UNIVERSIDADE D
COIMBRA

Milene Francisco dos Santos

ORDER OPTIMISATION IN CURVED BOUNDARY DOMAINS

DISCONTINUOUS GALERKIN METHOD

Dissertação no âmbito do Mestrado em Matemática, Ramo de Estatística, Otimização e Matemática Financeira orientada pelo Professor Doutor Adérito Luís Martins Araújo e pelo Professor Doutor José Luis Esteves dos Santos e apresentada ao Departamento de Matemática da Faculdade de Ciências e Tecnologia.

Junho de 2022

Order Optimisation in Curved Boundary Domains Discontinuous Galerkin Method

Milene Francisco Santos

UNIVERSIDADE D
COIMBRA

Master in Mathematics

Mestrado em Matemática

MSc Dissertation | Dissertação de Mestrado

Junho 2022

Acknowledgements

I would like to express my deepest gratitude to my supervisors, Professor Adérito Araújo and Professor José Luis dos Santos, for all the guidance and constant support during this year, for everything they taught me and for always being available for work discussions.

I would also like to acknowledge Professor Sílvia Barbeiro for her help with some issues during this thesis and for her guidance and encouragement.

I would like to acknowledge the financial support by FEDER – Fundo Europeu de Desenvolvimento Regional, through COMPETE 2020 – Programa Operacional Fatores de Competitividade, and the National Funds through FCT – Fundação para a Ciência e a Tecnologia, project no. POCI-01-0145-FEDER-028118. PTDC/MAT-APL/28118/2017.

I also want to thank my friends for all the support in my personal and academic life, for being there when I needed them and for all the good moments and laughs.

Last but not least, I would like to thank my family for their love, encouragement, assistance, and support.

Abstract

The problem that motivated the choice of the subject of this thesis is related to the description of the behaviour of electromagnetic waves in the human cornea, in order to understand the reasons that lead to the opacity of the cornea. Maxwell's equations, which describe the propagation of electromagnetic fields, are the natural mathematical model for our study. In this work, we have chosen to consider a simplified model given by the Helmholtz equation, assuming the harmonic variation in time of the electromagnetic fields.

To solve the Helmholtz equation we use the discontinuous finite element method Galerkin (DG). Since we are interested in solving the equation in a domain with a curved boundary, which intends to mimic the human cornea, polygonal meshes do not polygonal meshes do not fit exactly into the physical domain, which leads to a reduction in the accuracy of the numerical method.

In this thesis we propose two approaches to deal with the reduction in the accuracy of the DG method in a domain with a curved boundary, in solving the Helmholtz problem in two dimensions with homogeneous Dirichlet boundary conditions. The first method is called DG-ROD (Reconstruction for Off-site Data), is based on a polynomial reconstruction of the boundary condition imposed on the computational domain that takes into account the boundary condition imposed on the physical domain. The numerical tests show a reduction of the error and an increase in the order of convergence of the method, in relation to the classical DG method. The second method proposed, called DG-NM (Nelder-Mead) with step size control, is based on changing the boundary condition imposed on the computational domain by solving an unconstrained minimisation problem. This minimisation problem is solved with a variant of the Nelder-Mead method. The numerical tests evidence a decrease in the error in relation to the classical DG method and a decrease in the number of iterations in relation to the classical NM method. Both methods suggested in this thesis have the advantage of not requiring the generation of curved meshes to adjust the boundary nor complex nonlinear transformations to map the curved elements to the reference one, in relation to other alternatives to deal with curved boundary domains.

Keywords: Helmholtz's equation, discontinuous Galerkin method, curved boundary domains, polynomial reconstruction, optimisation, Nelder-Mead method.

Resumo

O problema que motivou a escolha do tema desta tese prende-se com descrição do comportamento das ondas electromagnéticas na córnea humana, a fim de compreender as razões que conduzem à opacidade da córnea. As equações de Maxwell, que descrevem a propagação de campos electromagnéticos, constituem o modelo matemático natural para o nosso estudo. Neste trabalho, optámos por considerar um modelo simplificado dado pela equação de Helmholtz, assumindo a variação harmónica no tempo dos campos electromagnéticos.

Para resolver a equação de Helmholtz recorremos ao método dos elementos finitos descontínuos Galerkin (DG). Uma vez que estamos interessados em resolver a equação num domínio com fronteira curva, que pretende mimetizar a córnea humana, as malhas poligonais não malhas poligonais não se ajustam exactamente ao domínio físico, o que conduz a uma redução da precisão do método numérico.

Nesta tese propomos duas abordagens para lidar com a redução da precisão do método da DG num domínio com fronteira curva, na resolução do problema de Helmholtz em duas dimensões com condições de fronteira de Dirichlet homogéneas. O primeiro método designa-se por DG-ROD (Reconstruction for Off-site Data) e baseia-se na reconstrução polinomial da condição de fronteira imposta no domínio computacional, tendo em conta a condição de fronteira imposta no domínio físico. Os testes numéricos realizados evidenciam a diminuição do erro e o aumento da ordem de convergência do método, em relação ao método DG clássico. O segundo método proposto designa-se DG-NM (Nelder-Mead) com controlo do passo e baseia-se na alteração da condição de fronteira imposta no domínio computacional através da resolução de um problema de minimização sem restrições. Esse problema de minimização é resolvido com uma variante do método de Nelder-Mead. Os testes numéricos realizados evidenciam a diminuição do erro em relação ao método DG clássico e uma diminuição do número de iterações em relação ao método de NM clássico. Ambos os métodos sugeridos nesta tese têm a vantagem de não utilizar malhas com elementos curvos nem transformações não lineares do elemento da malha para o elemento de referência, comparativamente a outras alternativas para lidar com domínios com fronteira curva.

Palavras-Chave: Equação de Helmholtz, método de Galerkin descontínuo, domínios com fronteira curva, rescontrução polinomial, otimização, método Nelder-Mead.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

The Helmholtz equation has an important role in physics and has several applications, particularly in the fields of optics, acoustics, electrostatics and quantum mechanics. This equation is usually associated with vibrating membranes (such as drums), lasers and the propagation of sound and electromagnetic waves. The Helmholtz equation, $-\nabla^2 u - \nu^2 u = 0$, with wave number ν, is the simplest model of wave propagation. For instance, if $U(x,t) = u(x)e^{i\omega t}$ is solution of the wave equation $\partial^2 U/\partial t^2 - c^2\nabla^2 U = 0$, then the function $u(x)$ satisfies the Helmholtz's equation with $\nu = \omega/c$. Assuming an analogous relationship in time, Maxwell's equations reduce to the time-harmonic Maxwell equations and, under certain conditions, can be further reduced to the Helmholtz equation. Similarly, the time-harmonic elastic wave equation (often called the Navier equation) also reduces to the Helmholtz equation under certain conditions. Thus, the main reason for interest of the Helmholtz equation in the area of physics is the fact that this equation describes the solution of the wave equation, when we consider a harmonic variation in time. This allows us to state that the Helmholtz equation is at the heart of the description of linear wave propagation and so efficient methods for solving the equation and studying the properties of its solutions have been discussed in literature.

This work is part of a more generic project that consists in analysing the incidence and reflection of light in the cornea [1]. The cornea (see Figure 1.1) corresponds to the transparent part of the outer layer of the eye and its curved interface provides three-quarters of the eye's focusing power (the rest being provided by the lens). Thus, maintaining the curvature and transparency of the cornea is essential for good vision, which is translated by less light reflection and therefore more information is captured. The reasons that lead to corneal opacity are not yet completely determined, but there is consensus that corneal transparency is related to the shape, size and organisation of the stromal extracellular matrix and its elements, in particular collagen fibrils and their refractive indices, which translate the speed of light as it passes through the medium in question (see [9], [11], [21] and the references therein).

To model the incidence and reflection of light on the cornea, we consider Maxwell's equations, which describe the electromagnetic field. In this thesis we will focus on Maxwell's equations in time-harmonic form, and consequently, its formulation as the Helmholtz equation.

The most common techniques for solving partial derivative equations are: the finite element method, the finite difference method and the finite volume method. The discontinuous Galerkin

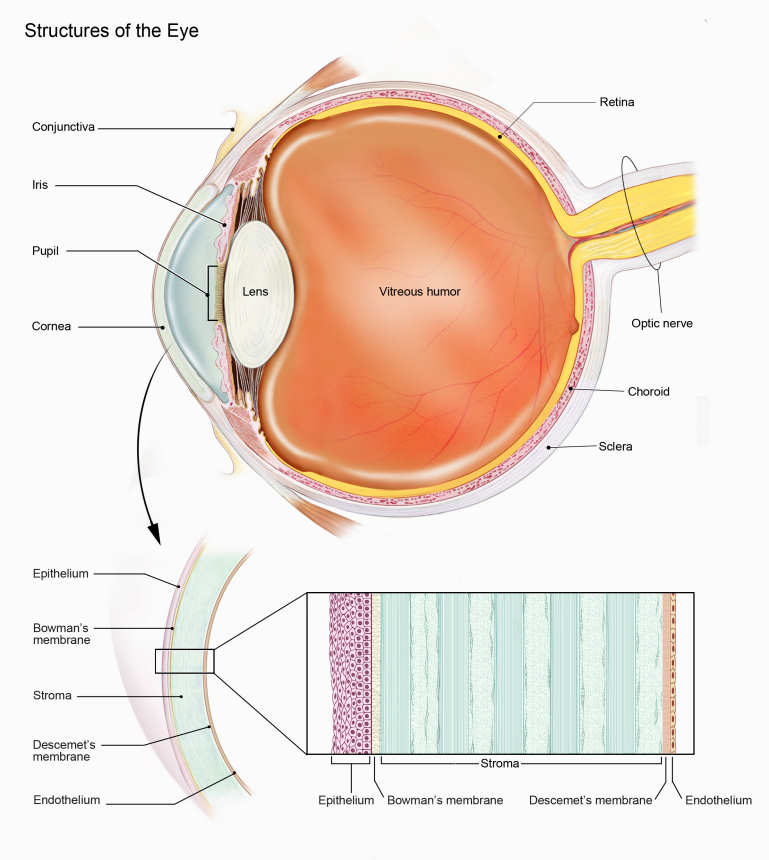


Fig. 1.1 Anatomy of the human eye with corneal cross-section.
Source: National Eye Institute.

(DG) finite element method, which we will use to solve the Helmholtz equation, results from the combination of the ideas used in the finite volume and finite element methods, taking much advantage of the individual advantages of each of them [15]. The DG method is a high-order precision method. Moreover, it is a local method, which allows more flexibility when complex meshes are considered and admits discontinuous solutions. Analogously to the numerical flow in the finite volume method, which transports information from one local element to another, the numerical flow in the DG method connects adjacent elements, allowing the construction of the global approximation.

Since we intend to solve the equation in a domain that mimics the human cornea, polygonal meshes do not exactly fit the curved physical domain, thus reducing the accuracy of the method. In order to overcome this problem, we will equip our numerical scheme with an optimisation method. In this thesis we suggest two alternatives based on changing the boundary condition imposed on the computational domain. One of the approaches is based on a polynomial reconstruction developed in the context of the finite volume method, following the work developed in [2]. The main idea of this method is to design a polynomial reconstruction of the boundary condition of the polygonal computational domain that takes into account the boundary condition in the physical domain. The second approach aims to overcome a drawback of the first alternative, improving the efficiency of the method. It is based on changing the boundary condition imposed on the computational domain by solving an unconstrained minimisation problem. This minimisation problem is solved with variant of

the Nelder-Mead (NM) method, which is one of the most popular derivative-free methods. Despite being a widely used algorithm, quite a few results are known on the convergence of this method. Moreover, the method may fail to converge to a stationary point of the objective function f because the simplices can became arbitrarily flat or needle shaped . In this sense, we suggest a modified Nelder-Mead algorithm that controls the geometry of the simplex.

In Chapter 2, we start by presenting the Maxwell's equations as a fundamental set of equations to describe electromagnetic wave propagation. Moreover, we deduce the Helmholtz's equation in the context of electromagnetism, under certain conditions, and we analyze the existence and uniqueness of the solution of the Helmholtz problem. Once presented the equation of interest, Chapter 3 is dedicated to a description of the numerical method used (the DG method) and of some aspects of the numerical implementation of the method. In Chapter 4, we discuss the treatment of curved boundary domains, highlighting the necessity for an alternative approach, which is proposed in this chapter. We end this chapter with a comparison of the numerical results obtained with the classical DG method and with the alternative method. In Chapter 5, we present another approach to overcome the difficulties in the boundary treatment of curved domains and we analyze some theoretical aspects of the method used to solve the problem. We also present the numerical experiments with this method. Finally, in the last chapter, we present a brief conclusion and perspectives for future work. We also include a comparison of the methods suggested in this thesis in Appendix A.

Chapter 2

Helmholtz's equation in electromagnetism

Maxwell's equations are essential to explain all chemical and biological phenomena that involve interactions between atoms. These equations describe how the electromagnetic field propagates in free space and any medium. In this thesis, we consider an isotropic, linear, and time-invariant medium and we deduce an equation directly related to Maxwell's equations, which is the Helmholtz equation.

In Section 2.1, we present Maxwell's equation and their constitutive relations. Furthermore, we discuss the two-dimensional reduction of these equations, considering the so-called transverse electric (TE) mode. Afterwards, in Section 2.2, we focus on a time-harmonic variation of the electromagnetic wave propagation which, under certain conditions allows us to reduce Maxwell's equations to the Helmholtz equation. We end the chapter with a brief analysis of the existence and uniqueness of solution for a particular Helmholtz problem.

2.1 Maxwell's equations

Maxwell's equations, formulated by James Clerk Maxwell in 1873, are considered the greatest intellectual event of the 19th century. These equations describe the electromagnetic interaction, one of the four fundamental interactions, in addition to gravitational, strong and weak interactions. The electromagnetic interaction relates the electric and magnetic fields to their sources, which are electric charges and currents, respectively. Maxwell's equations are essential to explain all chemical and biological phenomena involving interactions between atoms.

The work of Maxwell was based on Faraday's discoveries, which had already shown that a time-varying of a magnetic field produces an electric field. Starting from physical arguments but also observable, Maxwell proved that the opposite was also true, i.e., that a time-varying of an electric field produced a magnetic field. In addition, also showed that it is possible to create a self-sustained electromagnetic pulse, which propagates at the speed of light, concluding that light is simply a form of electromagnetic radiation. Maxwell described all these phenomena in only four equations which

can be written in the differential form for fields in a continuous medium $\Omega \subset \mathbb{R}^3$ [16]:

$$\frac{\partial \boldsymbol{D}}{\partial t} = \nabla \times \boldsymbol{H} - \boldsymbol{J}, \tag{2.1a}$$

$$\frac{\partial \boldsymbol{B}}{\partial t} = -\nabla \times \boldsymbol{E}, \tag{2.1b}$$

$$\nabla \cdot \boldsymbol{D} = \rho, \tag{2.1c}$$

$$\nabla \cdot \boldsymbol{B} = 0, \tag{2.1d}$$

where $\boldsymbol{E}(\boldsymbol{x},t)$ [V m^{-1}] denotes the electric field intensity, $\boldsymbol{H}(\boldsymbol{x},t)$ [A m^{-1}] the magnetic fiel intensity, $\boldsymbol{D}(\boldsymbol{x},t)$ [A s m^{-2}] the electric displacement field (electric flux), $\boldsymbol{B}(\boldsymbol{x},t)$ [V s m^{-2}] the magnetic induction field (magnetic flux), $\rho(\boldsymbol{x},t)$ [A s m^{-3}] the charge density and $\boldsymbol{J}(\boldsymbol{x},t)$ [A m^{-2}] the current density function. The SI units denotes meter [m], seconds [s], Volt [V] and Ampere [A].

In defining Maxwell's equations, we used the divergence operator $\nabla\cdot$, which is defined for a vector field $\boldsymbol{u} = (u_x, u_y, u_z)$, $u_i = u_i(x,y,z)$, $i = x,y,z$, by

$$\nabla \cdot \boldsymbol{u} = \frac{\partial u_x}{\partial x} + \frac{\partial u_y}{\partial y} + \frac{\partial u_z}{\partial z}.$$

We also used the curl operator defined by

$$\nabla \times \boldsymbol{u} = \begin{vmatrix} \hat{\boldsymbol{i}} & \hat{\boldsymbol{j}} & \hat{\boldsymbol{k}} \\ \frac{\partial}{\partial x} & \frac{\partial}{\partial y} & \frac{\partial}{\partial z} \\ u_x & u_y & u_z \end{vmatrix} = \left(\frac{\partial u_z}{\partial y} - \frac{\partial u_y}{\partial z}, \frac{\partial u_x}{\partial z} - \frac{\partial u_z}{\partial x}, \frac{\partial u_y}{\partial x} - \frac{\partial u_x}{\partial y} \right)^T,$$

where $\hat{\boldsymbol{i}}$, $\hat{\boldsymbol{j}}$, $\hat{\boldsymbol{k}}$ are the unit vectors for the $x-$, $y-$ and $z-$axes, respectively. In this thesis we will also need the definitions of gradient and Laplacian. If u is a scalar field, the gradient is defined by

$$\nabla u = \left(\frac{\partial u}{\partial x}, \frac{\partial u}{\partial y}, \frac{\partial u}{\partial z} \right)^T$$

and the Laplacian by

$$\nabla^2 u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2}.$$

The four Maxwell's equations are the basics of electricity and magnetism in differential form. Equation (2.1b) is the differential form of Faraday's law for induction and describes the creation of an induced electric field due to a time-varying magnetic flux. The creation of an induced magnetic field due to charge flow is described by Equation (2.1a) known as Ampère's law. The divergence equations (2.1c)) and (2.1d) are Gauss's electric law and Gauss's magnetic law, respectively. Equation (2.1c) describes the relation between the electric field distribution and the charge distribution. Equation (2.1d) is a statement of the absence of free magnetic monopoles. The equation (2.2) is known as continuity equation. Equations (2.1b) and (2.1a) are also called curl equations, and equations (2.1c) and (2.1d) are divergence equations. Differentiating (2.1c) with respect to time and using (2.1b) gives

$$\frac{\partial \rho}{\partial t} + \nabla \cdot J = 0 \tag{2.2}$$

which expresses the conservation of the charge of the system.

Although, the Maxwell's equations fully describe the propagation of electromagnetic radiation in any medium, they are not sufficient to determine the electromagnetic field in matter and additional relations known as constitutive equations are needed to model the electromagnetic field interaction with matter. To uniquely determine the electromagnetic field, Maxwell's equations must be supplemented by relations that describe the way fields interact with the matter. These relationships are called constitutive relations and, for linear materials, they can be written as $\boldsymbol{D} = \varepsilon \boldsymbol{E}$ and $\boldsymbol{B} = \mu \boldsymbol{H}$, where ε e μ are material's permittivity and permeability, respectively, and characterize the response of this material to electrical and magnetic fields. The current density, $\boldsymbol{J}$, is typically assumed to be related to the electric field, $\boldsymbol{E}$, through Ohm's law, $\boldsymbol{J} = \sigma \boldsymbol{E}$, where σ is the material's conductivity. Thus, this three parameters fully characterize the electromagnetic properties of a medium.

A usual assumption in optical problems is to consider the electrical conductivity and the charge density as zero, i.e., $\sigma = 0$ e $\rho = 0$. We also assume that the propagation material is isotropic[1], time-invariant and linear In this case, the values of ε and μ are scalar functions that depend only on the spatial variable. In the vacuum, the values of these quantities are fundamental physical constants given, in SI units, by $\mu_0 = 4\pi \times 10^{-7}$ kg m^{2}s^{-2}A^{-2} and $\varepsilon_0 = 1/\mu_0 c_0^2 = 8.85 \times 10^{-12}$ A^{2}s^{4}kg^{-1}m^{-2}, where c_0 denotes the speed of light in a vacuum.

