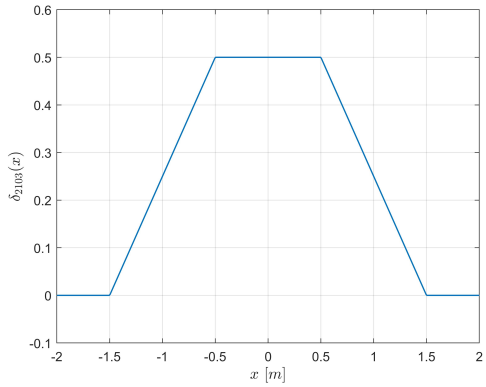


(a) One-dimensional version of δ_{2002}.

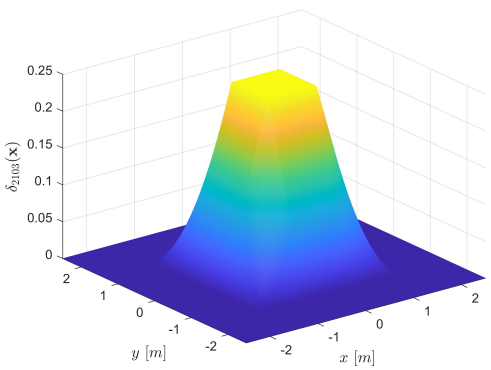


(b) Two-dimensional version of δ_{2002}.

Figure A.1: Representation of discrete delta function δ_{2002}, in one and two dimensions, taking $h = 1$.

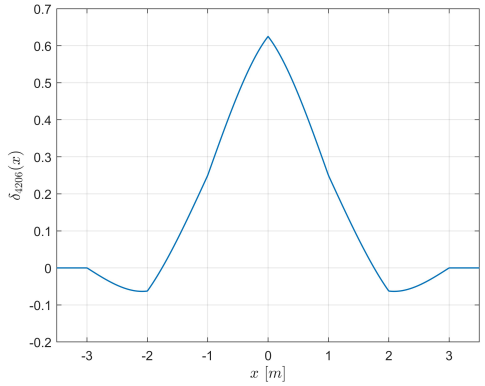


(a) One-dimensional version of δ_{2103}.

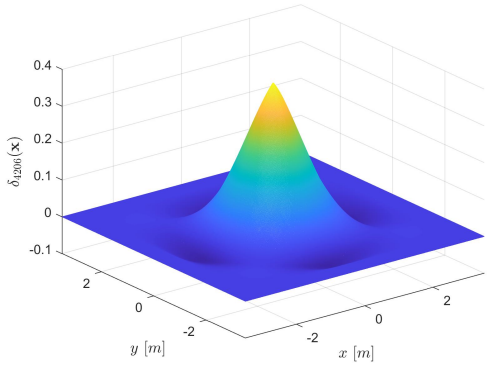


(b) Two-dimensional version of δ_{2103}.

Figure A.2: Representation of discrete delta function δ_{2103}, in one and two dimensions, taking $h = 1$.

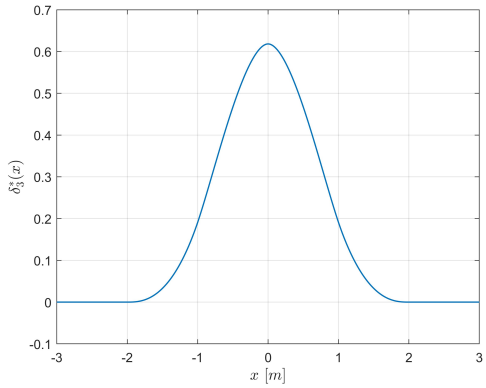


(a) One-dimensional version of δ_{4206}.