In this work, we consider the non-dimensional variables $\boldsymbol{x} = \boldsymbol{x}/L$ e $t = t/(L/c_0)$, where L is a reference length, and the constitutive relations of the form

$$\boldsymbol{D} = \varepsilon_r \boldsymbol{E}, \quad \boldsymbol{B} = \mu_r \boldsymbol{H}, \tag{2.3}$$

where ε_r and μ_r refer to the relative permittivity and relative permeability, respectively, given by $\varepsilon_r = \varepsilon/\varepsilon_0$ and $\mu_r = \mu/\mu_0$. The relative permittivity is the permittivity of a material in relation to the permittivity of free space and the relative permeability is defined similarly. Under these conditions, the normalized time-domain form of Maxwell's equations is given by

$$\varepsilon_r \frac{\partial \boldsymbol{E}}{\partial t} = \nabla \times \boldsymbol{H}, \tag{2.4}$$

$$\mu_r \frac{\partial \boldsymbol{H}}{\partial t} = -\nabla \times \boldsymbol{E}, \tag{2.5}$$

$$\nabla \cdot \varepsilon_r \boldsymbol{E} = 0, \quad \nabla \cdot \mu_r \boldsymbol{H} = 0. \tag{2.6}$$

If the continuity equation (2.2) holds, the two divergence equations (2.6) are implicitly satisfied. The electromagnetic wave propagation in such a medium formulated as a set of first order coupled differential equations has the form

$$\varepsilon_r \frac{\partial \boldsymbol{E}}{\partial t} = \nabla \times \boldsymbol{H},$$

$$\mu_r \frac{\partial \boldsymbol{H}}{\partial t} = -\nabla \times \boldsymbol{E},$$

where $\boldsymbol{E} = (E_x, E_y, E_z)$, $\boldsymbol{H} = (H_x, H_y, H_z)$.

[1] A material is isotropic if the optical properties are the same for any direction, at any given spatial location in that medium.

In order to reduce the dimensionality of the system, it is usual to assume, in the context of optical applications [13], that the electromagnetic fields is homogeneous in one of its directions, e.g. z-direction. Thus, all z-derivatives vanish and we obtain two disjunctive sets of equations called transverse electric (TE) mode and transverse magnetic (TM) mode. We will consider the mode that describes the propagation where the electric field lies in the plane of propagation. In this case, the Maxwell's curl-equations (2.4) and (2.5) are reduced to the TE mode, where $\boldsymbol{E} = (E_x, E_y)$ and $\boldsymbol{H} = H_z$

$$\varepsilon_r \frac{\partial E_x}{\partial t} = \frac{\partial H_z}{\partial y}, \qquad \varepsilon_r \frac{\partial E_y}{\partial t} = -\frac{\partial H_z}{\partial x}, \qquad \mu_r \frac{\partial H_z}{\partial t} = \frac{\partial E_x}{\partial y} - \frac{\partial E_y}{\partial x}. \tag{2.7}$$

The equations must be completed with appropriated boundary conditions.

2.2 Helmholtz's equation

In this section we focus on a time-harmonic variation of the electromagnetic wave propagation and we deduce the Helmholtz equation. Moreover, we analyze the existence and uniqueness of the solution of a Helmholtz problem with homogeneous Dirichlet boundary conditions.

2.2.1 From Maxwell to Helmholtz

Let us assume that the electric and magnetic fields are time-harmonic, i.e., they can be written as

$$\boldsymbol{E}(\boldsymbol{x},t) = e^{i\omega t}\hat{\boldsymbol{E}}(\boldsymbol{x}) \quad \text{and} \quad \boldsymbol{H}(\boldsymbol{x},t) = e^{i\omega t}\hat{\boldsymbol{H}}(\boldsymbol{x}),$$

where ω denotes the time frequency of the electromagnetic wave, i is the imaginary unit, and $\hat{\boldsymbol{E}}$ and $\hat{\boldsymbol{H}}$ are complex-valued functions. With this assumption, we can write the equations (2.4) and (2.5) as follows

$$\begin{aligned}
\varepsilon_r \frac{\partial e^{i\omega t}\hat{\boldsymbol{E}}}{\partial t} &= \nabla \times \left(e^{i\omega t}\hat{\boldsymbol{H}}\right) \Rightarrow e^{i\omega t} i\omega\varepsilon_r \hat{\boldsymbol{E}} = e^{i\omega t}\left(\nabla \times \hat{\boldsymbol{H}}\right) \Rightarrow i\omega\varepsilon_r \hat{\boldsymbol{E}} = \nabla \times \hat{\boldsymbol{H}}, \\
\mu_r \frac{\partial e^{i\omega t}\hat{\boldsymbol{H}}}{\partial t} &= -\nabla \times \left(e^{i\omega t}\hat{\boldsymbol{E}}\right) \Rightarrow e^{i\omega t} i\omega\mu_r \hat{\boldsymbol{H}} = -e^{i\omega t}\left(\nabla \times \hat{\boldsymbol{E}}\right) \Rightarrow i\omega\mu_r \hat{\boldsymbol{H}} = -\nabla \times \hat{\boldsymbol{E}}.
\end{aligned}$$

We also have,

$$\begin{aligned}
\nabla \cdot \left(\varepsilon_r e^{i\omega t}\hat{\boldsymbol{E}}\right) = 0 \Rightarrow e^{i\omega t}\nabla \cdot \left(\varepsilon_r \hat{\boldsymbol{E}}\right) = 0 \Rightarrow \nabla \cdot \varepsilon_r \hat{\boldsymbol{E}} = 0, \\
\nabla \cdot \left(\mu_r e^{i\omega t}\hat{\boldsymbol{H}}\right) = 0 \Rightarrow e^{i\omega t}\nabla \cdot \left(\mu_r \hat{\boldsymbol{H}}\right) = 0 \Rightarrow \nabla \cdot \mu_r \hat{\boldsymbol{H}} = 0,
\end{aligned}$$

leading to the first order form of Maxwell's equations in the frequency domain

$$i\omega\varepsilon_r \hat{\boldsymbol{E}} = \nabla \times \hat{\boldsymbol{H}}, \quad i\omega\mu_r \hat{\boldsymbol{H}} = -\nabla \times \hat{\boldsymbol{E}}, \tag{2.8}$$

$$\nabla \cdot \varepsilon_r \hat{\boldsymbol{E}} = 0, \quad \nabla \cdot \mu_r \hat{\boldsymbol{H}} = 0. \tag{2.9}$$

The previous equations are now recovered as

$$i\omega\varepsilon_r\hat{\boldsymbol{E}} = \nabla\times\hat{\boldsymbol{H}} \Rightarrow i\omega\nabla\times\hat{\boldsymbol{E}} = \nabla\times\frac{1}{\varepsilon_r}\nabla\times\hat{\boldsymbol{H}} \Rightarrow \nabla\times\frac{1}{\varepsilon_r}\nabla\times\hat{\boldsymbol{H}} = \mu_r\omega^2\hat{\boldsymbol{H}},$$
$$i\omega\mu_r\hat{\boldsymbol{H}} = -\nabla\times\hat{\boldsymbol{E}} \Rightarrow i\omega\nabla\times\hat{\boldsymbol{H}} = -\nabla\times\frac{1}{\mu_r}\nabla\times\hat{\boldsymbol{E}} \Rightarrow \nabla\times\frac{1}{\mu_r}\nabla\times\hat{\boldsymbol{E}} = \varepsilon_r\omega^2\hat{\boldsymbol{E}}.$$

Therefore, equations (2.8) and (2.9) can be written as

$$\nabla\times\frac{1}{\varepsilon_r}\nabla\times\hat{\boldsymbol{H}} = \mu_r\omega^2\hat{\boldsymbol{H}}, \quad \nabla\cdot\mu_r\hat{\boldsymbol{H}} = 0 \tag{2.10}$$
$$\nabla\times\frac{1}{\mu_r}\nabla\times\hat{\boldsymbol{E}} = \varepsilon_r\omega^2\hat{\boldsymbol{E}}, \quad \nabla\cdot\varepsilon_r\hat{\boldsymbol{E}} = 0. \tag{2.11}$$

We refer to these equations as the second-order curl-curl form. If we consider a homogeneous material, i.e., ε_r and μ_r are not space-dependent, and if we apply the following identity

$$\nabla\times\nabla\times\boldsymbol{u} = \nabla(\nabla\cdot\boldsymbol{u}) - \nabla^2\boldsymbol{u},$$

where $\boldsymbol{u} = (\hat{\boldsymbol{E}},\hat{\boldsymbol{H}})$, equations (2.10) and (2.11) both reduce to the Helmholtz equation

$$-\nabla^2\begin{bmatrix}\hat{\boldsymbol{E}}(\boldsymbol{x})\\ \hat{\boldsymbol{H}}(\boldsymbol{x})\end{bmatrix} = \omega^2\mu_r\varepsilon_r\begin{bmatrix}\hat{\boldsymbol{E}}(\boldsymbol{x})\\ \hat{\boldsymbol{H}}(\boldsymbol{x})\end{bmatrix}.$$

Now consider the TE-mode Maxwell's equations in 2D (2.7) with a time-harmonic variation in an homogeneous media. Note that, in this case, we have $\hat{\boldsymbol{E}} = (\hat{E}_x,\hat{E}_y)$, $\hat{\boldsymbol{H}} = \hat{H}_z$ and

$$\hat{E}_x = \frac{-i}{\omega\varepsilon_r}\frac{\partial\hat{H}_z}{\partial y} \Rightarrow \frac{\partial\hat{E}_x}{\partial y} = \frac{-i}{\omega\varepsilon_r}\frac{\partial^2\hat{H}_z}{\partial y^2}$$
$$\hat{E}_y = \frac{i}{\omega\varepsilon_r}\frac{\partial\hat{H}_z}{\partial x} \Rightarrow \frac{\partial\hat{E}_y}{\partial x} = \frac{i}{\omega\varepsilon_r}\frac{\partial^2\hat{H}_z}{\partial x^2}$$
$$\hat{H}_z = \frac{-i}{\omega\mu_r}\left(\frac{\partial\hat{E}_x}{\partial y} - \frac{\partial\hat{E}_y}{\partial x}\right).$$

Thus

$$\hat{H}_z = \frac{-i}{\omega\mu_r}\left(\frac{-i}{\omega\varepsilon_r}\frac{\partial^2\hat{H}_z}{\partial y^2} - \frac{i}{\omega\varepsilon_r}\frac{\partial^2\hat{H}_z}{\partial x^2}\right) = \frac{-1}{\omega^2\mu_r\varepsilon_r}\left(\frac{\partial^2\hat{H}_z}{\partial x^2} + \frac{\partial^2\hat{H}_z}{\partial y^2}\right),$$

and we get the Helmholtz equation in scalar form,

$$-\nabla^2\hat{H}_z(x,y) = \omega^2\mu_r\varepsilon_r\hat{H}_z(x,y). \tag{2.12}$$

As mentioned above $\hat{H}_z$ is a complex-valued function but, as can be easily seen, the real part of $\hat{H}_z$ also satisfies the same equation. In this thesis we focus on the scalar Helmholtz equation for real-valued functions.

2.2.2 Existence and uniqueness of solution

We will present present a brief study on the existence and uniqueness of the solution of the Helmholtz problem with homogeneous Dirichlet boundary conditions. Let $\Omega \subset \mathbb{R}^2$ be a bounded open set of smooth boundary $\partial\Omega$ and $\boldsymbol{x} = (x, y)$. Our aim is to solve the problem

$$\begin{aligned} -\nabla^2 u(\boldsymbol{x}) - \nu^2(\boldsymbol{x})u(\boldsymbol{x}) &= f(\boldsymbol{x}), \quad \boldsymbol{x} \in \Omega \\ u(\boldsymbol{x}) &= 0, \quad \boldsymbol{x} \in \partial\Omega, \end{aligned} \tag{2.13}$$

where $\nu = \omega\sqrt{\varepsilon_r \mu_r}$ is a strictly positive piecewise continuous real function that depends on the relative permittivity, ε_r, and the relative permeability, μ_r, of the medium, as well as on the (constant) frequency, ω. Let us assume that $\nu^2(\boldsymbol{x}) \leq \nu_{max}^2$, for all $\boldsymbol{x}$, and the source term $f \in L^2(\Omega)$. The Lebesgue space $L^2(\Omega)$ is defined as a space of mensurable functions $u : \Omega \to \mathbb{R}$ such that $||u||^2_{L^2(\Omega)} < +\infty$, equipped with norm $||u||^2_{L^2(\Omega)} = (u,u)_{L^2(\Omega)}$ and inner product

$$(u,w)_{L^2(\Omega)} = \int_\Omega u(\boldsymbol{x})w(\boldsymbol{x}) \; d\boldsymbol{x}.$$

Our goal is to find conditions that guarantee that (2.13) has a unique weak solution. We will use variational arguments and classical results from the theory of partial differential equations.

Let w be a sufficiently smooth test function that vanishes on $\partial\Omega$. We multiply (2.13) by w, and we integrate over Ω

$$-\int_\Omega \nabla^2 u(\boldsymbol{x})w(\boldsymbol{x}) \; d\boldsymbol{x} - \int_\Omega \nu^2(\boldsymbol{x})u(\boldsymbol{x})w(\boldsymbol{x}) \; d\boldsymbol{x} = \int_\Omega f(\boldsymbol{x})w(\boldsymbol{x}) \; d\boldsymbol{x}.$$

Integrating by parts and taking w satisfying the same boundary conditions as u, yields

$$\int_\Omega \nabla u(\boldsymbol{x}) \cdot \nabla w(\boldsymbol{x}) \; d\boldsymbol{x} - \int_\Omega \nu^2(\boldsymbol{x})u(\boldsymbol{x})w(\boldsymbol{x}) \; d\boldsymbol{x} = \int_\Omega f(\boldsymbol{x})w(\boldsymbol{x}) \; d\boldsymbol{x}.$$

The functions u and w need to have a first-order weak derivative, u satisfies the boundary conditions and w vanishes on $\partial\Omega$ in the sense of trace. Therefore, $u, w \in H_0^1(\Omega)$ where

$$H_0^1(\Omega) = \left\{ w \in L^2(\Omega) : \frac{\partial w}{\partial x}, \frac{\partial w}{\partial y} \in L^2(\Omega) \text{ and } w|_{\partial\Omega} = 0 \right\}.$$

Its norm given as

$$||w||_{H^1(\Omega)} = \left(||w||^2_{L^2(\Omega)} + |w|^2_{H^1(\Omega)} \right)^{1/2},$$

where $|w|_{H^1(\Omega)}$ denotes the semi-norm defined as

$$|w|^2_{H^1(\Omega)} = (\nabla w, \nabla w)_{L^2(\Omega)}.$$

Thus, the variational problem (2.13) can be reformulated as follows: find $u \in H_0^1(\Omega)$ such that

$$a(u,w) = (f,w)_{L^2(\Omega)}, \quad \forall w \in H_0^1(\Omega), \tag{2.14}$$

where the bilinear form $a(\cdot,\cdot)$ is defined as

$$a(u,w) = (\nabla u, \nabla w)_{L^2(\Omega)} - (\nu u, w)_{L^2(\Omega)}.$$

To prove the existence and uniqueness of (2.13), we use the Lax-Milgram theorem (whose proof can be seen in [31]) to show that there is a unique $u \in H_0^1(\Omega)$ that satisfies (2.14).

Theorem 2.2.1 (Lax-Milgram). *Let V be a Hilbert space with norm $||\cdot||_V$, $a(\cdot,\cdot)$ a bilinear form on $V \times V$ and $\ell(\cdot)$ a bounded linear functional on V such that $a(\cdot,\cdot)$ is coercive – there is $c_0 > 0$ such that $a(w,w) \geq c_0||w||_V^2$, $\forall w \in V$ and $a(\cdot,\cdot)$ is bounded – there is $c_1 > 0$ such that $|a(w_1,w_2)| \leq c_1||w_1||_V||w_2||_V$, $\forall w_1, w_2 \in V$. Then, there exists a unique $u \in V$ such that $a(u,w) = \ell(w)$, $\forall w \in V$.*

It can be easily proved that $a(\cdot,\cdot)$ is a symmetric bilinear form on $H_0^1(\Omega) \times H_0^1(\Omega)$ and that $\ell(\cdot)$ is linear and bounded on $H_0^1(\Omega)$. First, we will show that $a(\cdot,\cdot)$ is bounded. Considering $u, v \in H_0^1(\Omega)$ and using the Cauchy-Schwarz inequality

$$\begin{aligned} |a(u,w)| &= \left|(\nabla u, \nabla w)_{L^2(\Omega)} - (\nu^2 u, w)_{L^2(\Omega)}\right| \leq \left|(\nabla u, \nabla w)_{L^2(\Omega)}\right| + \left|(\nu^2 u, w)_{L^2(\Omega)}\right| \\ &\leq ||\nabla u||_{L^2(\Omega)}||\nabla w||_{L^2(\Omega)} + \nu_{max}^2||u||_{L^2(\Omega)}||w||_{L^2(\Omega)}. \end{aligned}$$

Taking into account the definition of norms and considering $c_1 = 1 + \nu_{max}^2$, we conclude that the bilinear form is bounded since

$$|a(u,w)| \leq ||u||_{H^1(\Omega)}||w||_{H^1(\Omega)} + \nu_{max}^2||u||_{H^1(\Omega)}||w||_{H^1(\Omega)} = c_1||u||_{H^1(\Omega)}||w||_{H^1(\Omega)}.$$

To prove the coercivity, we use the Poincaré-Friedrichs' inequality

$$||u||_{L^2(\Omega)} \leq c_* |u|_{H^1(\Omega)}, \quad \forall u \in H_0^1(\Omega),$$

whose proof can be found in [10]. Considering $u \in H_0^1(\Omega) \setminus \{0\}$, we have

$$\frac{1}{c_*^2} \leq \lambda_{min} = \inf_{u \in H_0^1(\Omega)\setminus\{0\}} \frac{|u|_{H^1(\Omega)}^2}{||u||_{L^2(\Omega)}^2}.$$

Now, since $|u|_{H^1(\Omega)} \neq 0$, we have

$$\begin{aligned} a(u,u) &= (\nabla u, \nabla u)_{L^2(\Omega)} - (\nu^2 u, u)_{L^2(\Omega)} \geq |u|_{H^1(\Omega)}^2 - \nu_{max}^2 \frac{||u||_{L^2(\Omega)}^2}{|u|_{H^1(\Omega)}^2}|u|_{H^1(\Omega)}^2 \\ &\geq |u|_{H^1(\Omega)}^2 - \frac{\nu_{max}^2}{\lambda_{min}}|u|_{H^1(\Omega)}^2 = \left(1 - \frac{\nu_{max}^2}{\lambda_{min}}\right)\left(\frac{1}{2}|u|_{H^1(\Omega)}^2 + \frac{1}{2}|u|_{H^1(\Omega)}^2\right). \end{aligned}$$

If we consider $\nu_{max}^2 < \lambda_{min}$, using the Poincaré-Friedrichs' inequality

$$\begin{aligned} a(u,u) &\geq \left(1 - \frac{\nu_{max}^2}{\lambda_{min}}\right)\left(\frac{1}{2}|u|_{H^1(\Omega)}^2 + \frac{1}{2c_*^2}||u||_{L^2(\Omega)}^2\right) \\ &\geq \left(1 - \frac{\nu_{max}^2}{\lambda_{min}}\right)\min\left\{\frac{1}{2}, \frac{1}{2c_*^2}\right\}||u||_{H^1(\Omega)}^2 = c_0||u||_{H^1(\Omega)}^2 \end{aligned}$$

where

$$c_0 = \left(1 - \frac{\nu_{max}^2}{\lambda_{min}}\right) \min\left\{\frac{1}{2}, \frac{1}{2c_*^2}\right\} > 0.$$

Thus, if $\nu_{max}^2 < \lambda_{min}$, by the Lax-Milgram theorem, there exists a unique $u \in H_0^1(\Omega)$ such that $a(u,w) = (f,w)_{L^2(\Omega)}, \quad \forall w \in H_0^1(\Omega)$. Furthermore, considering the Cauchy-Schwarz inequality, we have

$$c_0||u||_{H^1(\Omega)}^2 \leq a(u,u) = (f,u)_{L^2(\Omega)} \leq |(f,u)_{L^2(\Omega)}| \leq ||u||_{L^2(\Omega)}||f||_{L^2(\Omega)} \leq ||f||_{L^2(\Omega)}||u||_{H^1(\Omega)},$$

which allows to conclude that

$$||u||_{H^1(\Omega)} \leq \frac{1}{c_0}||f||_{L^2(\Omega)}$$

and therefore the problem (2.14) is well posed.

Note that if $\nu_{max}^2 \geq \lambda_{min}$ the coercivity is not fulfilled and therefore the Lax-Milgram Theorem cannot be applied. However, if we consider the Helmholtz problem with impedance boundary conditions [22, 27], the existence and uniqueness of solution can be prove using the Fredholm theory and the Gårding's inequality .

In the remaining part of this thesis, we will consider the Helmholtz problem homogeneous Dirichlet boundary conditions with sufficiently small scalar wave number, i.e., $\nu^2(\boldsymbol{x}) = \nu^2$, for all $\boldsymbol{x} \in \Omega$, such that $\nu^2 < \lambda_{min}$.

Chapter 3

Discontinuous Galerkin finite element method

In this chapter we will describe the application of the discontinuous Galerkin (DG) finite element method [15] to the Helmholtz problem with a sufficiently small (constant) wave number and Dirichlet boundary conditions, considering a polygonal computational domain. The DG finite element method seems to have been introduced as a way to solve the neutron transport equation, in 1973 [25]. Since then, there has been significant developments in the extensions of this method leading to numerous applications, particularly in the fields of acoustics, electromagnetism and fluid dynamics. With the growing need to solve geometrically complex large-scale problems, this method has gained relevance since it gathers many desirable features. The DG method admits discontinuous solutions and it is a method with a high order of accuracy. Moreover, being a local method it allows great flexibility when considering complex meshes. Analogously to the finite volume method, a main ingredient of the DG scheme is the numerical flow which transports information from one local element to another, connecting adjacent elements and allowing to build the global approximation.

We begin, in Section 3.1, by presenting the DG formulation for the 2D Helmholtz equation on a polygonal domain. Section 3.2 is devoted to the discussion of some details of the implementation of the algorithm, which are responsible for the efficiency and robustness of the DG method. In particular, we analyse the issue of mesh generation and the tools needed for polynomial interpolation. Finally, in Section 3.3, we apply the method to particular problem for which its exact solution is known and analyse its order of convergence.

3.1 Discontinuous Galerkin formulation for the Helmholtz equation

Let $\Omega \subset \mathbb{R}^2$ be a bounded open set of smooth boundary $\partial\Omega$ and $\boldsymbol{x} = (x, y)$. We want to solve the problem

$$\begin{aligned} -\nabla^2 u(\boldsymbol{x}) - \nu^2 u(\boldsymbol{x}) &= f(\boldsymbol{x}), \quad \boldsymbol{x} \in \Omega \\ u(\boldsymbol{x}) &= 0, \quad \boldsymbol{x} \in \partial\Omega, \end{aligned} \tag{3.1}$$

with a sufficiently small (constant) wave number.

Assume that the physical domain Ω can be approximated by the computational domain Ω_h, represented as a union of K non-overlapping straight-sided triangles $T^k, k = 1, ..., K$, i.e.,

$$\Omega \approx \Omega_h = \bigcup_{k=1}^{K} T^k. \tag{3.2}$$

The triangulation $\mathscr{T}_h = \left\{T^k, k = 1, ..., K\right\}$ is assumed to be geometrically conforming, that is the intersection of two elements is either a complete edge, a vertex or the empty set. The parameter h represents the maximum element diameter, i.e.,

$$h = \max_{T^k \in \mathscr{T}_h} \{h_k\}, \qquad h_k = \sup_{P_1, P_2 \in T^k} ||P_1 - P_2||.$$

We also assume that the triangulation is sufficiently smooth [13] in the sense that we can define a upper boundary for the quotient between h_k and the maximum diameter of a ball inscribed in each element $T^k \in \mathscr{T}_h$.

The numerical approximation u_h to the exact solution u of the Helmholtz problem (3.1) is defined in the following way

$$u_h(\boldsymbol{x}) = \bigoplus_{k=1}^{K} u_h^k(\boldsymbol{x}) \in \mathscr{V}_h, \tag{3.3}$$

where

$$\mathscr{V}_h = \{u_h \in L^2(\Omega_h) : u_h^k = u_h|_{T^k} \in \mathscr{P}_N(T^k), \forall T^k \in \mathscr{T}_h\},$$

and $\mathscr{P}_N(T^k)$ denotes the space of polynomials of degree less than or equal to N on T^k. Then, on each element T^k, the local solution u_h^k can be expressed as a polynomial of degree N

$$\boldsymbol{x} \in T^k \in \mathscr{T}_h : \quad u_h^k(\boldsymbol{x}) = \sum_{i=1}^{N_p} u_h^k(\boldsymbol{x}_i^k) \ell_i^k(\boldsymbol{x}), \tag{3.4}$$

where $\ell_i^k(\boldsymbol{x})$ is the multidimensional Lagrange polynomial defined by the grid points $\boldsymbol{x}_i^k$, $i = 1, ..., N_p$, $N_p = (N+1)(N+2)/2$, on T^k, which can be determined using an optimised explicit Warp & Blend construction procedure [29].

We introduce a new vector function $\boldsymbol{q} = (q_x, q_y)^T$, such that $\boldsymbol{q} = \nabla u$. Thus, the Helmholtz equation may be written by $-\nabla \cdot \boldsymbol{q} - \nu^2 u = f$, where the local solution for the auxiliary variables can be expressed as $\boldsymbol{q}_h^k = (q_{h,x}^k, q_{h,y}^k)^T$ with

$$\boldsymbol{x} \in T^k \in \mathscr{T}_h : \quad q_{h,j}^k(\boldsymbol{x}) = \sum_{i=1}^{N_p} q_{h,j}^k(\boldsymbol{x}_i^k) \ell_i^k(\boldsymbol{x}), \quad j = x, y. \tag{3.5}$$

To define the numerical method, we require that the local residuals given by $-\nabla \cdot \boldsymbol{q}_h^k(\boldsymbol{x}) - \nu^2 u_h^k(\boldsymbol{x}) - f(\boldsymbol{x})$ and $\boldsymbol{q}(\boldsymbol{x}) - \nabla u(\boldsymbol{x})$ are orthogonal to the Lagrange polynomials with respect to the inner product on $\mathscr{V}_h$ given by

$$(u, v)_{L^2(\Omega_h)} = \sum_{k=1}^{K} (u, v)_{L^2(T^k)},$$

where $(u,v)_{L^2(T^k)}$ denotes the usual inner product on $L^2(T^k)$, as defined in the previous chapter. Therefore, we have

$$-\left(\ell_i^k, \nabla\cdot \boldsymbol{q}_h^k\right)_{L^2(T^k)} - \nu^2\left(\ell_i^k, u_h^k\right)_{L^2(T^k)} = \left(\ell_i^k, f\right)_{L^2(T^k)}. \tag{3.6}$$

and

$$\left(\ell_i^k, \cdot\boldsymbol{q}_h^k\right)_{L^2(T^k)} - \left(\ell_i^k, \nabla u_h^k\right)_{L^2(T^k)} = 0. \tag{3.7}$$

Integrating (3.6) by parts yields

$$-\int_{\partial T^k} \hat{\boldsymbol{n}}\cdot\boldsymbol{q}_h^k(\boldsymbol{x})\ell_i^k(\boldsymbol{x})\ d\boldsymbol{x} + \left(\nabla\ell_i^k, \boldsymbol{q}_h^k\right)_{L^2(T^k)} - \nu^2\left(\ell_i^k, u_h^k\right)_{L^2(T^k)} = \left(\ell_i^k, f\right)_{L^2(T^k)},$$

where $\hat{\boldsymbol{n}}$ is the outward unit normal vector pointing from T^k to adjacent element T^l. Now, replacing the physical flux by a numerical flux $\boldsymbol{q}_{h,k}^*$, we obtain the weak form

$$-\int_{\partial T^k} \hat{\boldsymbol{n}}\cdot\boldsymbol{q}_{h,k}^*(\boldsymbol{x})\ell_i^k(\boldsymbol{x})\ d\boldsymbol{x} + \left(\nabla\ell_i^k, \boldsymbol{q}_h^k\right)_{L^2(T^k)} - \nu^2\left(\ell_i^k, u_h^k\right)_{L^2(T^k)} = \left(\ell_i^k, f\right)_{L^2(T^k)}.$$

Integration by parts again, we get the strong form

$$-\left(\nabla\cdot\boldsymbol{q}_h^k, \ell_i^k\right)_{L^2(T^k)} + \int_{\partial T^k}\hat{\boldsymbol{n}}\cdot\left(\boldsymbol{q}_h^k(\boldsymbol{x}) - \boldsymbol{q}_{h,k}^*(\boldsymbol{x})\right)\ell_i^k(\boldsymbol{x})\ d\boldsymbol{x} - \nu^2\left(\ell_i^k, u_h^k\right)_{L^2(T^k)} = \left(\ell_i^k, f\right)_{L^2(T^k)}.$$

The numerical flux is defined considering the internal penalty fluxes given by (see [15])

$$\boldsymbol{q}_{h,k}^* = \{\{\nabla u_h^k\}\} - \tau^k[\![u_h^k]\!], \quad u_{h,k}^* = \{\{u_h^k\}\}. \tag{3.8}$$

The average and jumps along a normal, $\hat{\boldsymbol{n}}$, are defined, respectively, by

$$\{\{u\}\} = \frac{u^- + u^+}{2}, \ [\![u]\!] = \hat{\boldsymbol{n}}^- u^- + \hat{\boldsymbol{n}}^+ u^+, \ [\![\boldsymbol{u}]\!] = \hat{\boldsymbol{n}}^-\cdot\boldsymbol{u}^- + \hat{\boldsymbol{n}}^+\cdot\boldsymbol{u}^+,$$

where the superscripts "$-$" and "$+$" refer to the interior and exterior information, respectively. Note that the jumps along a normal, $\hat{\boldsymbol{n}}$, are defined differently depending if u is a scalar or a vector, $\boldsymbol{u}$, whereas the average is defined in the same way. The parameter τ^k is a real parameter that may depend on the diameter of triangle T^k and the degree of the polynomial defined in that triangle. In [15], authors considered

$$\tau^k \geq C\frac{(N+1)^2}{h}, \quad C \geq 1.$$

For $\tau^k = 0$, equation (3.8) reduces to the central flux. In the numerical tests presented in this thesis, we considered

$$\tau^k = 200\frac{(N+1)^2}{h_k}.$$

Using the same arguments for the equation (3.7) and taking into account (3.4) and (3.5), we get the strong form of the DG method

$$
\begin{aligned}
&-\nabla S^k \boldsymbol{q}_h^k + \int_{\partial T^k} \hat{\boldsymbol{n}} \cdot \left(\boldsymbol{q}_h^k(\boldsymbol{x}) - \boldsymbol{q}_{h,k}^*(\boldsymbol{x}) \right) \boldsymbol{\ell}^k(\boldsymbol{x}) \; d\boldsymbol{x} - \nu^2 M^k \boldsymbol{u}_h^k = M^k \boldsymbol{f}_h, \\
&M^k \boldsymbol{q}_{h,x}^k = S_x^k \boldsymbol{u}_h^k - \int_{\partial T^k} \hat{n}_x \left(\boldsymbol{u}_h^k(\boldsymbol{x}) - \boldsymbol{u}_{h,k}^*(\boldsymbol{x}) \right) \boldsymbol{\ell}^k(\boldsymbol{x}) \; d\boldsymbol{x}, \\
&M^k \boldsymbol{q}_{h,y}^k = S_y^k \boldsymbol{u}_h^k - \int_{\partial T^k} \hat{n}_y \left(\boldsymbol{u}_h^k(\boldsymbol{x}) - \boldsymbol{u}_{h,k}^*(\boldsymbol{x}) \right) \boldsymbol{\ell}^k(\boldsymbol{x}) \; d\boldsymbol{x},
\end{aligned}
$$

where, for $i, j = 1, \ldots, N_p$,

$$
M_{ij}^k = \left(\ell_i^k, \ell_j^k \right)_{L^2(T^k)}, \quad \nabla S^k = \left[S_x^k, S_y^k \right], \quad S_{x,ij}^k = \left(\ell_i^k, \partial \ell_j^k / \partial x \right)_{L^2(T^k)}, \quad S_{y,ij}^k = \left(\ell_i^k, \partial \ell_j^k / \partial y \right)_{L^2(T^k)},
$$

$$
\boldsymbol{u}_h^k = \left[u_h^k(x_j^k, y_j^k) \right]_{j=1}^{N_p}, \quad \boldsymbol{q}_{h,j}^k = \left[q_{h,j}^k(x_n^k, y_n^k) \right]_{n=1}^{N_p}, \; j = x, y, \quad \boldsymbol{\ell}^k(\boldsymbol{x}) = \left[\ell_j^k(\boldsymbol{x}) \right]_{j=1}^{N_p}
$$

We conclude that the DG solution can be obtained by solving a system of linear equations

$$
(-A - \nu^2 B) U = F,
$$

where $U = [\boldsymbol{u}_h^k]_{k=1}^K$, B is the standard block-diagonal mass matrix $N_p K \times N_p K$ (where the diagonal block is the local mass matrix M^k), A is a symmetric block matrix $N_p K \times N_p K$ with the remaining terms on the right hand-side of the equation, where each element with a common edge with the computational boundary contributes with 3 blocks and each inner element contributes with 4 blocks. Furthermore, F denotes the source term with the contribution of the boundary conditions.

Algorithm 1 DG method

1. Mesh generation
2. Determine the matrices A, B e F.
3. Solve the system of linear equations: $(-A - \nu^2 B) U = F$

3.2 Implementation and numerical aspects

In order to bridge the gap between the mathematical formulation and the computational implementation, we discuss some details of the implementation of the algorithm responsible for the efficiency and robustness of the DG method. After giving some ideas related to mesh generation, we present the tools needed for polynomial interpolation over triangles, in particular how the nodes and the local matrices can be determined.

3.2.1 Mesh generation

The mesh $\mathscr{T}_h$ was generated by the free mesh generator Gmsh (version 4.6.0) [12]. This software package allows us to obtain important information for the DG method. When defining the boundary of the domain Ω, we assign a value to a certain boundary condition: for instance, 6 corresponds to

the Dirichlet boundary conditions and 7 to the Neumann boundary conditions. Assume, for example, that we assign the value 6 to the entire boundary $\partial\Omega$. Then, the triangulation is done and from the generated mesh $\mathscr{T}_h$ we know the abscissae and ordinates of the vertices in the grid, which are stored in the vectors VX and VY, respectively. We also know the number of elements, K, and the matrices $BCType$ and $EToV$ needed to compute the connectivities between the elements. These matrices are described below.

The matrix $BCType$, of size $K \times 3$, whose entries are 0 and 6 (in the case of Dirichlet boundary conditions), allows us to identify the elements with a common edge with the computational boundary and, in particular, which side of these triangles is on the computational boundary. If $(BCType)_{ij} \neq 0$, then the element i has common edge with the computational boundary. Moreover, we know that the side j of the element i is located at the computational boundary, $i = 1,\ldots,K$, $j = 1,2,3$, and has a Dirichlet boundary condition.

The matrix $EToV$ (Element-To-Vertice), of size $K \times 3$, contains in each row i the indices of the three vertices that form an element i, $i = 1,\ldots,K$. Furthermore, it is assumed that the three vertices are ordered counterclockwise. From this matrix we can combine the K elements into a continuous region of elements by computing the connectivity of the elements. We determine the following connectivity matrices: $EToE$ and $EToF$. The matrix $EToE$ (Element-To-Element), of size $K \times 3$, allow us to identify the neighbor elements of the element i, on each side of the triangle i and counterclockwise. For instance, if $(EToE)_{ij} = l$ (see Figure 3.1), then the side j of the element i connects to the element l, $i,l = 1,\ldots,K$, $j = 1,2,3$. Moreover, if $(EToE)_{ij} = i$, then side j of the element i self-connects and is therefore taken to be a boundary side. On the other hand, the matrix $EToF$ (Element-To-Face), of size $K \times 3$, identifies the neighbor sides of each side of the triangle i and counterclockwise. E.g., if $(EToF)_{ij} = k$ (see Figure 3.1), then the side j of the element i connects to side k (of the triangle l, considering the previous example). Thus, this matrix has an important role in the flux between elements.

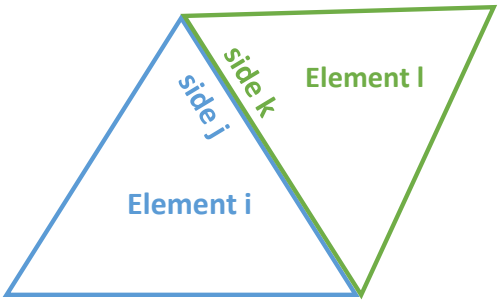


Fig. 3.1 Element-to-element and element-to-face connectivity.

3.2.2 Modes and interpolation nodes in two dimensions

In order to simplify the numerical aspects, we introduce a local coordinate system $\boldsymbol{r} = (r,s)$ and a standard triangle defined as

$$\Delta = \{\boldsymbol{r} = (r,s) \text{ such that } r,s \geq -1, r+s \leq 0\}.$$

We also consider a linear mapping $\boldsymbol{r} = (r,s) \longrightarrow \boldsymbol{x} = (x,y)$ connecting Δ with T^k, whose details can be seen in [15], and the Jacobian matrix of this transformation is given by

$$\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{r}} = \begin{bmatrix} x_r & x_s \\ y_r & y_s \end{bmatrix}.$$

Considering J^k its determinant (the Jacobian),

$$\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{x}} = \begin{bmatrix} r_x & r_y \\ s_x & s_y \end{bmatrix} = \frac{1}{J^k} \begin{bmatrix} y_s & -x_s \\ -y_r & x_r \end{bmatrix}.$$

We can now focus on the development of polynomials and operators defined on the standard triangle Δ. Consider local two-dimensional polynomial basis of order N, $\{\psi_n(\boldsymbol{x})\}_{n=1}^{N_p}$, $N_p = (N+1)(N+2)/2$, we can define the local solution, where we have dropped the superscript Δ to simply the notation, as

$$u_h(\boldsymbol{r}) = \sum_{n=1}^{N_p} \hat{u}_n \psi_n(\boldsymbol{r}) = \sum_{i=1}^{N_p} u_h(\boldsymbol{r}_i)\ell_i(\boldsymbol{r}), \tag{3.9}$$

where the coefficients $\hat{u}_n$, $n = 1,\cdots,N_p$, ar defined such that the approximation is interpolatory on the grids points $\boldsymbol{r}_i$, that is $u(\boldsymbol{r}_i) = u_h(\boldsymbol{r}_i)$. We can now establish the connection between the modes, $\hat{\boldsymbol{u}}$, and the nodal values, $\boldsymbol{u}_h$,

$$V\hat{\boldsymbol{u}} = \boldsymbol{u}_h, \tag{3.10}$$

where $\hat{\boldsymbol{u}} = [\hat{u}_1,\cdots,\hat{u}_{Np}]^T$ are the N_p expansion coefficientes, $\boldsymbol{u}_h = [u_h(\boldsymbol{r}_1),\cdots,u_h(\boldsymbol{r}_{N_p})]^T$ and V is the generalized Vandermonde matrix. Furthermore, with relation (3.10) we are capable to evaluate 2D Lagrange polynomials for which an explicit expression does not exist [15].

Considering (3.9) we get that $\boldsymbol{u}_h^T\boldsymbol{\ell}(\boldsymbol{r}) = \hat{\boldsymbol{u}}^T\psi(\boldsymbol{r})$ and, taking into account (3.10), this yields

$$V^T\boldsymbol{\ell}(r) = \boldsymbol{\psi}(r) \Leftrightarrow \begin{bmatrix} \psi_1(\boldsymbol{r}_1) & \psi_1(\boldsymbol{r}_2) & \ldots & \psi_1(\boldsymbol{r}_{N_p}) \\ \psi_2(\boldsymbol{r}_1) & \psi_2(\boldsymbol{r}_2) & \ldots & \psi_2(\boldsymbol{r}_{N_p}) \\ \vdots & \vdots & \ldots & \vdots \\ \psi_{N_p}(\boldsymbol{r}_1) & \psi_{N_p}(\boldsymbol{r}_2) & \ldots & \psi_{N_p}(\boldsymbol{r}_{N_p}) \end{bmatrix} \begin{bmatrix} \ell_1(\boldsymbol{r}) \\ {}^2(\boldsymbol{r}) \\ \vdots \\ \ell_{N_p}(\boldsymbol{r}) \end{bmatrix} = \begin{bmatrix} \psi_1(\boldsymbol{r}) \\ \psi_2(\boldsymbol{r}) \\ \vdots \\ \psi_{N_p}(\boldsymbol{r}) \end{bmatrix}. \tag{3.11}$$

To ensure stable numerical behavior of the generalized Vandermonde matrix we need to identify an orthonormal polynomial basis, $\psi_j(\boldsymbol{r})$, defined on the standard triangle Δ and to identify a family of interpolation points. As a first approach, we could consider

$$\psi_m(\boldsymbol{r}) = r^i s^j, \quad (i,j) \geq 0, \quad i+j \leq N, \quad m = j+(N+1)i+1-\frac{i}{2}(i-1), \quad (i,j) \geq 0, \quad i+j \leq N,$$

which spans the space of N–dimensional polynomials in two variables, (r,s). This basis leads to the ill-conditioning of the system and, as such, is not a good choice [15]. Using the Gram-Schmidt process, we obtain the orthonormal basis

$$\psi_m(\boldsymbol{r}) = \sqrt{2}P_i^{(0,0)}(2(1+r)/(1-s))P_j^{(2i+1,0)}(s)(1-s)^i, \tag{3.12}$$

where $P_n^{(\alpha,\beta)}(x)$ is the n–th order Jacobi polynomial. When $\alpha=\beta=0$, we get the Legendre polynomial.