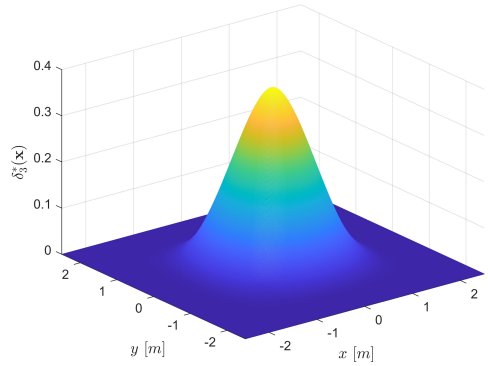


(b) Two-dimensional version of δ_{4206}.

Figure A.3: Representation of discrete delta function δ_{4206}, in one and two dimensions, taking $h = 1$.



(a) One-dimensional version of δ_3^*.



(b) Two-dimensional version of δ_3^*.

Figure A.4: Representation of discrete delta function δ_3^*, in one and two dimensions, taking $h = 1$.

Appendix B

Calculus

In this appendix, the Gauss divergence theorem is reminded in B.1. In B.2 the second-order schemes used in the directional derivative expressions for non-interior cells (equations 2.28 to 2.31) are derived.

B.1 Gauss Divergence Theorem

Consider a compact subset $V \subset \mathbb{R}^3$, with boundary $S = \partial V$, that is characterised by an outward pointing unit normal $\mathbf{n}$. For a continuously differentiable vector field $\mathbf{u} : V \to \mathbb{R}^3$, equation B.1 holds, being known as the Gauss divergence theorem. This result may be found, for example, in book [21].

$$\iiint_V \nabla \cdot \mathbf{u} \, dV = \oiint_S \mathbf{u} \cdot \mathbf{n} \, dS \tag{B.1}$$

B.2 Face Directional Derivative in Non-Interior Cells

In this appendix, one derives the unusual formula presented in equation 2.28, concerning the computation of the directional derivative for the east face of the easternmost cell of the computational domain. The problem layout along x is presented in figure B.1. Note that, even though only an x axis is represented, everything remains valid for the computation of $\frac{\partial \bar{\phi}}{\partial x}\big|_e$ in two (and even three) dimensions.

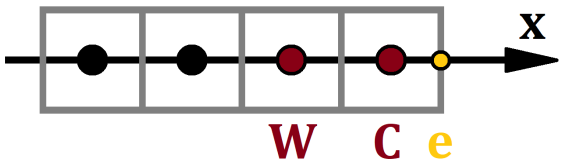


Figure B.1: Layout, along x, of a in the easternmost boundary of the computational domain.

Performing a Taylor series expansion for values $\bar{\phi}_C$ and $\bar{\phi}_W$ around x_e, one arrives to equations B.2.

$$\begin{cases} \bar{\phi}_C = \bar{\phi}_e + (x_C - x_e)\frac{\partial \bar{\phi}}{\partial x}\big|_e + \frac{(x_C - x_e)^2}{2!}\frac{\partial^2 \bar{\phi}}{\partial x^2}\big|_e + \frac{(x_C - x_e)^3}{3!}\frac{\partial^3 \bar{\phi}}{\partial x^3}\big|_e + \ldots \\ \bar{\phi}_W = \bar{\phi}_e + (x_W - x_e)\frac{\partial \bar{\phi}}{\partial x}\big|_e + \frac{(x_W - x_e)^2}{2!}\frac{\partial^2 \bar{\phi}}{\partial x^2}\big|_e + \frac{(x_W - x_e)^3}{3!}\frac{\partial^3 \bar{\phi}}{\partial x^3}\big|_e + \ldots \end{cases} \tag{B.2}$$

One is looking for coefficients $\alpha, \beta, \gamma \in \mathbb{R}$ such that the derivative in position x_e is approximated with second-order accuracy by as expression invoking $\bar{\phi}_e$, $\bar{\phi}_C$ and $\bar{\phi}_W$, as generally written in equation B.3.