As a first choice to determine the N_p interpolation nodes, equidistant points could be considered. As it is proved in [15], this choice leads to ill-conditioned linear systems. For the one-dimensional case, the Legendre-Gauss-Lobatto (LGL) quadrature points lead to a well-conditioned DG formulation and for higher dimensions the LGL nodes are used to define suitable distributed nodes through the Warp & Blend construction procedure described in [29].

The main idea of this construction procedure is to transform an equidistant grid into a grid that is better suited for interpolation, considering an equilateral triangle with vertices $\left(0,\frac{2}{\sqrt{3}}\right)$, $\left(1,-\frac{1}{\sqrt{3}}\right)$ and $\left(-1,-\frac{1}{\sqrt{3}}\right)$. For that purpose, we create, for each edge j, $j=1,2,3$, a warp (deformation) function $\boldsymbol{w}^j$ that uses the one-dimensional warp function

$$w(x) = \sum_{i=1}^{N_p}(x_i^{LGL}-x_i^e)\ell_i^e(x), \quad r\in[-1,1], \tag{3.13}$$

where $x_i^e = -1+2i/N$, $i=0,\ldots,N$, are the equidistant points on $[-1,1]$, x_i^{LGL} are the LGL points and $\ell_i^e(x)$ are the Lagrange polynomials based on x_i^e. Thus, $w(x)$ measures the difference between the equidistant points and the LGL points. We can extend the edge warp into the triangle by considering blending function b^j, $j=1,2,3$, (see [29]). In Figure 3.2, we present a schematic representation of the deformation of the sum of the *Warp & Blend* functions for the three edges, where the arrows show how the nodes are moved from the equidistant nodal set.

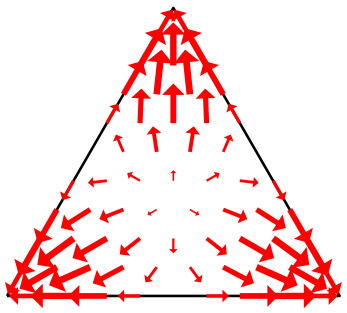


Fig. 3.2 Deformation of the equidistant nodes, considering the sum of the *Warp & Blend* functions for the three edges. Adapted from [29].

Considering a generalized warping function, the sum of the *Warp & Blend* functions for the three edges is given by

$$g(\lambda^1,\lambda^2,\lambda^3) = \sum_{j=1}^{3}(1+(\alpha\lambda^j)^2)b^j\boldsymbol{w}^j, \tag{3.14}$$

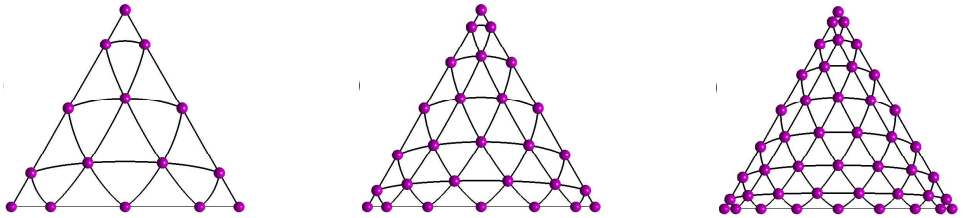
where $(\lambda^1,\lambda^2,\lambda^3)$ are the barycentric coordinates. Thus, (3.14) form a set of α–optimised nodes suitable for interpolation. A measure of the quality of the interpolation is the Lebesgue constant defined as

$$\Lambda = \max_{\boldsymbol{x}\in T^k} \sum_{i=1}^{N_p} |\ell_i^k(\boldsymbol{x})|,$$

since

$$\begin{aligned} ||u-u_h||_\infty = ||u-u^*+u^*-u_h||_\infty \leq ||u-u^*||_\infty + ||u^*-u_h||_\infty = ||u-u^*||_\infty + ||u^* - \sum_{i=1}^{N_p} u(\boldsymbol{x}_i^k)\ell(\boldsymbol{x})||_\infty \\ \leq ||u-u^*||_\infty + \max_{\boldsymbol{x}\in T^k} \sum_{i=1}^{N_p} |\ell_i^k(\boldsymbol{x})| \times ||u-u^*||_\infty = (1+\Lambda)||u-u^*||_\infty, \end{aligned}$$

where u^* represents the best approximating polynomial of order N. Thus, it is natural to choose α that minimise the Lebesgue constant. In Figure 3.3 we illustrate some examples of the resulting nodal sets, considering polynomials of order $N = 4,6,8$. Note that the black lines highlight the effect of the *Warp & Blend* deformation on the lines that connect the undeformed equidistant nodes. Furthermore, note that the nodes are computed in the equilateral triangle and the orthonormal basis is defined in the standard triangle Δ. Thus, we need to map the nodes into Δ using a linear mapping $\boldsymbol{x} = (x,y) \longrightarrow \boldsymbol{r} = (r,s)$.



(a) $N = 4$. (b) $N = 6$. (c) $N = 8$.

Fig. 3.3 Node distribution for $N = 4,6,8$ on the equilateral triangle. Adapted from [29].

3.2.3 Element-wise operations

We can now define the local representation of the matrices M^k, S_x^k and S_y^k in the form

$$\begin{aligned} M_{ij}^k &= \int_{T^k} \ell_i^k(\boldsymbol{x})\ell_j^k(\boldsymbol{x})\, d\boldsymbol{x} = J^k \int_\Delta \ell_i(\boldsymbol{r})\ell_j(\boldsymbol{r})\, d\boldsymbol{r} = J^k M_{ij}, \\ S_{x,ij}^k &= \int_{T^k} \ell_i^k(\boldsymbol{x}) \frac{\partial \ell_j^k}{\partial x}(\boldsymbol{x})\, d\boldsymbol{x} = J^k \int_\Delta \ell_i(\boldsymbol{r}) \left(\frac{\partial \ell_j}{\partial r}(\boldsymbol{r}) r_x + \frac{\partial \ell_j}{\partial s}(\boldsymbol{r}) s_x \right) d\boldsymbol{r} \\ &= r_x J^k \int_\Delta \ell_i(\boldsymbol{r}) \frac{\partial \ell_j}{\partial r}(\boldsymbol{r})\, d\boldsymbol{r} + s_x J^k \int_\Delta \ell_i(\boldsymbol{r}) \frac{\partial \ell_j}{\partial s}(\boldsymbol{r})\, d\boldsymbol{r} \\ S_{y,ij}^k &= \int_{T^k} \ell_i^k(\boldsymbol{x}) \frac{\partial \ell_j^k}{\partial y}(\boldsymbol{x})\, d\boldsymbol{x} = J^k \int_\Delta \ell_i(\boldsymbol{r}) \left(\frac{\partial \ell_j}{\partial r}(\boldsymbol{r}) r_y + \frac{\partial \ell_j}{\partial s}(\boldsymbol{r}) s_y \right) d\boldsymbol{r} \\ &= r_y J^k \int_\Delta \ell_i(\boldsymbol{r}) \frac{\partial \ell_j}{\partial r}(\boldsymbol{r})\, d\boldsymbol{r} + s_y J^k \int_\Delta \ell_i(\boldsymbol{r}) \frac{\partial \ell_j}{\partial s}(\boldsymbol{r})\, d\boldsymbol{r}. \end{aligned}$$

Considering the generalized Vandermonde matrix we get

$$l_i(\boldsymbol{r}) = \sum_{n=1}^{N_p} \left(V^T\right)^{-1}_{in} \psi_n(\boldsymbol{r}).$$

Thus, recalling the orthonormal basis (3.12),

$$\begin{aligned} M_{ij} &= \int_\Delta \sum_{n=1}^{N_p} \left(V^T\right)^{-1}_{in} \psi_n(\boldsymbol{r}) \sum_{m=1}^{N_p} \left(V^T\right)^{-1}_{jm} \psi_m(\boldsymbol{r})\, d\boldsymbol{r} \\ &= \sum_{n=1}^{N_p}\sum_{m=1}^{N_p} \left(V^T\right)^{-1}_{in} \left(V^T\right)^{-1}_{jm} (\psi_n(\boldsymbol{r}), \psi_m(\boldsymbol{r}))_{L^2(\Delta)} \\ &= \sum_{n=1}^{N_p} \left(V^T\right)^{-1}_{in} \left(V^T\right)^{-1}_{jn}, \end{aligned}$$

which implies that $M = \left(VV^T\right)^{-1}$. We conclude that $M^k = J^k \left(VV^T\right)^{-1}$.

To determine the matrices S_x^k and S_y^k, we introduce the differentiation matrices D_r and D_s, defined by

$$D_{r,ij} = \left.\frac{\partial \ell_j}{\partial r}\right|_{\boldsymbol{r}_i}, \quad D_{s,ij} = \left.\frac{\partial \ell_j}{\partial s}\right|_{\boldsymbol{r}_i}, \tag{3.15}$$

and we denote $D_x = r_x D_r + s_x D_s$ e $D_y = r_y D_r + s_y D_s$. Moreover,we express the derivative with respect to r of the j–th Lagrange polynomial as

$$\frac{\partial \ell_j}{\partial r}(\boldsymbol{r}) = \sum_{n=1}^{N_p} \left.\frac{\partial \ell_j}{\partial r}\right|_{\boldsymbol{r}_n} \ell_n(\boldsymbol{r}),$$

and similarly for derivative with respect to s of the j–the Lagrange polynomial.

To define the matrices (3.15), it is require

$$V_{r,ij} = \left.\frac{\partial \psi_j}{\partial r}\right|_{\boldsymbol{r}_i}, \quad V_{s,ij} = \left.\frac{\partial \psi_j}{\partial s}\right|_{\boldsymbol{r}_i}. \tag{3.16}$$

The coefficients of these matrices can be easily evaluated. In fact, from the relationship (3.11), we obtain that $V^T D_r^T = (V_r)^T$, which implies $D_r V = V_r$. Analogously, $D_s V = V_s$. If we recall the orthonormal basis (3.12)

$$\frac{\partial \psi_j}{\partial r} = \frac{\partial a}{\partial r}\frac{\partial \psi_j}{\partial a}, \quad \frac{\partial \psi_j}{\partial s} = \frac{\partial a}{\partial s}\frac{\partial \psi_j}{\partial a} + \frac{\partial \psi_j}{\partial b}, \quad \text{where} \quad \frac{\partial a}{\partial r} = \frac{2}{1-s}, \quad \frac{\partial a}{\partial s} = \frac{-2(1+r)}{(1-s)^2}.$$

Taking into account the matrices just defined, the matrices S_x^k e S_y^k can now be computed. Note that

$$\begin{aligned}
\left(M^k D_x\right)_{ij} &= J^k \sum_{n=1}^{N_p} M_{in} D_{x,nj} = J^k \sum_{n=1}^{N_p} M_{in} \left(r_x D_{r,nj} + s_x D_{s,nj}\right) \\
&= J^k \left(r_x \sum_{n=1}^{N_p} M_{in} D_{r,nj} + s_x \sum_{n=1}^{N_p} M_{in} D_{s,nj} \right) \\
&= J^k \left(r_x \sum_{n=1}^{N_p} \int_\Delta \ell_i(\boldsymbol{r}) \ell_n(\boldsymbol{r}) \frac{\partial \ell_j}{\partial r}\bigg|_{\boldsymbol{r}_n} d\boldsymbol{r} + s_x \sum_{n=1}^{N_p} \int_\Delta \ell_i(\boldsymbol{r}) \ell_n(\boldsymbol{r}) \frac{\partial \ell_j}{\partial s}\bigg|_{\boldsymbol{r}_n} d\boldsymbol{r} \right) \\
&= J^k \left(r_x \int_\Delta \ell_i(\boldsymbol{r}) \sum_{n=1}^{N_p} \frac{\partial \ell_j}{\partial r}\bigg|_{\boldsymbol{r}_n} \ell_n(\boldsymbol{r}) \; d\boldsymbol{r} + s_x \int_\Delta \ell_i(\boldsymbol{r}) \sum_{n=1}^{N_p} \frac{\partial \ell_j}{\partial s}\bigg|_{\boldsymbol{r}_n} \ell_n(\boldsymbol{r}) \; d\boldsymbol{r} \right) \\
&= J^k \left(r_x \int_\Delta \ell_i(\boldsymbol{r}) \frac{\partial \ell_j}{\partial r}(\boldsymbol{r}) \; d\boldsymbol{r} + s_x \int_\Delta \ell_i(\boldsymbol{r}) \frac{\partial \ell_j}{\partial s}(\boldsymbol{r}) \; d\boldsymbol{r} \right) = J^k \left(r_x S_{r,ij} + s_x S_{s,ij}\right) = \left(S_x^k\right)_{ij}.
\end{aligned}$$

Thus, we obtain

$$\begin{aligned}
S_x^k &= M^k D_x = J^k M (r_x D_r + s_x D_s) = r_x J^k S_r + s_x J^k S_s, \\
S_y^k &= M^k D_y = J^k M (r_y D_r + s_y D_s) = r_y J^k S_r + s_y J^k S_s.
\end{aligned}$$

3.3 Numerical results

We intend to solve the Helmholtz problem using the DG method. In order to validate the implementation of the method and evaluate the error and the order of convergence, we consider a numerical example for which the exact solution is known. Taking into account the problem that motivated our study, described in the introductory chapter, we will consider the Helmholtz problem in a curved domain, which aims to simulate the human cornea. In this chapter, we will consider the domain defined by a unit circle centred at the origin.

Considering $\nu = 1$ and $\Omega = \{(x,y) \in \mathbb{R}^2 : x^2 + y^2 < 1\}$, we aim at solving the following problem

$$\begin{aligned}
-\nabla^2 u(x,y) - u(x,y) &= f(x,y), \quad (x,y) \in \Omega \\
u(x,y) &= 0, \quad (x,y) \in \partial\Omega,
\end{aligned} \tag{3.17}$$

where $f(x,y) = 4x\left((x^2+y^2)\sin(1-x^2-y^2) + 2\cos(1-x^2-y^2)\right) - x\sin(1-x^2-y^2)$, and the exact solution is given by $u(x,y) = x\sin(1-x^2-y^2)$ (see Figure 3.4).

If we denote by u_h an approximate solution determined with the numerical method, we say that the method has order of convergence p, for a given norm $\|\cdot\|$, if

$$||u - u_h|| \leq Ch^p,$$

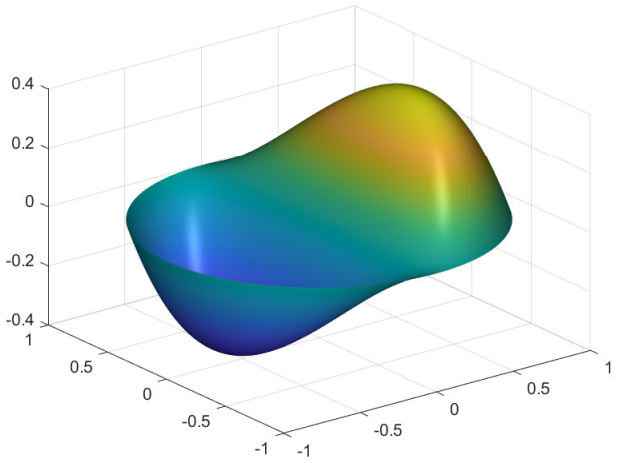


Fig. 3.4 Exact solution of (3.17).

with C a real constant independent of h. In this work, we consider the maximum norms, evaluating the error for the grid points on the mesh, $\boldsymbol{x}^k \in T^k, k = 1, \ldots, K$,

$$E_\infty = ||u(\boldsymbol{x}^k) - u_h(\boldsymbol{x}^k)||_\infty = \max_{\boldsymbol{x}^k \in T^k \in \mathscr{T}_h} |u(\boldsymbol{x}^k) - u_h(\boldsymbol{x}^k)|.$$

In order to estimate the order of convergence of the method, we considered different spatial polygonal meshes generated by Gmsh (version 4.6.0) [12], with different mesh parameters h. Considering two distinct values of h, say h_1 and h_2 and the corresponding numerical solutions u_{h_1} and u_{h_2}, we compute the maximum norms $E_{\infty,1}$ and $E_{\infty,2}$, respectively. Assuming $E_{\infty,1}/E_{\infty,2} = (h_1/h_2)^p$, we have that the order of convergence can be estimated by

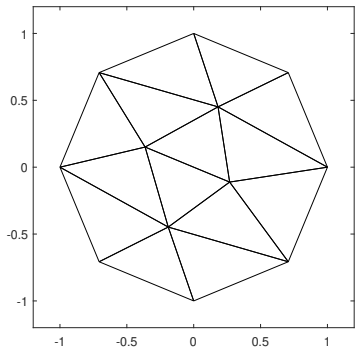
$$p = \frac{\log\left(E_{\infty,1}/E_{\infty,2}\right)}{\log\left(h_1/h_2\right)}$$

In Table 3.1, we report the errors and the convergence rates for the classical DG method for polynomials of degree N, with $N = 1, 2, 3, 4$. The numerical solutions obtained with the DG method considering different meshes are represented in Figure 3.5.

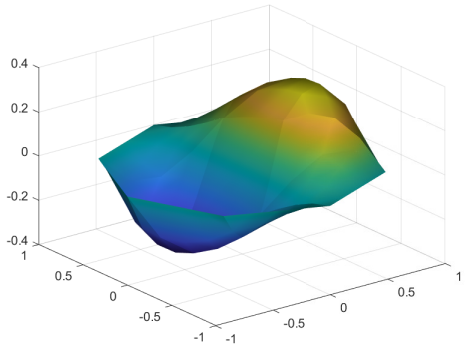
Table 3.1 Errors and convergence rates for the classical DG formulation.

K	h	$N=1$		$N=2$		$N=3$		$N=4$	
		E_∞	p	E_∞	p	E_∞	p	E_∞	p
14	9.34e-01	8.37e-02	-	1.24e-01	-	1.07e-01	-	1.25e-01	-
64	4.70e-01	3.17e-02	1.4	3.66e-02	1.8	2.98e-02	1.9	3.66e-02	1.8
262	2.34e-01	6.77e-03	2.2	7.53e-03	2.3	6.05e-03	2.3	7.54e-03	2.3
1096	1.13e-01	1.45e-03	2.1	1.70e-03	2.0	1.36e-03	2.0	1.70e-03	2.0
4316	5.69e-02	4.57e-04	1.7	4.27e-04	2.0	3.42e-04	2.0	4.27e-04	2.0

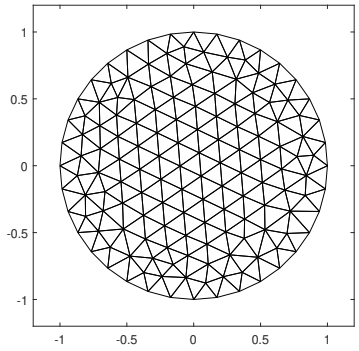
The results obtained in our simulations by the DG method show that the order of convergence for the DG method is $p = 2$. We notice that by increasing the order of the polynomial, N, the order



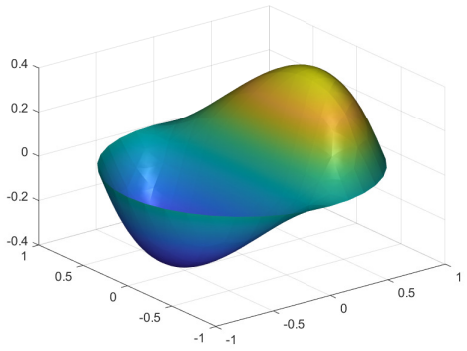
(a) Mesh $\mathscr{T}_h$ with $h =$ 9.34e-01



(b) DG solution



(c) Mesh $\mathscr{T}_h$ with $h =$ 2.34e-01



(d) DG solution

Fig. 3.5 Numerical results obtained for polynomials of degree $N = 4$, with different mesh parameters.

convergence does not increase. In other words, the method reaches a second-order convergence, independently of the degree of the polynomial. This reduction in the accuracy occurs because we are dealing with a curved boundary domain and we are solving the Helmholtz's equation for polygonal meshes that do not exactly fit the physical domain. This highlights the importance of the boundary condition treatment, especially for high-order methods because the errors in the boundary may pollute the solution inside the domain, rendering the use of a high-order scheme useless [4]. In particular, the DG solutions are highly sensitive to the accuracy of approximations of the curved boundaries [3, 4] and it has been shown that given homogeneous Dirichlet boundary conditions on a physical domain Ω if these conditions are imposed on the polygonal domain Ω_h, any finite element method will be at most second-order accurate [28]. To overcome this problem, in the following chapters, we propose two different strategies to deal with curved domains, which arise naturally in our domain of interest for the application, that consider polygonal meshes.