$$\left.\frac{\partial \bar{\phi}}{\partial x}\right|_e \quad \underset{\sim}{O2} \quad \frac{\alpha\bar{\phi}_W + \beta\bar{\phi}_C + \gamma\bar{\phi}_e}{h} \tag{B.3}$$

By expanding the right-hand side of equation B.3 with the results from the Taylor series expansions of B.2, expression B.4 results. The term $\mathcal{O}(h)^2$ is used to represent an infinite sum of terms, in which the one with lower power of the grid size is h^2. Notice that $x_C - x_e = -\frac{h}{2}$ and $x_W - x_e = -\frac{3h}{2}$.

$$\frac{\alpha\bar{\phi}_W + \beta\bar{\phi}_C + \gamma\bar{\phi}_e}{h} = \frac{\bar{\phi}_e}{h}\left(\alpha+\beta+\gamma\right) - \left(\frac{3}{2}\alpha + \frac{1}{2}\beta\right)\left.\frac{\partial\bar{\phi}}{\partial x}\right|_e + \frac{h}{2!}\left(\frac{9}{4}\alpha + \frac{1}{4}\beta\right)\left.\frac{\partial^2\bar{\phi}}{\partial x^2}\right|_e + \mathcal{O}(h)^2 \tag{B.4}$$

Thus, for equation B.3 to hold, coefficients α, β, γ should verify the three conditions presented in equation B.5. The term multiplying the derivative should be unitary and the others should cancel out.

$$\begin{cases} \alpha+\beta+\gamma = 0 \\ \frac{3}{2}\alpha + \frac{1}{2}\beta = -1 \\ \frac{9}{4}\alpha + \frac{1}{4}\beta = 0 \end{cases} \rightarrow \begin{cases} \alpha = 1/3 \\ \beta = -3 \\ \gamma = 8/3 \end{cases} \tag{B.5}$$

Having determined coefficients α, β and γ, one is able to substitute them in expression B.3. With this, one arrives to equation 2.28, expressing the directional derivative in the east face of a cell located at the easternmost boundary of the domain. Its second-order nature is proved by re-writing equation B.4 with the values of coefficients α, β and γ being substituted. This step is performed in equation B.6.

$$\frac{(1/3)\bar{\phi}_W + (-3)\bar{\phi}_C + (8/3)\bar{\phi}_e}{h} = \left.\frac{\partial\bar{\phi}}{\partial x}\right|_e + \mathcal{O}(h)^2 \tag{B.6}$$

Similar procedures allow the deduction of equations 2.29, 2.30 and 2.31, for the directional derivatives in identical problems at the westernmost, northernmost and southernmost domain boundaries.

Appendix C

1D Results - Additional Data

In this appendix, some additional data concerning the one-dimensional results of chapter 4 is presented. The graphs that compare the σ-equation strategy and the $\bar{\phi}$-equation strategy are here displayed for discrete delta functions δ_{2103}, δ_3^* and δ_{4206}, in appendix C.1. Moreover, the graphs that compare the several corrective procedures, when applied to the σ-equation strategy using two superfluous points, are here presented for discrete delta functions δ_{2002}, δ_{2103} and δ_3^*, in appendix C.2.

C.1 Comparison between Strategies

In addition to figure 4.11 comparing the σ-equation strategies and $\bar{\phi}$-equation strategies for δ_{2002}, the equivalent graphs for δ_{2103}, δ_3^* and δ_{4206} are presented in figures C.1, C.2 and C.3,respectively.

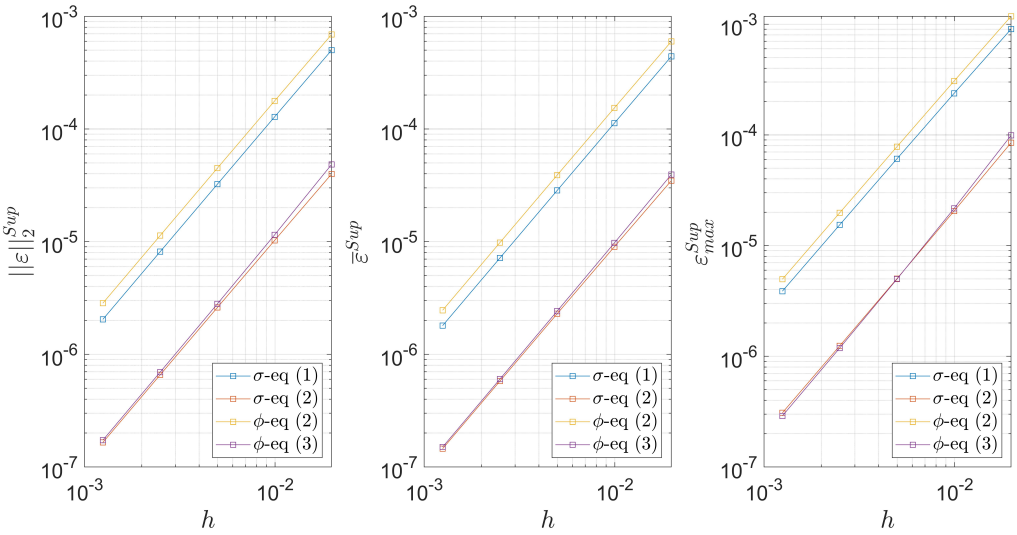


Figure C.1: Gathering of all the superfluous decays with σ-equation and $\bar{\phi}$-equation strategies for δ_{2103}.

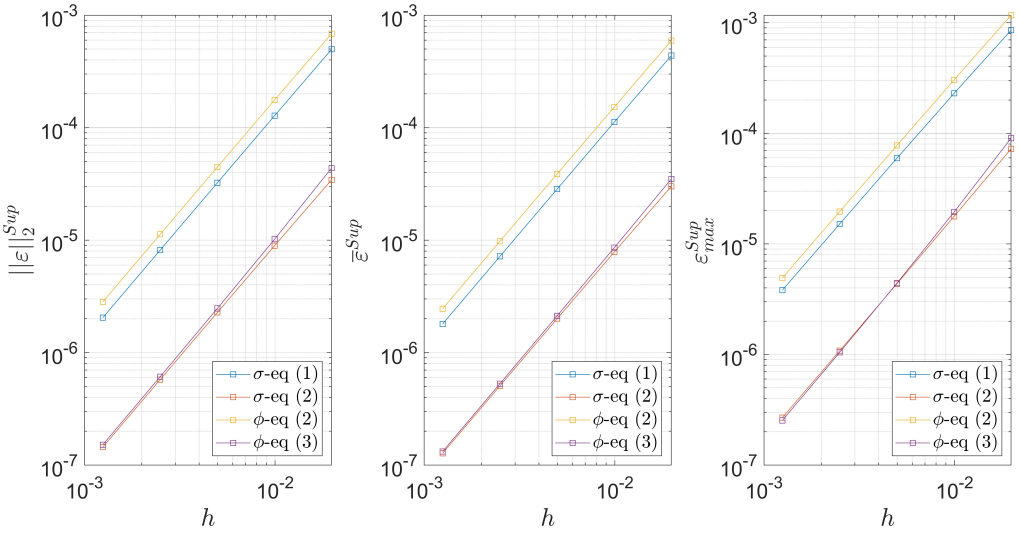


Figure C.2: Gathering of all the superfluous decays with σ-equation and $\bar{\phi}$-equation strategies for δ_3^*.