Chapter 4

Curved boundary treatment

The treatment of boundary value problems in curved boundary domains has been a subject of growing interest in the numerical analysis community. The question that arises concerns the reduction of the order of convergence of numerical methods when considering the approximation of the domain by a polygonal mesh. As we saw in the previous chapter, the DG method turns out to be a second-order accurate method when boundary conditions are not exactly located at the nodes of the mesh and the edge of elements [3, 4].

The aim of this chapter is to present an efficient method for solving the Helmholtz problem in a curved domain. In Section 4.1, we give an overview of the numerical methods that have been presented to overcome the problems that occur when considering domains with curved boundary. In order to avoid the generation of curved meshes, some alternatives considering polygonal meshes have been proposed. Following the work developed in [2] for finite volumes, in this thesis we focus on the polynomial reconstruction method. The main idea of this method is to design a polynomial reconstruction of the boundary condition of the polygonal computational domain that takes into account the boundary condition in the physical domain. We describe this approach in detail in Section 4.2. In Section 4.3, we present the numerical tests to assess the accuracy, converge rate and efficiency of this approach to solve the Helmholtz equation with Dirichlet boundary conditions in a curved boundary domain, with the DG method.

4.1 Treatment of curved boundary domains

A way to deal with curved boundaries is to consider the so-called isoparametric elements, introduced by Bassi and Rebay in the context of DG methods [4]. The elements are called isoparametric since the same functions are used to express the transformation from the reference element to the real element and the solution in the reference element. This approach requires the use of non-linear transformations of the reference triangle, which requires high computational effort. In addition, it requires the generation of curved meshes, which turns out to be impractical for complex geometries.

Some alternative methods have been proposed in an attempt to overcome these difficulties. In [17], the authors present a curvature boundary condition approach, in replacement of reflecting boundary conditions, for steady two-dimensional Euler equations. This technique uses a polygonal computational domain Ω_h and, assuming that the only available information is the mesh itself, the

physical boundary is approximated by an arc of the circle passing through the vertices v_i of the elements T^k, $k = 1,\ldots,K$, such that $v_i \in \partial\Omega_h$, $i = 1,2$. The unit normal to the circle passing through each interpolation point on the computational boundary is computed and it replaces the unit normal to the computational boundary. Despite the improvement in the quality of the solution, compared to the solutions obtained with reflecting boundary conditions, the applicability of the approach seems limited to other boundary conditions.

Another method proposed in the literature is the so-called Extensions from Subdomains [6]. The main idea of this approach consists in determining a new Dirichlet boundary condition on the polygonal computational boundary from the one evaluated on the physical boundary. This method does not require local mapping or generation of curved meshes, which simplifies the numerical schemes. However, in order to determine the new Dirichlet conditions is required to define a path between the computational boundary and the physical one. Moreover, the method is only available for second-order operators.

In [32], the author proposes a modified DG scheme defined on polygonal meshes. The method avoids integrals inside curved elements. However, integrations along boundary curve segments are still necessary. Recently, this approach was extended to solving three-dimensional Euler equations and it was simplified by considering the relation between the normal vector of the computational domain and surface Jacobian [30]. In this case, not only the integrals over any curved element are avoided, but also the integrations along boundary curve segments are not required.

In the context of the finite volume method, a method called ROD (Reconstruction for Off-site Data) was presented [8]. As the name of the method suggests, a polynomial reconstruction is developed which takes into account the real boundary conditions (which are not in the polygonal computational domain). This approach does not require the generation of curved meshes to adjust the boundary, nor complex nonlinear transformations, which contributes to computational efficiency and simplifies the numerical schemes. We now intend to generalize this approach to the DG method.

4.2 Polynomial reconstruction formulation

Let $\Omega \subset \mathbb{R}^2$ be a bounded open set of smooth boundary $\partial\Omega$ and $\boldsymbol{x} = (x,y)$. We aim at solving the problem

$$\begin{aligned} -\nabla^2 u(\boldsymbol{x}) - \nu^2 u(\boldsymbol{x}) &= f(\boldsymbol{x}), \quad \boldsymbol{x} \in \Omega \\ \mathscr{B}(u,\boldsymbol{x}) = \alpha(\boldsymbol{x})u(\boldsymbol{x}) - g(\boldsymbol{x}) &= 0, \quad \boldsymbol{x} \in \partial\Omega, \end{aligned}$$

where the wave number ν is sufficiently small. Assume that the physical domain Ω can be approximated by the polygonal computational domain Ω_h defined as (3.2), with boundary $\partial\Omega_h$.

As we have seen in the previous chapter, the numerical solution of the DG method may be written as (3.3) and the local solution u_h^k, in each element T^k, can be expressed as a polynomial of degree N given by (3.4), i.e.,

$$\boldsymbol{x} \in T^k \in \mathscr{T}_h : \quad u_h^k(\boldsymbol{x}) = \sum_{i=1}^{N_p} u_h^k(\boldsymbol{x}_i^k)\ell_i^k(\boldsymbol{x}),$$

where $\ell_i^k(\boldsymbol{x})$ is the multidimensional Lagrange polynomial defined by the grid points $\boldsymbol{x}_i^k$, $i = 1,...,N_p$, on T^k, and $N_p = (N+1)(N+2)/2$.

Let us consider a a set of point $P_r \in \partial\Omega$, $r = 1,...,R$, on the physical boundary $\partial\Omega$, , such that $P_{r+R} = P_r$, $r \in \mathbb{Z}$. This set of points defines what we will call the collar $\mathscr{C}_h$. Moreover, assume that, for each point $P_r \in \mathscr{C}_h$, the boundary condition $\mathscr{B}(.,P_r)$ is known.

For each element T^k with a common edge e^k with the computational boundary $\partial\Omega_h$, we aim at determining a polynomial

$$\pi^k(\boldsymbol{x};\boldsymbol{a}) = \sum_{i=1}^{N_p} a_i \ell_i^k(\boldsymbol{x}), \tag{4.1}$$

where $\boldsymbol{a} = [a_1,\cdots,a_{N_p}]^T$, such that it is the closest polynomial from the local solution u_h^k that satisfies the boundary condition at a set of points on the boundary. We denote that polynomial by π^{*k}. Note that the polynomial is uniquely determined by the vector of coefficients $\boldsymbol{a}^*$.

An approach to determine the polynomial π^{*k} consists in minimising the norm $L^2(T^k)$ of the distance between the polynomials π^k and u_h^k, subject to the same constraint described above. Thus, for each element T^k with a common edge e^k with the computational boundary $\partial\Omega_h$, we intend to determine the polynomial (4.1) such that

$$\begin{aligned}\pi^{*k}(\boldsymbol{x};\boldsymbol{a}^*) = \underset{\boldsymbol{a}\in\mathbb{R}^{N_p}}{\arg\min}\ & \frac{1}{2}\int_{T^k} (\pi^k(\boldsymbol{x}_i^k;\boldsymbol{a}) - u_h^k(\boldsymbol{x}_i^k))^2\ d\boldsymbol{x} \\ \text{s.a}\quad & \mathscr{B}(\pi^k(\cdot;\boldsymbol{a}),\boldsymbol{P}^k) = 0,\end{aligned}$$

where $\boldsymbol{P}^k = [P_1^k,\ldots,P_{R^k}^k]^T$ are the R^k points of the collar $\mathscr{C}_h$ for the element T^k (see Figure 4.1) and $\boldsymbol{a}^* = [a_1^*,\cdots,a_{N_p}^*]^T$ is the vector of coefficients of π^{*k}.

Another approach could be to minimise the norm of the difference between the coefficients of the polynomials π^k and u_h^k, such that π^k satisfies the boundary condition at a set of R^k points on the boundary $\partial\Omega$

$$\pi^{*k}(\boldsymbol{x};\boldsymbol{a}*) = \underset{\boldsymbol{a}\in\mathbb{R}^{N_p}}{\arg\min}\ \frac{1}{2}\sum_{i=1}^{N_p}(a_i - u_h^k(\boldsymbol{x}_i^k))^2 \tag{4.2}$$

$$\text{s.a}\quad \mathscr{B}(\pi^k(\cdot;\boldsymbol{a}),\boldsymbol{P}^k) = 0, \tag{4.3}$$

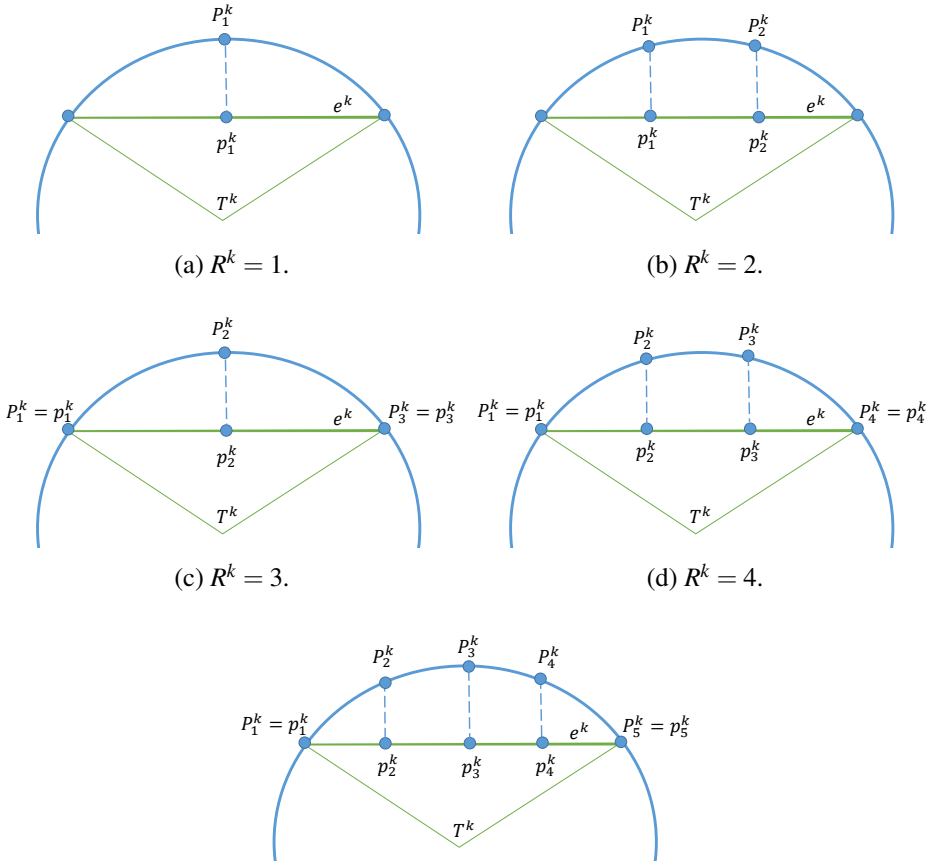
where $\boldsymbol{P}^k = [P_1^k,\ldots,P_{R^k}^k]^T$. In this thesis, we consider this second approach.

In order to solve the constrained minimisation problem and determine the coefficients a_i*, $i = 1,\ldots,N_p$, of π^{*k} we introduce the Lagrangian function given by

$$\mathfrak{L}(\boldsymbol{a},\boldsymbol{\lambda}) = \frac{1}{2}\sum_{i=1}^{N_p}(a_i - u_h^k(\boldsymbol{x}_i^k))^2 + \sum_{l=1}^{R^k}\lambda_l \mathscr{B}(\pi^k(\cdot\,;\boldsymbol{a}),P_l),$$

where $\boldsymbol{a} = [a_1,\cdots,a_{N_p}]^T$ is the vector of coefficients in (4.1) and $\boldsymbol{\lambda} = [\lambda_1,\ldots,\lambda_{R^k}]^T$ is the set of Lagrange multipliers, $\lambda_l \in \mathbb{R}$, $l = 1,\ldots,R^k$. Thus, the problem of constrained minimisation corresponds to find $\boldsymbol{a}$ and $\boldsymbol{\lambda}$ such that

$$\nabla_{\boldsymbol{a}}\mathfrak{L}(\boldsymbol{a},\boldsymbol{\lambda}) = 0, \quad \nabla_{\boldsymbol{\lambda}}\mathfrak{L}(\boldsymbol{a},\boldsymbol{\lambda}) = 0,$$



(a) $R^k = 1$. (b) $R^k = 2$.

(c) $R^k = 3$. (d) $R^k = 4$.

(e) $R^k = 5$.

Fig. 4.1 Example of an element T^k with a common edge e^k with the computational boundary $\partial\Omega_h$ and R^k points of the collar $\mathscr{C}_h$ for that element, where $R^k = 1,2,3,4,5$.

where

$$\nabla_{a_j}\mathfrak{L}(\boldsymbol{a},\boldsymbol{\lambda}) = a_j - u_h^k(\boldsymbol{x}_j^k) + \sum_{l=1}^{R^k}\lambda_l\nabla_{a_j}\mathscr{B}(\pi^k(\cdot\,;\boldsymbol{a}),P_l), \quad j = 1,...,N_p,$$
$$\nabla_{\lambda_l}\mathfrak{L}(\boldsymbol{a},\boldsymbol{\lambda}) = \mathscr{B}(\pi^k(\cdot\,;\boldsymbol{a}),P_l), \quad l = 1,...,R^k.$$

Taking into account the expression of the boundary condition, we may write

$$\nabla_{a_j}\mathscr{B}(\pi^k(\cdot\,;\boldsymbol{a}),P_l) = \nabla_{a_j}\left(\alpha(P_l)\sum_{i=1}^{N_p}a_i\ell_i^k(P_l) - g(P_l)\right) = \alpha(P_l)\ell_j^k(P_l), \qquad j = 1,...,N_p,$$
$$\nabla_{\lambda_l}\mathfrak{L} = \mathscr{B}(\pi^k(\cdot\,;\boldsymbol{a}),P_l) = \alpha(P_l)\sum_{i=1}^{N_p}a_i\ell_i^k(P_l) - g(P_l), \qquad l = 1,...,R^k.$$

Then, if we consider the column vector

$$B_l = [\alpha(P_l)\ell_j^k(P_l)]_{j=1}^{N_p}, \quad l = 1,\dots R^k,$$

and the matrix $B = [B_1 \dots B_{R^k}]$, we may rewrite the minimisation problem under the matrix form

$$\begin{bmatrix} I & B \\ B^\intercal & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{a} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \boldsymbol{u}_h^k \\ g(\boldsymbol{P}) \end{bmatrix}, \tag{4.4}$$

where $\mathbf{0}$ is the null matrix $R^k \times R^k$ and $\boldsymbol{u}_h^k$ is the vector of coefficients obtain by the DG method, $\boldsymbol{u}_h^k = [u_h^k(\boldsymbol{x}_j^k)]_{j=1}^{N_p}$.

The solution $\boldsymbol{a}^*$ provides the expected polynomial (4.1). Note that the Hessian of the Lagrangian function with respect to $\boldsymbol{a}$ is the identity matrix I, which is positive definite. Therefore, $\boldsymbol{a}^*$ is a strict local minimiser. Moreover, since $\mathfrak{L}$ is strictly convex with respect to $\boldsymbol{a}$, the local minimiser is unique and, consequently, it also is the unique global minimiser of $\mathfrak{L}$.

We call the DG method combined with polynomial reconstruction by the DG-ROD method. The DG-ROD method starts with an iteration of the DG method, obtaining, for each element $T^k \in \mathscr{T}_h$, the polynomial u_h^k. After this first iteration, for each T^k with a common edge e^k with the boundary $\partial\Omega_h$, we determine a polynomial π^k that satisfies the boundary condition at a set of R^k points on the boundary. Then we update the DG solution by imposing that on this edge e^k the boundary condition is given by the value of π^k. The procedure is repeated while the norm of the difference between two successive iterations (which were obtained with the polynomial reconstruction) is greater than a certain tolerance tol (we can also define the maximum number of iterations). The procedure is performed at least twice in order to compare the solutions obtained with the polynomial reconstruction. This process is described in Algorithm 2.

Algorithm 2 DG-ROD method

1. $U^{(0)} = \mathbf{DG}(g)$
2. Set $it = 0$
3. Set $flag = 1$
4. while $flag = 1$
5. it = it + 1
6. Evaluate $g(P_l)$ and B_l, $l = 1, \dots, R^k$
7. Solve (4.4) in order to obtain $\boldsymbol{a}^*$
8. Update the boundary condition by (4.1)
9. $U^{(it)} = \mathbf{DG}(\pi^{*k})$
10. if $it = 1$, then $flag = 1$
11. elseif $||U^{(it)} - U^{(it-1)}||_\infty > tol$, then $flag = 1$
12. else $flag = 0$
13. end
14. end

4.3 Numerical results

In this section, we present some numerical results performed with the DG-ROD method. We consider the Helmholtz problem in a curved two-dimensional domain, which will be approximated by polygonal meshes. The error and convergence order analysis will be done by numerical experiments for the problem (3.17) presented in the previous chapter. Considering the same conditions and polygonal meshes presented in Section 3.3, in Table 4.1 we report the errors and convergence rates for the DG-ROD method with $tol = 10^{-8}$, a maximum number of iterations equals to 100 and considering a set of $R^k = 1,2,3,4,5$ points on the boundary $\partial\Omega$ for each element T^k with common edge e^k with the computational boundary $\partial\Omega_h$. In Figure 4.2, we plot the error depending on the mesh parameter h for polynomials of degree $N = 1,2,3,4$ considering $R^k = 5$.

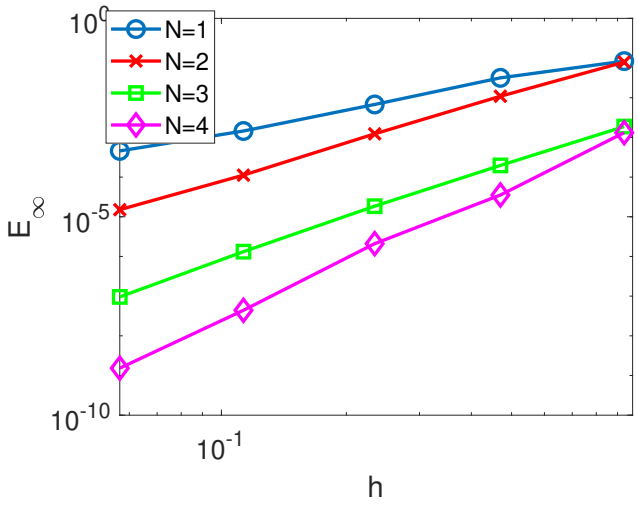


Fig. 4.2 Global error E_∞ vs mesh parameter h, considering $R^k = 5$ points.

The results obtained in our simulations by the iterative DG-ROD method for polynomials of degree N, with $N = 1,2,3,4$, show that this method allows to obtain a smaller error and higher order of convergence compared to the classical DG formulation. Thus, the DG-ROD method, unlike the classical DG method, allows to achieve high order in domains for curved boundary domains.

Moreover, the numerical results suggest that there is a relation between the number of points R^k for each element T^k with a common edge with the computational boundary $\partial\Omega_h$ and the degree N of the polynomial. In other words, to increase the convergence rate to $N+1$ for polynomials of degree N, we need to consider $N+1$ points on the boundary for each element T^k with a common edge with the computational boundary $\partial\Omega_h$. Results for nonconstant Dirichlet boundary conditions were also obtained and suggest this same relation.

In order to analyze the decrease of the error with the number of iterations, in Figure 4.3 we present the global error and the error between two successive iterations as the number of iterations increases. These results were obtained considering 100 iterations of the DG-ROD method for polynomials of degree $N = 4$ and a polygonal mesh $\mathscr{T}_h$ with $h =$5.69e-02.

We note that the most significant decrease in error occurs in the first iterations. Moreover, for $N = 4$ we note a slightly change in the global error $||u - u_h||_\infty$ considering $R^k = 3$ in relation to

Table 4.1 Errors and convergence rates for the DG-ROD method.

$R^k = 1$

K	h	$N=1$		$N=2$		$N=3$		$N=4$	
		E_∞	p	E_∞	p	E_∞	p	E_∞	p
14	9.34e-01	7.34e-02	-	2.15e-02	-	3.03e-02	-	7.23e-02	-
64	4.70e-01	2.56e-02	1.5	2.02e-03	3.4	5.02e-03	2.6	1.73e-02	2.1
262	2.34e-01	9.60e-03	1.4	1.96e-04	3.4	1.06e-03	2.2	3.68e-03	2.2
1096	1.13e-01	2.69e-03	1.8	1.81e-05	3.3	2.46e-04	2.0	8.75e-04	2.0
4316	5.69e-02	7.47e-04	1.9	1.53e-06	3.6	6.25e-05	2.0	2.25e-04	2.0

$R^k = 2$

K	h	$N=1$		$N=2$		$N=3$		$N=4$	
		E_∞	p	E_∞	p	E_∞	p	E_∞	p
14	9.34e-01	8.16e-02	-	1.19e-01	-	8.69e-03	-	3.10e-02	-
64	4.70e-01	2.14e-02	2.0	2.50e-02	2.3	2.50e-03	1.8	7.25e-03	2.1
262	2.34e-01	9.16e-03	1.2	2.20e-03	3.5	5.38e-04	2.2	1.53e-03	2.2
1096	1.13e-01	2.54e-03	1.8	1.95e-04	3.3	1.25e-04	2.0	3.54e-04	2.0
4316	5.69e-02	7.04e-04	1.9	2.80e-05	2.8	3.19e-05	2.0	8.97e-05	2.0

$R^k = 3$

K	h	$N=1$		$N=2$		$N=3$		$N=4$	
		E_∞	p	E_∞	p	E_∞	p	E_∞	p
14	9.34e-01	8.37e-02	-	2.15e-02	-	2.77e-02	-	7.13e-02	-
64	4.70e-01	3.17e-02	1.4	1.77e-03	3.6	2.51e-03	3.5	1.70e-02	2.1
262	2.34e-01	6.77e-03	2.2	1.71e-04	3.4	2.36e-04	3.4	3.69e-03	2.2
1096	1.13e-01	1.45e-03	2.1	1.66e-05	3.2	2.40e-05	3.1	8.77e-04	2.0
4316	5.69e-02	4.57e-04	1.7	1.38e-06	3.6	3.87e-06	2.7	2.26e-04	2.0

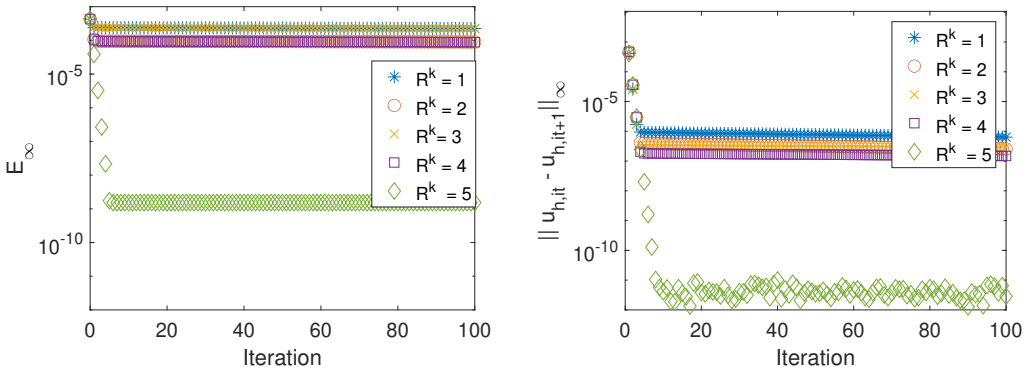
$R^k = 4$

K	h	$N=1$		$N=2$		$N=3$		$N=4$	
		E_∞	p	E_∞	p	E_∞	p	E_∞	p
14	9.34e-01	8.37e-02	-	2.15e-02	-	1.26e-03	-	2.92e-02	-
64	4.70e-01	3.17e-02	1.4	1.68e-03	3.7	1.50e-04	3.1	6.88e-03	2.1
262	2.34e-01	6.77e-03	2.2	1.70e-04	3.3	1.98e-05	2.9	1.49e-03	2.2
1096	1.13e-01	1.45e-03	2.1	1.64e-05	3.2	1.34e-06	3.7	3.46e-04	2.0
4316	5.69e-02	4.57e-04	1.7	1.37e-06	3.6	9.67e-08	3.8	8.82e-05	2.0

$R^k = 5$

K	h	$N=1$		$N=2$		$N=3$		$N=4$	
		E_∞	p	E_∞	p	E_∞	p	E_∞	p
14	9.34e-01	8.37e-02	-	7.90e-02	-	1.88e-03	-	1.32e-03	-
64	4.70e-01	3.17e-02	1.4	1.09e-02	2.9	1.97e-04	3.3	3.54e-05	5.3
262	2.34e-01	6.77e-03	2.2	1.23e-03	3.1	1.85e-05	3.4	2.10e-06	4.1
1096	1.13e-01	1.45e-03	2.1	1.12e-04	3.3	1.32e-06	3.6	4.40e-08	5.3
4316	5.69e-02	4.57e-04	1.7	1.51e-05	2.9	9.57e-08	3.8	1.53e-09	4.9

the global error when considering $R^k = 1$ point. Compared to $R^k = 1$, for $R^k = 3$ we only add the information of the vertices of the edges on the computational boundary, $\partial\Omega_h$ (see Figure 4.1). If we



(a) Error $||u-u_h||_\infty$.

(b) Errors between two successive iterations.

Fig. 4.3 Errors obtained with the DG-ROD method for polynomials of degree $N=4$, considering a polygonal mesh $\mathscr{T}_h$, with $h=$5.69e-02.

considered 3 points $P_r \in \partial\Omega \setminus \partial\Omega_h$, $r=1,2,3$, the error would decrease but we still would not obtain a high order of accuracy for $N=4$ (see Table 4.2). Note the same relation for $R^k=2$ and $R^k=4$.

Although the error has not decreased significantly from $R^k=1$ to $R^k=3$ and from $R^k=2$ to $R^k=4$, for polynomial of degree $N=4$, we highlight the importance of considering the vertices of the edges on the computational boundary. If we consider only 3 points $P_r \in \partial\Omega \setminus \partial\Omega_h$, $r=1,2,3$, the order of accuracy does not improve for polynomials of degree $N=4$. However, if we add the 2 vertices of the edges on the computational boundary (which are known points), the method achieves a high order of accuracy.

For polynomials of degree $N=4$, the method obtains a high order when considering 5 points P_r such that $P_r \in \partial\Omega \setminus \partial\Omega_h$, $r=1,\ldots,5$ (see Table 4.2). A high order of accuracy can also be obtained considering 3 points P_r such that $P_r \in \partial\Omega \setminus \partial\Omega_h$, $r=1,2,3$, and the 2 vertices of the edges on the computational boundary $\partial\Omega_h$ (see Table 4.1). This alternative, which was chosen in this thesis, has the advantage that we only need to determine 3 points on the boundary $\partial\Omega$, because the other 2 points are already known.

Table 4.2 Errors and convergence rates for the DG-ROD method for polynomials of degree $N=4$ considering $P_r \in \partial\Omega \setminus \partial\Omega_h$.

$N=4$							
K	h	$R^k=3$		$R^k=4$		$R^k=5$	
		E_∞	p	E_∞	p	E_∞	p
14	9.34e-01	8.51e-03	-	1.44e-03	-	1.35e-03	-
64	4.70e-01	1.79e-03	2.3	3.83e-04	1.9	3.61e-05	5.3
262	2.34e-01	3.77e-04	2.2	1.85e-04	1.0	2.11e-06	4.1
1096	1.13e-01	8.69e-05	2.0	5.17e-05	1.8	4.39e-08	5.3
4316	5.69e-02	2.22e-05	2.0	1.35e-05	2.0	1.53e-09	4.9

As we have seen, to implement the DG-ROD method, we need to alternately apply the DG method and the polynomial reconstruction process. In order to improve the effectiveness of the method, in the next chapter we will consider an approach that avoids this iterative process.

Chapter 5

A variant of the Nelder–Mead algorithm

Motivated by our application of interest, which consists in solving a problem on a curved boundary domain, we aim to present a strategy to overcome the difficulties in the boundary treatment. As we have seen, in the last chapter, the DG-ROD method is based on a polynomial reconstruction of the boundary condition imposed on Ω_h. The coefficients of the reconstructions are determined such that the polynomials are close to the numerical solution and adequately satisfy the boundary condition imposed on the physical domain Ω. This requires an iterative process between the DG method and a polynomial reconstruction. In order to avoid that iterative process, in this chapter, we suggest a different approach. We aim at determining the boundary condition values that should be imposed on Ω_h such that the difference between the numerical solution and the exact one, both evaluated at a set of point $\boldsymbol{P} \in \partial\Omega$, is minimised. Consider the Helmholtz equation with Dirichlet boundary conditions

$$\begin{aligned} -\nabla^2 u(\boldsymbol{x}) - \nu^2 u(\boldsymbol{x}) &= f(\boldsymbol{x}), \quad \boldsymbol{x} \in \Omega \\ u(\boldsymbol{x}) - g(\boldsymbol{x}) &= 0, \quad \boldsymbol{x} \in \partial\Omega \end{aligned} \tag{5.1}$$

and assume that the computational boundary condition is given by

$$u(\boldsymbol{x};\boldsymbol{b}) - \boldsymbol{b} = 0, \quad \boldsymbol{x} \in \partial\Omega_h. \tag{5.2}$$

where $\boldsymbol{b} = [b_i]_{i=1}^{(N+1)N_b}$ is the vector of the decision variables which represent the ideal boundary conditions values on the interpolation points on $\partial\Omega_h$, $(N+1)$ is the number of interpolation points on each side of each element T^k and N_b is the number of elements T^k with a common edge with the computational boundary $\partial\Omega_h$. Therefore, our goal is to determine the vector $\boldsymbol{b}$ such that

$$\boldsymbol{b}^* = \underset{\boldsymbol{b}\in\mathbb{R}^{(N+1)N_b}}{\arg\min} ||u_h(\boldsymbol{P};\boldsymbol{b}) - g(\boldsymbol{P})||_\infty,$$

where $u_h(\cdot;\boldsymbol{b})$ is the DG solution for the Helmholtz problem with boundary conditions defined as in (5.2), $\boldsymbol{P}$ is the set of all points on $\partial\Omega$ used and $\boldsymbol{b}^*$ provides the computational boundary values. Thus, by solving this minimisation problem we get the optimal boundary conditions in the polygonal domain in order to the DG method obtain directly the solution that best fits the boundary conditions in the curved domain.

To solve the minimisation problem, we propose a method that is a variant of the Nelder-Mead (NM) algorithm. The NM method [23] is a direct search algorithm for solving unconstrained minimisation problems. Since its publication, this method gained high popularity in several application areas due to its simplicity and its ability to adapt to the objective function. Despite being one of the most popular and widely used derivative-free methods, quite a few results are known on the convergence of the NM method [18, 19]. Moreover, the method may fail to converge to a stationary point of the objective function f [20]. Another method that is also used to solve minimisation problems is the directional direct search method, which guarantees the global convergence of the algorithm and uses positive basis. The new method proposed in this thesis can be seen as a combination of the two methods just mentioned, working as a modified NM algorithm that takes into account the cosine measure of a positive basis, ensuring control on the geometry of the simplices of the method.

We start by describing the Nelder-Mead method in Section 5.1 and by presenting a brief analysis of its convergence. We review the directional direct search method as well some basic properties of positive spanning sets and positive bases in Section 5.2. The new method is described in Section 5.3 as a variant of the NM method and some of its theoretical proprieties are shown in Section 5.4. In Section 5.5, we present the numerical results for the Helmholtz problem with homogeneous boundary conditions, comparing the results obtained with the classical NM method and with its variant suggested in this thesis.

5.1 Nelder-Mead method

The Nelder-Mead algorithm [23], first published in 1965, is one of the most popular derivative-free methods. Since its publication, the NM method has been widely used in various application areas [7, 24]. The reasons for its success are its simplicity, the fact that it does not require derivatives and its ability to adapt to the curvature of the function [7]. Despite its wide use, few theoretical results are known on the convergence of the NM method.

5.1.1 Brief description

The Nelder-Mead algorithm is a derivative-free method for solving the unconstrained optimisation problem

$$\min_{\boldsymbol{x}\in\mathbb{R}^n} f(\boldsymbol{x}), \tag{5.3}$$

for functions $f:\mathbb{R}^n\to\mathbb{R}$. The main idea of this method is to generate a sequence of simplices to approximate an optimal point of (5.3). A simplex of dimension n is the convex hull of an affinely independent set of points $S=\{\boldsymbol{x}_1,\ldots,\boldsymbol{x}_{n+1}\}$ [7]. The affine independence of an $(n+1)-$family $(\boldsymbol{x}_1,\ldots,\boldsymbol{x}_{n+1})$ is equivalent to linear independence of one/all of the $n-$families $(\boldsymbol{x}_1-\boldsymbol{x}_i,\ldots,\boldsymbol{x}_{i-1}-\boldsymbol{x}_i,\boldsymbol{x}_{i+1}-\boldsymbol{x}_i,\ldots,\boldsymbol{x}_{n+1}-\boldsymbol{x}_i)$, $i=1,\ldots,n+1$ [5]. We denote the vertices of the simplex by $\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_{n+1}$. A simplex of dimension 0 is a point, of dimension 1 is a closed line segment, of dimension 2 is a triangle and of dimension 3 is a tetrahedron.

At each iteration, the vertices $\boldsymbol{x}_j$, $j=1,\ldots,n+1$, of the simplex are ordered by increasing values of the objective function f

$$f(\boldsymbol{x}_{best})\leq\cdots\leq f(\boldsymbol{x}_{bad})\leq f(\boldsymbol{x}_{worst}).$$

The worst point, $\boldsymbol{x}_{worst}$, is replaced by a new point in the line that connects $\boldsymbol{x}_{worst}$ and $\hat{\boldsymbol{c}}$,

$$\hat{\boldsymbol{c}} + \alpha_j(\hat{\boldsymbol{c}} - \boldsymbol{x}_{worst}), \quad \alpha_j \in \mathbb{R},$$

where

$$\hat{\boldsymbol{c}} = \frac{1}{n}\left[\left(\sum_{i=1}^{n+1} \boldsymbol{x}_i\right) - \boldsymbol{x}_{worst}\right]$$

is the centroid of the best n vertices and the value α_j, $j \in \{r, e, oc, ic\}$, indicates the type of iteration: reflection (r), expansion (e), outer contraction (oc) and inner contraction (ic). The standard values for these parameters are $\alpha_r = 1$, $\alpha_e = 2$, $\alpha_{oc} = \frac{1}{2}$ and $\alpha_{ic} = -\frac{1}{2}$. We denote the reflected point by $\boldsymbol{x}_r$, the expansion point by $\boldsymbol{x}_e$, the outer contraction point by $\boldsymbol{x}_{oc}$ and the inner contraction point by $\boldsymbol{x}_{ic}$. The algorithm can also performs a simplex shrink, where all the vertices, except $\boldsymbol{x}_{best}$, are shrinking by the simplex at $\boldsymbol{x}_{best}$. The new n vertices are computed by $\boldsymbol{x}_{best} + \alpha_s(\boldsymbol{x}_j - \boldsymbol{x}_{best})$, $j = 1, \ldots, n+1$, such that $\boldsymbol{x}_j \neq \boldsymbol{x}_{best}$. The typical value for the shrink coefficient is $\alpha_s = \frac{1}{2}$. In order to replace $\boldsymbol{x}_{worst}$, first the algorithm tries the reflected point and analyses if $f(\boldsymbol{x}_r)$ is lower than $f(\boldsymbol{x}_{bad})$. If unsuccessful or if $\boldsymbol{x}_r$ is the best point known, the algorithm examines one of the two contractions points or the expansion point, respectively. If an acceptable point is found, then that point replaces $\boldsymbol{x}_{worst}$. Otherwise, the algorithm performs a shrink. In both cases, a new simplex is produced. The Nelder-Mead method is described in Algorithm 3 and the possible transformations are represented in Figure 5.1.

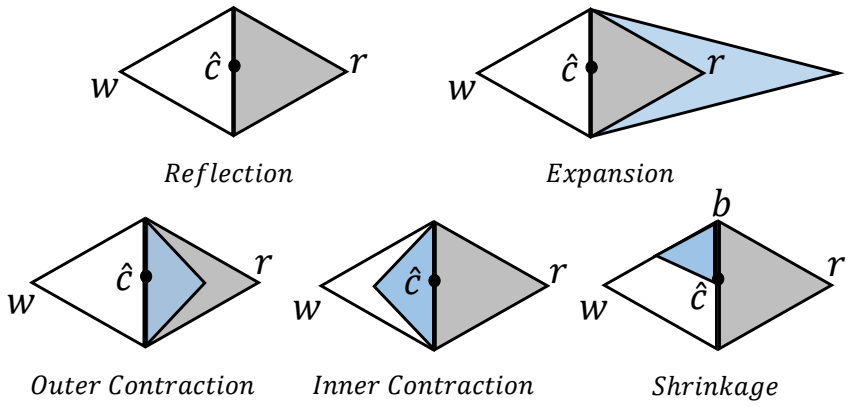


Fig. 5.1 The five possible transformations of a simplex, where $\hat{\boldsymbol{c}}$ is the centroid and w, r and b denote $\boldsymbol{x}_{worst}$, $\boldsymbol{x}_r$ and $\boldsymbol{x}_{best}$, respectively. Adapted from [14].

Stopping conditions for Algorithm 3 could consist of terminating the run when acceptably small values are obtained for the step size or the change in the value of the objective function.

The original NM algorithm paper [23] contains several ambiguities about strictness of inequalities and tie breaking. The algorithm described in Algorithm 3 is consider the "modern interpretation" of the NM method [18] and the major difference from the original algorithm is that in the original version the expansion point $\boldsymbol{x}_e$ is accepted if $f(\boldsymbol{x}_e) < f(\boldsymbol{x}_{best})$. This method, as described in Algorithm 3, is implemented in the MATLAB as the function `fminsearch`. The initial simplex is constructed

Algorithm 3 Nelder-Mead method

Initialization:

- Choose the initial simplex of vertices $S^{(1)} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{n+1}\}$ and evaluate f at these points;
- Choose the coefficients: $0 < \alpha_s < 1$ and $-1 < \alpha_{ic} < 0 < \alpha_{oc} < \alpha_r < \alpha_e$.