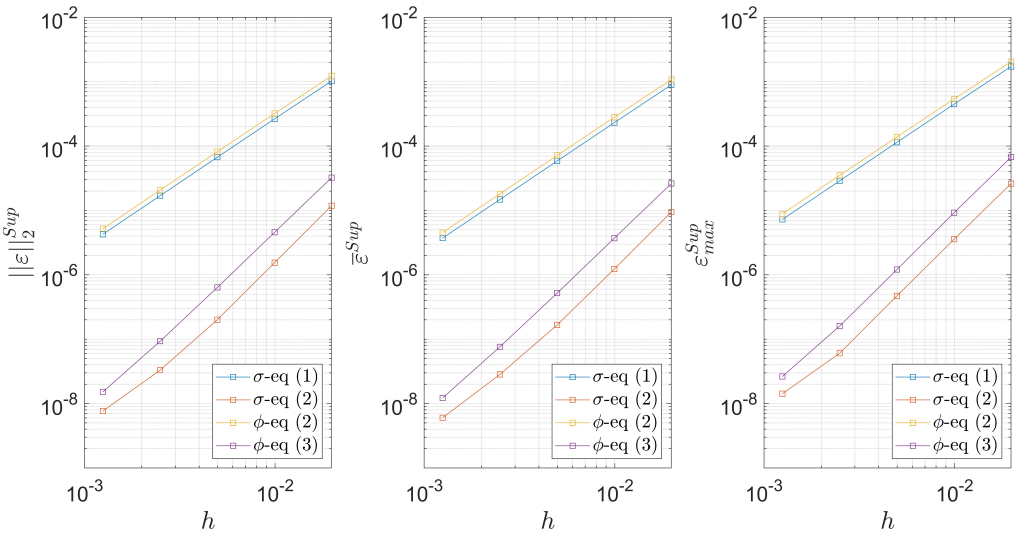


Figure C.3: Gathering of all the superfluous decays with σ-equation and $\bar{\phi}$-equation strategies for δ_{4206}.

C.2 Comparison between Corrective Procedures

In addition to figure 4.12 comparing the several correction procedures for δ_{4206}, the equivalent graphs for δ_{2002}, δ_{2103} and δ_3^* are presented in figures C.4, C.5 and C.6,respectively.

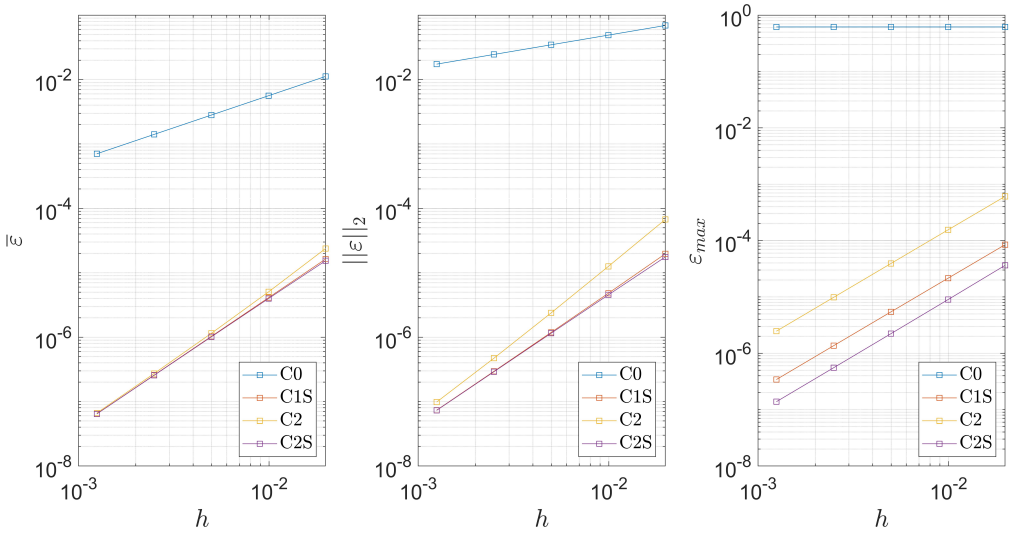


Figure C.4: Gathering of all the global decays using different corrective procedures, for δ_{2002}.