For $it = 1, 2, \ldots$

0. Set $S = S^{(it)}$.
1. **Order**: Sort the vertices of the simplex so that their function values are in ascending order
$$f(\boldsymbol{x}_{best}) \leq \cdots \leq f(\boldsymbol{x}_{bad}) \leq f(\boldsymbol{x}_{worst})$$
2. **Reflect**: Calculate $\boldsymbol{x}_r = \hat{\boldsymbol{c}} + \alpha_r(\hat{\boldsymbol{c}} - \boldsymbol{x}_{worst})$, where $\hat{\boldsymbol{c}} = \frac{1}{n}\left[\left(\sum_{i=1}^{n+1} \boldsymbol{x}_i\right) - \boldsymbol{x}_{worst}\right]$. If $f(\boldsymbol{x}_{best}) \leq f(\boldsymbol{x}_r) < f(\boldsymbol{x}_{bad})$, then replace $\boldsymbol{x}_{worst}$ by $\boldsymbol{x}_r$ and terminate the iteration: $S^{(it+1)} = S^{(it)} \setminus \{\boldsymbol{x}_{worst}\} \cup \{\boldsymbol{x}_r\}$ (Step 6).
3. **Expand:** If $f(\boldsymbol{x}_r) < f(\boldsymbol{x}_{best})$, then calculate $\boldsymbol{x}_e = \hat{\boldsymbol{c}} + \alpha_e(\hat{\boldsymbol{c}} - \boldsymbol{x}_{worst})$. If $f(\boldsymbol{x}_e) < f(\boldsymbol{x}_r)$, replace $\boldsymbol{x}_{worst}$ by $\boldsymbol{x}_e$ and terminate the iteration: $S^{(it+1)} = S^{(it)} \setminus \{\boldsymbol{x}_{worst}\} \cup \{\boldsymbol{x}_e\}$ (Step 6). Otherwise replace $\boldsymbol{x}_{worst}$ by $\boldsymbol{x}_r$ and terminate the iteration: $S^{(it+1)} = S^{(it)} \setminus \{\boldsymbol{x}_{worst}\} \cup \{\boldsymbol{x}_r\}$ (Step 6).
4. **Contract:** If $f(\boldsymbol{x}_r) \geq f(\boldsymbol{x}_{bad})$
   (a) **Outside contraction:** If $f(\boldsymbol{x}_r) < f(\boldsymbol{x}_{worst})$ calculate $\boldsymbol{x}_{oc} = \hat{\boldsymbol{c}} + \alpha_{oc}(\hat{\boldsymbol{c}} - \boldsymbol{x}_{worst})$. If $f(\boldsymbol{x}_{oc}) \leq f(\boldsymbol{x}_r)$, then replace $\boldsymbol{x}_{worst}$ by $\boldsymbol{x}_{oc}$ and terminate the iteration: $S^{(it+1)} = S^{(it)} \setminus \{\boldsymbol{x}_{worst}\} \cup \{\boldsymbol{x}_{oc}\}$ (Step 6). Otherwise, perform a shrink (Step 5).
   (b) **Inner contraction:** If $f(\boldsymbol{x}_{worst}) \leq f(\boldsymbol{x}_r)$, calculate $\boldsymbol{x}_{ic} = \hat{\boldsymbol{c}} + \alpha_{ic}(\hat{\boldsymbol{c}} - \boldsymbol{x}_{worst})$. If $f(\boldsymbol{x}_{ic}) < f(\boldsymbol{x}_{worst})$, then replace $\boldsymbol{x}_{worst}$ by $\boldsymbol{x}_{ic}$ and terminate the iteration: $S^{(it+1)} = S^{(it)} \setminus \{\boldsymbol{x}_{worst}\} \cup \{\boldsymbol{x}_{ic}\}$ (Step 6). Otherwise, perform a shrink (Step 5).
5. **Shrink:** For all $j = 1, \ldots, n+1$ such that $\boldsymbol{x}_j \neq \boldsymbol{x}_{best}$ calculate $\boldsymbol{x}_{sj} = \boldsymbol{x}_{best} + \alpha_s(\boldsymbol{x}_j - \boldsymbol{x}_{best})$. Consider a new simplex with $\boldsymbol{x}_{best}$ and the n new vertices and terminate the iteration: $S^{(it+1)} = \{\boldsymbol{x}_{best}\} \cup \{\boldsymbol{x}_{sj}, j \neq best\}$ (Step 6).
6. **Stopping criterion:** If the stopping criterion is not satisfied, increment it to $it+1$ and return to Step 0. Otherwise $\boldsymbol{x}_{best}$ is the value that approximates the minimiser of f.

around the initial point given to the function, say $\boldsymbol{x}_0$, where $\boldsymbol{x}_1 = \boldsymbol{x}_0$. Further n points are obtained by perturbing one of the coordinates of $\boldsymbol{x}_0$ by 5% or 0.00025 if the coordinate value is zero, i.e, for $i, j = 1, \ldots, n$

$$x_{j+1,i} = \begin{cases} x_{0,i}, & \text{if} \quad i \neq j \\ (1+0.05)x_{0,i}, & \text{if} \quad i = j \quad \wedge \quad x_{0,i} \neq 0, \\ 0.00025, & \text{if} \quad i = j \quad \wedge \quad x_{0,i} = 0, \end{cases} \tag{5.4}$$

where $x_{j+1,i}$ denotes the i-th coordinate of $\boldsymbol{x}_{j+1}$.

As stopping criterions, `fminsearch` considers the maximum number of function evaluations (the default value is $200 \times n$), the maximum number of iterations allowed (the default value is $200 \times n$), the change in the value of the objective function during a step (the default value is $TolFun =$ 1e-04) and the size of a step (the default value is $TolX =$ 1e-04). The function `fminsearch` stops when it satisfies both $TolFun$ and $TolX$. Note that, for the default values considered, the maximum number of

function evaluations is always reached at the same time or before the maximum number of iterations allowed since the method performs at least one function evaluation per iteration.

The NM method does not require calculating the derivative of the objective function. Moreover, it has a robust approach, in the sense that it can be used to find the optimal point of a wide range of functions (since the simplex fits the function). Other advantage is the fact that the algorithm performs a relatively small number of function evaluations per iteration (1 if the iteration is a reflection, 2 if the iteration is an expansion or contraction and $n+2$ if the iteration is a shrink). However, when increasing the number of variables the efficiency of the method may be reduced. Furthermore, it has been shown that the method may fail to converge to a stationary point [20].

5.1.2 A convergence analysis of the Nelder-Mead method

Despite being a widely used method, only a few theoretical results are known on the convergence of this method. A famous two dimensional example by McKinnon [20] shows that NM algorithm may fail to converge to a stationary point of f, even if f is strictly convex and has continuous derivatives. In the examples presented in [20], the method repeatedly applies the inside contraction step with the best vertex remaining fixed. McKinnon referred to this behaviour as repeated focused inside contraction (RFIC). No other type of step occurs for these examples. The simplicies tend to a straight line which is orthogonal to the steepest descent direction. The functions are defines as

$$f(x,y) = \begin{cases} \theta\phi|x|^{\tau} + y + y^2, & x < 0, \\ \theta|x|^{\tau} + y + y^2, & x \geq 0, \end{cases} \tag{5.5}$$

where θ and ϕ are positive constants. Note that $(0,-1)$ is a descent direction from the origin. The function is strictly convex if $\tau > 1$. It has continuous first derivatives if $\tau > 1$, continuous second derivatives if $\tau > 2$, and continuous third derivatives if $\tau > 3$. The initial simplex of the NM method is defined by the following vertices

$$\boldsymbol{x}_1 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \qquad \boldsymbol{x}_2 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad \text{and} \quad \boldsymbol{x}_3 = \begin{bmatrix} \frac{1+\sqrt{33}}{8} \\ \frac{1-\sqrt{33}}{8} \end{bmatrix}. \tag{5.6}$$

For values of θ, ϕ and τ that satisfy a certain condition, the method converge to the origin, which is not a stationary point [20]. An example of values that satisfy that condition are $\tau = 2$, $\theta = 6$ and $\phi = 60$. The NM algorithm converges to the origin (which is the best vertex of the initial simplex) rather than to the minimiser $(0,-1/2)$, performing an infinite sequence of inside contractions.

For strictly convex objective functions with bounded level sets, [18] showed the convergence of the NM algorithm to the minimiser in one dimension. For such functions of two variables, it was shown that the function values at the simplex vertices converge to the same value and that the diameter of the simplices converges to zero.

To avoid the simplices to become needle-shaped, in [19], it was suggested a restricted Nelder–Mead algorithm in two dimensions that does not allow expansion steps. The authors proved that, in certain conditions, the algorithm always converges to the minimiser.

As we have seen, in the examples by McKinnon, the NM method can stagnate and fail to converge a stationary point due to the deterioration of the simplex geometry. Motivated by this fact, our goal is to find a strategy to ensure the control of the geometry of the simplices of a variant of this method.

5.2 Directional direct search method and positive basis

In this section, we introduce the directional direct search method of the directional type [7], which ensures the global convergence. Before introducing this method, we start by presenting some basic properties of positive spanning sets and positive basis [7, 26], which are used in this algorithm.

The positive span of a finite set of vectors $S = \{\boldsymbol{d}_1, \ldots, \boldsymbol{d}_k\} \subset \mathbb{R}^n$, denoted by $\mathrm{pos}\,(\mathrm{S})$, is the convex cone [1] is given by

$$\mathrm{pos}\,(\mathrm{S}) = \{\lambda_1 \boldsymbol{d}_1 + \cdots + \lambda_\mathrm{k} \boldsymbol{d}_\mathrm{k} : \lambda_\mathrm{i} \geq 0\}.$$

We say that a finite set $S \subset \mathbb{R}^n$ is a positive spanning set of a convex cone $C \in \mathbb{R}^n$ if $\mathrm{pos}\,(\mathrm{S}) = \mathrm{C}$. In this case, S is said to positively span the convex cone C. In particular, S positively spans $\mathbb{R}^n$ if $\mathrm{pos}\,(\mathrm{S}) = \mathbb{R}^\mathrm{n}$.

Theorem 5.2.1. *If $S = \{\boldsymbol{d}_1, \ldots, \boldsymbol{d}_k\} \subset \mathbb{R}^n$ positively spans the linear subspace V, then $S \setminus \{\boldsymbol{d}_i\}$ linearly spans V for any $i = 1, \cdots, k$.*

The proof of this theorem can be seen in [26]. The following result, whose proof can be seen in [26], guarantees the existence of a descent direction among the vectors in a positive spanning set of $\mathbb{R}^n$ when the gradient of the objective function is not zero.

Theorem 5.2.2. *Suppose that $S = \{\boldsymbol{d}_1, \ldots, \boldsymbol{d}_k\} \subset \mathbb{R}^n$ is a set of vectors in $\mathbb{R}^n$. Then S positively spans $\mathbb{R}^n$ if and only if for every nonzero vector $\boldsymbol{w} \in \mathbb{R}^n$, there exist an index $i \in \{1, \ldots, k\}$ such that $\boldsymbol{w}^T \boldsymbol{d}_i > 0$.*

A set of vectors $S = \{\boldsymbol{d}_1, \ldots, \boldsymbol{d}_k\} \subset \mathbb{R}^n$ is said to be positively dependent if some $\boldsymbol{x}_i$ is a positive combination of the others; otherwise, it is positively independent. That is, S is positively independent if $\boldsymbol{d}_i \notin \mathrm{pos}\,(\mathrm{S} \setminus \boldsymbol{d}_\mathrm{i})$ for all $i = 1, \ldots, k$.

The set of vectors $S = \{\boldsymbol{d}_1, \ldots, \boldsymbol{d}_k\} \subset \mathbb{R}^n$ is a positive basis in $\mathbb{R}^n$ if S is a positively independent set whose positive spans is $\mathbb{R}^n$. The next theorem, whose proof can be seen in [26], provides a procedure to construct a positive basis from a basis.

Theorem 5.2.3. *Let $B = \{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k\}$ be a basis of a linear subspace V of $\mathbb{R}^n$ and let $\mathfrak{J} = \{J_1, \ldots, J_\ell\}$ be a collection of subsets of $K = \{1, \ldots, k\}$ such that $\bigcup_{i=1}^{\ell} J_i = K$ and such that $J_r \not\subseteq \bigcup_{\substack{i=1 \\ i \neq r}}^{\ell} J_i$ for all $r = 1, \ldots, \ell$. Then the set $\tilde{B} = B \bigcup \{ -\sum_{j \in J_1} \lambda_{1,j} \boldsymbol{v}_j, \ldots, -\sum_{j \in J_\ell} \lambda_{\ell,j} \boldsymbol{v}_j \}$, where $\lambda_{i,j} > 0$ for all $i = 1, \ldots, \ell$, $j \in J_i$, is a positive basis of V.*

We define the cosine measure of a positive spanning set (with nonzero vectors) D is defined by

$$\mathtt{cm}(D) = \min_{0 \neq \boldsymbol{v} \in \mathbb{R}^n} \max_{\boldsymbol{d} \in D} \frac{\boldsymbol{v}^T \boldsymbol{d}}{||\boldsymbol{v}||||\boldsymbol{d}||}.$$

[1] A set $C \in \mathbb{R}^n$ is called a convex cone if for any $\boldsymbol{x}_1, \boldsymbol{x}_2 \in C$ and $\lambda_1, \lambda_2 \geq 0$, $\lambda_1 \boldsymbol{x}_1 + \lambda_2 \boldsymbol{x}_2 \in C$.

Note that $\mathtt{cm}(D) \in (0,1)$. Moreover, values of the cosine measure close to zero indicate a deterioration of the positive spanning property.

For example, in $\mathbb{R}^2$, the positive basis $\{(1,0),(0,1),(-1,0),(0,-1)\}$ has a cosine measure equals to $\cos(\pi/4) = \sqrt{2}/2$.

Algorithm 4 Directional direct search method

Initialization:
Choose the initial point $\boldsymbol{x}^{(1)}$ and choose $\alpha_0 > 0$, $\beta_1 \geq 1$ and $0 < \beta_2 \leq \beta_3 < 1$. Let D be a set of positive basis.
For $it = 1, 2, \ldots$

1. **Search step:** Try to compute a point with $f(\boldsymbol{x}) < f(\boldsymbol{x}^{(it)})$ by evaluating the function f at a finite number of points. If such a point is found, then set $\boldsymbol{x}^{(it+1)} = \boldsymbol{x}$, declare the iteration and the search step successful, and skip the poll step.
2. **Poll step:** Choose a positive basis $D^{(it)}$ from the set D. Order the poll set $P^{(it)} = \{\boldsymbol{x}^{(it)} + \alpha^{(it)}\boldsymbol{d} : \boldsymbol{d} \in D^{(it)}\}$. Start evaluating f at the poll points following the chosen order. If a poll point $\boldsymbol{x}^{(it)} + \alpha^{(it)}\boldsymbol{d}$ is found such that $f(\boldsymbol{x}^{(it)} + \alpha^{(it)}\boldsymbol{d}) < f(\boldsymbol{x}^{(it)})$, then stop polling, set $\boldsymbol{x}^{(it+1)} = \boldsymbol{x}^{(it)} + \alpha^{(it)}\boldsymbol{d}$, and declare the iteration and the poll step successful. Otherwise, declare the iteration (and the poll step) unsuccessful and set $\boldsymbol{x}^{(it+1)} = \boldsymbol{x}^{(it)}$.
3. **Step parameter update**: If the iteration was successful, then maintain or increase the step size parameter: $\alpha^{(it+1)} \in [\alpha^{(it)}, \beta_1\alpha^{(it)}]$. Otherwise, decrease the step size parameter: $\alpha^{(it+1)} \in [\beta_2\alpha^{(it)}, \beta_3\alpha^{(it)}]$.

Now we present the directional direct search method, which is described in Algorithm 4. The main idea of this algorithm at each iteration it is to find a new point $\boldsymbol{x}^{(it+1)}$ such that $f(\boldsymbol{x}^{(it+1)} < f(\boldsymbol{x}^{(it)})$. The process of finding the new point can be described in two phases: the search step and the poll step. The search step is optional and it consists of evaluating the function at a finite number of points (the choice of the points is arbitrary). The poll step is performed only if the search step has been unsuccessful. It consists of a local search around the current point $\boldsymbol{x}^{(it)}$, exploring the set of points $P^{(it)} = \{\boldsymbol{x}^{(it)} + \alpha^{(it)}\boldsymbol{d}, \quad \boldsymbol{d} \in D^{(it)}\}$, $\alpha^{(it)} > 0$ and $D^{(it)}$ is a positive basis or a positive spanning set. The new point can be accepted based on a simple decrease of the objective function: $f(\boldsymbol{x}^{(it+1)}) < f(\boldsymbol{x}^{(it)})$. As an alternative, it can be imposed a sufficient decrease condition: $f(\boldsymbol{x}^{(it+1)}) \leq f(\boldsymbol{x}^{(it)}) - \rho(\alpha^{(it)})$, where the forcing function $\rho : \mathbb{R}^+ \to \mathbb{R}^+$ is continuous, positive and satisfies

$$\lim_{t \to 0^+} \frac{\rho(t)}{t} = 0 \quad \text{and} \quad \rho(t_1) \leq \rho(t_2) \quad \text{if} \quad t_1 < t_2. \tag{5.7}$$

A simple example of a forcing function is $\rho(t) = Ct^2$, with C a positive constant.

Therefore, this method guarantees the possibility of decreasing the value of the function until a stationary point is reached. Moreover, there is a control in the geometry of the poll directions $\boldsymbol{d} \in D^{(it)}$ in order to guarantee that all points in a neighborhood can be reached (since the poll directions constitute a positive spanning set of $\mathbb{R}^n$). The algorithm also allows imposing a relation between the decrease of the value of the objective function and the step length (via the forcing function $\rho(\cdot)$).

5.3 NM method and step size control

As we have seen, despite the popularity of the NM method, its convergence to a stationary point is not guaranteed. In this section, we suggest a modification of the classical NM. The new proposed method aims to take advantage of the classical NM method and the directional direct search method. The NM method requires a relatively small number of function evaluations in each iteration and it is a simple algorithm to understand. On the other hand, the directional direct search method has the advantage of guaranteeing the convergence of the method. One disadvantage on the classical NM method is the fact that the simplices can become arbitrarily flat or needle-shaped. Thus, in order to control the geometry of the simplex, we use the cosine measure of a positive basis defined by the possible poll directions in the simplex.

Let us consider $S = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{n+1}\} \subset \mathbb{R}^n$ the set of vertices of a simplex in $\mathbb{R}^n$. The NM method considers $\hat{\boldsymbol{c}} = \frac{1}{n}\left[\left(\sum_{i=1}^{n+1} \boldsymbol{x}_i\right) - \boldsymbol{x}_{worst}\right]$, now we assume that $\boldsymbol{c} = \frac{1}{n+1}\sum_{i=1}^{n+1} \boldsymbol{x}_i$. We denote by $D(S)$ the following set $D(S) = \{\boldsymbol{d}_i = \boldsymbol{c} - \boldsymbol{x}_i,\ i = 1, \ldots, n+1\}$, which is the set of possible directions (in Proposition 5.4.1, we assure that $D(S)$ is a positive basis in $\mathbb{R}^n$ and therefore it can be considered a set of poll directions). If the algorithm performs a nonshrink step, we consider $P = \{\boldsymbol{x}_i + \alpha_j \boldsymbol{d}_i,\ \boldsymbol{d}_i \in D(S),\ i = 1, \ldots, n+1,\ j \in \{r, e, oc, ic\}\}$ the poll points. On other hand, if the algorithm performs a shrink step we do the procedure as described in Algorithm 3. This changes are illustrated in Figure 5.2.

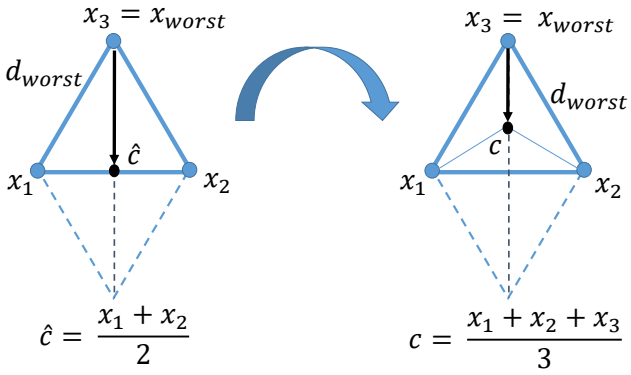


Fig. 5.2 Classical NM method (left) and modified NM method (right).

A crucial step in this new method is to control the geometry of the simplex through the cosine measure. In the classical NM method when the simplices become arbitrarily flat or needle-shape, the cosine measure is close to zero. Thus, we require the cosine measure to be above a positive threshold, $\gamma > 0$, i.e., that $\mathtt{cm}(D) \geq \gamma$. The next proposition establish that if the correspondent positive basis of a simplex satisfies $\mathtt{cm}(D) \geq \gamma$, then the same is true for the new positive basis obtained from the reflected simplex.

Proposition 5.3.1. *Let S be the set of vertices of a simplex in $\mathbb{R}^n$ and $\overline{S}$ be the set of vertices obtained from the reflected simplex. If $\mathtt{cm}(D(S)) \geq \gamma$, then $\mathtt{cm}(D(\overline{S})) \geq \gamma$.*

Proof. We consider that we have a (genuine or isometric) reflection, therefore the angles measure is preserved. Thus, if $\mathtt{cm}(D(S)) \geq \gamma$, then $\mathtt{cm}(D(\overline{S})) \geq \gamma$. □

However, even if the initial simplex satisfies this condition the simplex obtained by contraction or expansion may not satisfy that condition. In these case, the step size has to be increase or decreased up to α_r, respectively, until the condition is satisfied.

In this new method, we also require a sufficient decrease condition to accept new iterates. The method is described in Algorithm 5.

Algorithm 5 Nelder-Mead method with step size control

Initialization:

- Choose the initial point $\boldsymbol{x}_1^{(1)}$. The other n vertices of the initial simplex are determined as in (5.4);
- Evaluate f at $S^{(1)} = \{\boldsymbol{x}_1^{(1)}, \ldots, \boldsymbol{x}_{n+1}^{(1)}\}$;
- Choose the coefficients: $0 < \alpha_s < 1$, $0 < \alpha_{ic} < \frac{n+1}{n} < \alpha_{oc} < \alpha_r < \alpha_e$ and $\gamma > 0$;
- Choose the forcing function $\rho(\cdot)$ (5.7):
- Consider $\boldsymbol{c}^{(1)} = \frac{1}{n+1}\sum_{i=1}^{n+1} \boldsymbol{x}_i^{(1)}$ and $D(S^{(1)}) = \{\boldsymbol{d}_i^{(1)} = \boldsymbol{c}^{(1)} - \boldsymbol{x}_i^{(1)}, i = 1, \ldots, n+1\}$. If $\mathtt{cm}(D(S^{(1)})) \geq \gamma$, continue.