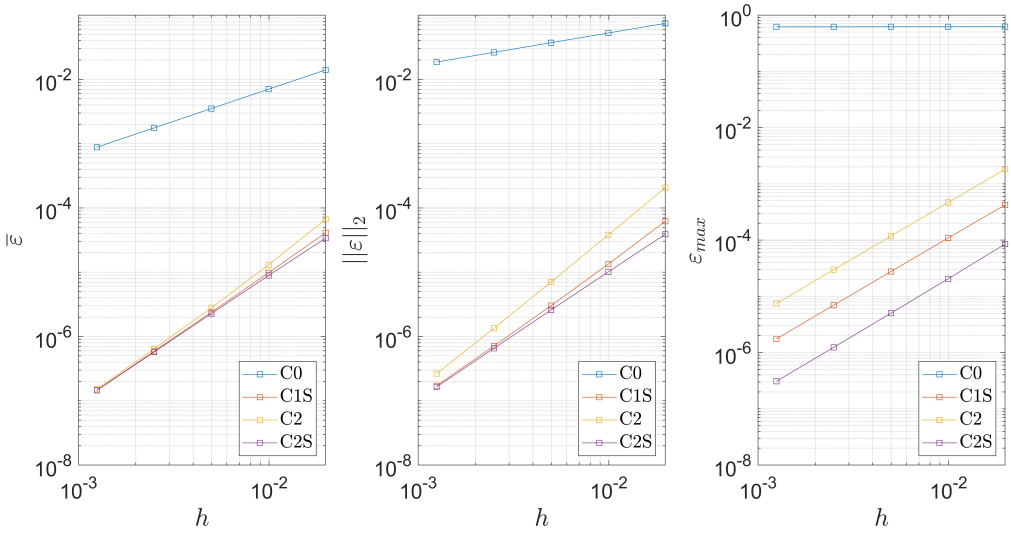


Figure C.5: Gathering of all the global decays using different corrective procedures, for δ_{2103}.

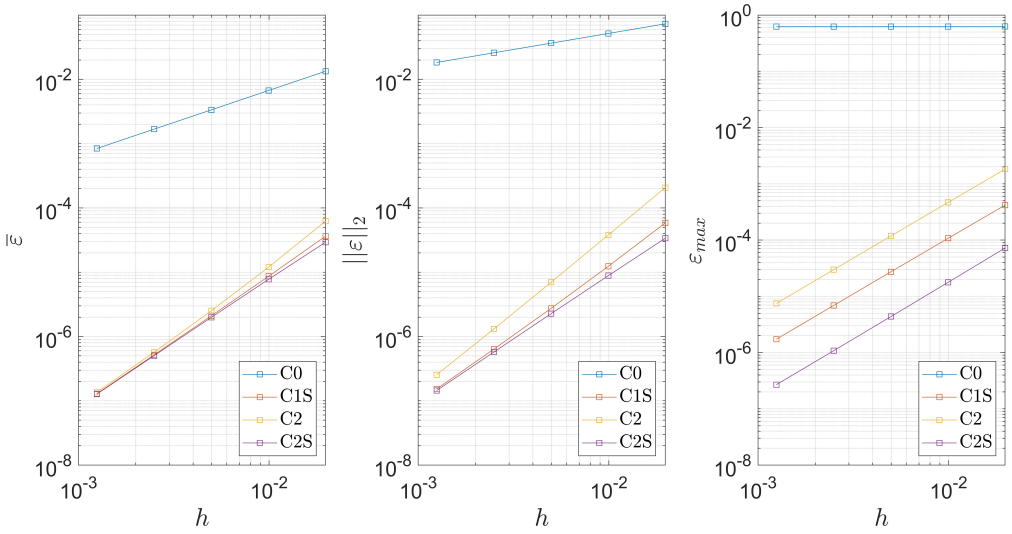


Figure C.6: Gathering of all the global decays using different corrective procedures, for δ_3^*.