For $it = 1, 2, \ldots$

0. Set $S = S^{(it)}$.
1. **Order**: Sort the vertices of the simplex so that their function values are in ascending order
$$f(\boldsymbol{x}_{best}) \leq \cdots \leq f(\boldsymbol{x}_{bad}) \leq f(\boldsymbol{x}_{worst})$$
2. **Reflect**: Calculate $\boldsymbol{x}_r = \boldsymbol{x}_{worst} + \alpha_r \boldsymbol{d}_{worst}$. If $f(\boldsymbol{x}_{best}) \leq f(\boldsymbol{x}_r) \leq f(\boldsymbol{x}_{bad}) - \rho(\|\boldsymbol{d}_{worst}\|)$, then replace $\boldsymbol{x}_{worst}$ by $\boldsymbol{x}_r$ and terminate the iteration: $S^{(it+1)} = S^{(it)} \setminus \{\boldsymbol{x}_{worst}\} \cup \{\boldsymbol{x}_r\}$ (Step 6).
3. **Expand:** If $f(\boldsymbol{x}_r) < f(\boldsymbol{x}_{best})$, then calculate $\boldsymbol{x}_e = \boldsymbol{x}_{worst} + \alpha_e \boldsymbol{d}_{worst}$. If $\mathtt{cm}(D(S^{(it)} \setminus \{\boldsymbol{x}_{worst}\} \cup \{\boldsymbol{x}_e\})) < \gamma$, decrease α_e until $\mathtt{cm}(D(S^{(it)} \setminus \{\boldsymbol{x}_{worst}\} \cup \{\boldsymbol{x}_e\})) \geq \gamma$. Replace $\boldsymbol{x}_{worst}$ by $\boldsymbol{x}_e$ ($S^{(it+1)} = S^{(it)} \setminus \{\boldsymbol{x}_{worst}\} \cup \{\boldsymbol{x}_e\}$) or $\boldsymbol{x}_r$ ($S^{(it+1)} = S^{(it)} \setminus \{\boldsymbol{x}_{worst}\} \cup \{\boldsymbol{x}_r\}$), taking into account if $f(\boldsymbol{x}_e) \leq f(\boldsymbol{x}_r) - \rho(\|\boldsymbol{d}_{worst}\|)$ or not and terminate the iteration: (Step 6).
4. **Contract:** If $f(\boldsymbol{x}_r) > f(\boldsymbol{x}_{bad}) - \rho(\|\boldsymbol{d}_{worst}\|)$
   (a) **Outside contraction:** If $f(\boldsymbol{x}_r) < f(\boldsymbol{x}_{worst})$ calculate $\boldsymbol{x}_{oc} = \boldsymbol{x}_{worst} + \alpha_{oc}\boldsymbol{d}_{worst}$. If $\mathtt{cm}(D(S^{(it)} \setminus \{\boldsymbol{x}_{worst}\} \cup \{\boldsymbol{x}_{oc}\})) < \gamma$, increase α_{oc} until $\mathtt{cm}(D(S^{(it)} \setminus \{\boldsymbol{x}_{worst}\} \cup \{\boldsymbol{x}_{oc}\})) \geq \gamma$. If $f(\boldsymbol{x}_{oc}) \leq f(\boldsymbol{x}_r) - \rho(\|\boldsymbol{d}_{worst}\|)$, then replace $\boldsymbol{x}_{worst}$ by $\boldsymbol{x}_{oc}$ and terminate the iteration: $S^{(it+1)} = S^{(it)} \setminus \{\boldsymbol{x}_{worst}\} \cup \{\boldsymbol{x}_{oc}\}$ (Step 6). Otherwise, perform a shrink (Step 5).
   (b) **Inner contraction:** If $f(\boldsymbol{x}_{worst}) \leq f(\boldsymbol{x}_r)$, calculate $\boldsymbol{x}_{ic} = \boldsymbol{x}_{worst} + \alpha_{ic}\boldsymbol{d}_{worst}$. If $\mathtt{cm}(D(S^{(it)} \setminus \{\boldsymbol{x}_{worst}\} \cup \{\boldsymbol{x}_{ic}\})) < \gamma$, increase α_{ic} until $\mathtt{cm}(D(S^{(it)} \setminus \{\boldsymbol{x}_{worst}\} \cup \{\boldsymbol{x}_{ic}\})) \geq \gamma$. If $f(\boldsymbol{x}_{ic}) < f(\boldsymbol{x}_{worst}) - \rho(\|\boldsymbol{d}_{worst}\|)$, then replace $\boldsymbol{x}_{worst}$ by $\boldsymbol{x}_{ic}$ and terminate the iteration: $S^{(it+1)} = S^{(it)} \setminus \{\boldsymbol{x}_{worst}\} \cup \{\boldsymbol{x}_{ic}\}$ (Step 6). Otherwise, perform a shrink (Step 5).
5. **Shrink:** For all $j = 1, \ldots, n+1$ such that $\boldsymbol{x}_j \neq \boldsymbol{x}_{best}$ calculate $\boldsymbol{x}_{sj} = \boldsymbol{x}_{best} + \alpha_s(\boldsymbol{x}_j - \boldsymbol{x}_{best})$. Consider a new simplex with $\boldsymbol{x}_{best}$ and the n new vertices and terminate the iteration: $S^{(it+1)} = \boldsymbol{x}_{best} \cup \{\boldsymbol{x}_{sj},\ j \neq best\}$ (Step 6).
6. **Stopping criterion:** If the stopping criterion is not satisfied, increment it to $it+1$ and return to Step 0. Otherwise $\boldsymbol{x}_{best}$ is the value that approximates the minimiser of f.

We consider

$$\alpha_s = \frac{1}{2}, \quad \alpha_r = 2\frac{n+1}{n}, \quad \alpha_e = 3\frac{n+1}{n}, \quad \alpha_{oc} = \frac{3}{2}\frac{n+1}{n}, \quad \alpha_{ic} = \frac{1}{2}\frac{n+1}{n}, \tag{5.8}$$

where n is the number of variables. Assuming this values, $||\boldsymbol{x}_{ic} - \hat{\boldsymbol{c}}|| = \frac{1}{2}||\boldsymbol{x}_{worst} - \hat{\boldsymbol{c}}||$, $||\boldsymbol{x}_r - \hat{\boldsymbol{c}}|| = ||\boldsymbol{x}_{worst} - \hat{\boldsymbol{c}}||$, $||\boldsymbol{x}_e - \hat{\boldsymbol{c}}|| = 2||\boldsymbol{x}_{worst} - \hat{\boldsymbol{c}}||$, $||\boldsymbol{x}_{oc} - \hat{\boldsymbol{c}}|| = \frac{1}{2}||\boldsymbol{x}_{worst} - \hat{\boldsymbol{c}}||$. For $\alpha_{ic} = \frac{n+1}{n}$, $\boldsymbol{x}_{ic} = \hat{\boldsymbol{c}}$. The case for $n = 2$ is illustrated in Figure 5.3.

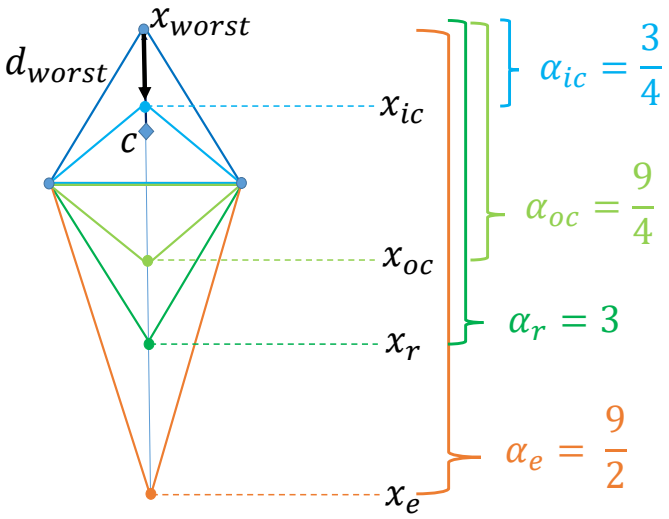


Fig. 5.3 Transformations of a simplex for $n = 2$ and its coeficients α_j, $j \in \{r, e, oc, ic\}$.

We use the NM method with step size control to solve the following minimisation problem

$$\min_{\boldsymbol{b} \in \mathbb{R}^{(N+1)N_b}} ||u_h(\boldsymbol{P}; \boldsymbol{b}) - g(\boldsymbol{P})||_\infty, \tag{5.9}$$

and obtain the optimal solution $\boldsymbol{b}^*$. Then, we solve the Helmholtz equation on a polygonal computational domain with boundary conditions defined by $\boldsymbol{b}^*$. In this way, we avoid the iterative process described by the DG-ROD method.

5.4 Properties of the new method

In this section we proof some properties of the variant of the NM method suggested in this thesis. We start by analysing the set of directions in each iterations, which forms a positive basis in $\mathbb{R}^n$. The next proposition state that the set of directions $D(S)$ of a simplex S forms a positive basis in $\mathbb{R}^n$.

Proposition 5.4.1. *Let S be the set of the vertices of the a simplex. Then $D(S)$ forms a positive basis in $\mathbb{R}^n$.*

Proof. As $\boldsymbol{x}_i$, $i = 1, \ldots, n+1$ are vertices of a simplex, by definition, the vectors $\boldsymbol{x}_i - \boldsymbol{x}_1$, $i = 2, \ldots, n+1$, are linearly independent. Moreover, we prove that $\boldsymbol{c} - \boldsymbol{x}_i$, $i = 1, \ldots, n$, are linearly independent. Let

$\beta_1,\ldots,\beta_n \in \mathbb{R}$,

$$\begin{aligned}
&\sum_{i=1}^{n} \beta_i (\boldsymbol{c}-\boldsymbol{x}_i) = 0 \\
\Leftrightarrow &\sum_{i=1}^{n} \beta_i \left(\frac{1}{n+1} \sum_{j=1}^{n+1} \boldsymbol{x}_j - \boldsymbol{x}_i \right) = 0 \Leftrightarrow \sum_{i=1}^{n} \beta_i \left(\sum_{\substack{j=1 \\ j\neq i}}^{n+1} \boldsymbol{x}_j - n\boldsymbol{x}_i \right) = 0
\end{aligned}$$

By adding $(n-1)^2(\boldsymbol{x}_1-\boldsymbol{x}_1)$ and reordering the terms, we obtain

$$\begin{aligned}
&\beta_1 (\boldsymbol{x}_2-\boldsymbol{x}_1+\boldsymbol{x}_3-\boldsymbol{x}_1+\cdots+\boldsymbol{x}_{n+1}-\boldsymbol{x}_1)+ \\
&\beta_2 (\boldsymbol{x}_1-\boldsymbol{x}_2+\boldsymbol{x}_3-\boldsymbol{x}_2+\boldsymbol{x}_1-\boldsymbol{x}_1+\cdots+\boldsymbol{x}_{n+1}-\boldsymbol{x}_2+\boldsymbol{x}_1-\boldsymbol{x}_1)+\cdots+ \\
&\beta_n (\boldsymbol{x}_1-\boldsymbol{x}_n+\boldsymbol{x}_2-\boldsymbol{x}_n+\boldsymbol{x}_1-\boldsymbol{x}_1+\cdots+\boldsymbol{x}_{n+1}-\boldsymbol{x}_n+\boldsymbol{x}_1-\boldsymbol{x}_1) = 0 \\
\Leftrightarrow &\beta_1 ((\boldsymbol{x}_2-\boldsymbol{x}_1)+(\boldsymbol{x}_3-\boldsymbol{x}_1)+\cdots+(\boldsymbol{x}_{n+1}-\boldsymbol{x}_1))+ \\
&\beta_2 (-n(\boldsymbol{x}_2-\boldsymbol{x}_1)+(\boldsymbol{x}_3-\boldsymbol{x}_1)+\cdots+(\boldsymbol{x}_{n+1}-\boldsymbol{x}_1))+\cdots+ \\
&\beta_n ((\boldsymbol{x}_2-\boldsymbol{x}_1)+(\boldsymbol{x}_3-\boldsymbol{x}_1)+\cdots-n(\boldsymbol{x}_n-\boldsymbol{x}_1)+(\boldsymbol{x}_{n+1}-\boldsymbol{x}_1)) = 0 \\
\Leftrightarrow &(\beta_1-n\beta_2+\beta_3+\cdots+\beta_n)(\boldsymbol{x}_2-\boldsymbol{x}_1)+(\beta_1+\beta_2-n\beta_3+\cdots+\beta_n)(\boldsymbol{x}_3-\boldsymbol{x}_1)+\cdots+ \\
&(\beta_1+\beta_2+\beta_3+\cdots-n\beta_n)(\boldsymbol{x}_n-\boldsymbol{x}_1)+(\beta_1+\beta_2+\beta_3+\cdots+\beta_n)(\boldsymbol{x}_{n+1}-\boldsymbol{x}_1) = 0.
\end{aligned}$$

Recalling the linear independence of $\{\boldsymbol{x}_i-\boldsymbol{x}_1\}$ we have

$$\begin{bmatrix}
1 & -n & 1 & \ldots & 1 & 1 \\
1 & 1 & -n & \ldots & 1 & 1 \\
1 & 1 & 1 & \ldots & 1 & 1 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
1 & 1 & 1 & \ldots & 1 & -n \\
1 & 1 & 1 & \ldots & 1 & 1
\end{bmatrix}
\begin{bmatrix}
\beta_1 \\ \beta_2 \\ \beta_3 \\ \vdots \\ \beta_{n-1} \\ \beta_n
\end{bmatrix} = \mathbf{0},$$

that implies

$$\begin{bmatrix}
1 & -n & 1 & \ldots & 1 & 1 \\
0 & n+1 & -n-1 & \ldots & 0 & 0 \\
0 & 0 & n+1 & \ldots & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & \ldots & n+1 & -n-1 \\
0 & 0 & 0 & \ldots & 0 & n+1
\end{bmatrix}
\begin{bmatrix}
\beta_1 \\ \beta_2 \\ \beta_3 \\ \vdots \\ \beta_{n-1} \\ \beta_n
\end{bmatrix} = \mathbf{0} \Rightarrow \beta_i = 0, \quad i = 1,\ldots,n.$$

Thus, $\{\boldsymbol{d}_i = \boldsymbol{c}-\boldsymbol{x}_i, i = 1,\ldots,n\}$ is a linear independent set in $\mathbb{R}^n$. Therefore, $B = \{\boldsymbol{d}_i = \boldsymbol{c}-\boldsymbol{x}_i, i = 1,\ldots,n\}$ forms a basis in $\mathbb{R}^n$. Moreover, considering the Theorem 5.2.3, $\tilde{B} = B \cup \{-\sum_{j=1}^{n} \boldsymbol{d}_j\}$ forms

a positive basis in $\mathbb{R}^n$. Note that

$$\begin{aligned}
-\sum_{j=1}^{n} \boldsymbol{d}_j &= -\sum_{j=1}^{n} (\boldsymbol{c}-\boldsymbol{x}_j) = -\sum_{j=1}^{n} \left(\frac{1}{n+1} \sum_{i=1}^{n+1} \boldsymbol{x}_i - \boldsymbol{x}_j \right) \\
&= -\frac{1}{n+1} \sum_{j=1}^{n} \left(\sum_{i=1, i\neq j}^{n+1} \boldsymbol{x}_i - n\boldsymbol{x}_j \right) = -\frac{1}{n+1} \sum_{j=1}^{n} ((-n+n-1)\boldsymbol{x}_j + n\boldsymbol{x}_{n+1}) \\
&= -\frac{1}{n+1} \sum_{j=1}^{n} (n\boldsymbol{x}_{n+1} - \boldsymbol{x}_j) = \frac{1}{n+1} \sum_{j=1}^{n} \boldsymbol{x}_j - \boldsymbol{x}_{n+1} = \boldsymbol{c} - \boldsymbol{x}_{n+1} = \boldsymbol{d}_{n+1}.
\end{aligned}$$

We conclude that $D(S) = \{\boldsymbol{d}_1, \boldsymbol{d}_2, \ldots, \boldsymbol{d}_{n+1}\}$ forms a positive basis in $\mathbb{R}^n$. □

Now, we aim at proving that if the set of directions $D(S^{(it)})$ of the (it)-th simplex forms a positive basis in $\mathbb{R}^n$, then the set of directions of the next simplex also forms a positive basis in $\mathbb{R}^n$.

Proposition 5.4.2. *Let $S^{(it)} = \{\boldsymbol{x}_i^{(it)}, i = 1, \ldots, n+1\}$ be the vertices of the (it)-th simplex. If $D(S^{(it)}) = \{\boldsymbol{d}_i^{it)} = \boldsymbol{c}^{(it)} - \boldsymbol{x}_i^{(it)},\ i = 1, \ldots, n+1\}$ is a positive basis in $\mathbb{R}^n$, then, $D(S^{(it+1)}) = \{\boldsymbol{d}_i^{it+1)} = \boldsymbol{c}^{(it+1)} - \boldsymbol{x}_i^{(it+1)},\ i = 1, \ldots, n+1\}$ also forms a positive basis in $\mathbb{R}^n$.*

Proof. We consider two cases depending if the *it* iteration performs a nonshrink step or if it performs a shrink step.

Consider that the (it) iteration performs a nonshrink step and without loss of generality assume that $\boldsymbol{x}_{worst}^{(it)} = \boldsymbol{x}_1^{(it)}$. Thus $\boldsymbol{x}_1^{(it+1)} = \boldsymbol{x}_1^{(it)} + \alpha_j \boldsymbol{d}_1^{(it)}$, $j \in \{r, e, oc, ic\}$, and $\boldsymbol{x}_i^{(it+1)} = \boldsymbol{x}_i^{(it)}$, $i = 2, \ldots, n+1$. Note that, for $j \in \{r, e, oc, ic\}$

$$\begin{aligned}
\boldsymbol{c}^{(it+1)} &= \frac{1}{n+1} \sum_{i=1}^{n+1} \boldsymbol{x}_i^{(it+1)} = \frac{1}{n+1} \left(\boldsymbol{x}_1^{(it)} + \alpha_j \boldsymbol{d}_1^{(it)} + \sum_{i=2}^{n+1} \boldsymbol{x}_i^{(it)} \right) = \frac{1}{n+1} \left(\boldsymbol{x}_1^{(it)} + \alpha_j (\boldsymbol{c}^{(it)} - \boldsymbol{x}_1^{(it)}) + \sum_{i=2}^{n+1} \boldsymbol{x}_i^{(it)} \right) \\
&= \frac{1}{n+1} \left(\alpha_j \boldsymbol{c}^{(it)} - \alpha_j \boldsymbol{x}_1^{(it)} + \sum_{i=1}^{n+1} \boldsymbol{x}_i^{(it)} \right) = \frac{1}{n+1} \left((n+1+\alpha_j) \boldsymbol{c}^{(it)} - \alpha_j \boldsymbol{x}_1^{(it)} \right).
\end{aligned}$$

Moreover, for $j \in \{r, e, oc, ic\}$

$$\begin{aligned}
\boldsymbol{d}_1^{(it+1)} &= \boldsymbol{c}^{(it+1)} - \boldsymbol{x}_1^{(it+1)} = \frac{1}{n+1} \left((n+1+\alpha_j) \boldsymbol{c}^{(it)} - \alpha_j \boldsymbol{x}_1^{(it)} \right) - \boldsymbol{x}_1^{(it)} - \alpha_j (\boldsymbol{c}^{(it)} - \boldsymbol{x}_1^{(it)}) \\
&= \frac{n+1-n\alpha_j}{n+1} (\boldsymbol{c}^{(it)} - \boldsymbol{x}_1^{(it)}) = \frac{n+1-n\alpha_j}{n+1} \boldsymbol{d}_1^{(it)}
\end{aligned}$$

and for $i = 2, \ldots, n+1$

$$\begin{aligned}
\boldsymbol{d}_i^{(it+1)} &= \boldsymbol{c}^{(it+1)} - \boldsymbol{x}_i^{(it+1)} = \frac{1}{n+1} \left((n+1+\alpha_j) \boldsymbol{c}^{(it)} - \alpha_j \boldsymbol{x}_1^{(it)} \right) - \boldsymbol{x}_i^{(it)} \\
&= \boldsymbol{c}^{(it)} - \boldsymbol{x}_i^{(it)} + \frac{\alpha_j}{n+1} (\boldsymbol{c}^{(it)} - \boldsymbol{x}_1^{(it)}) = \boldsymbol{d}_i^{(it)} + \frac{\alpha_j}{n+1} \boldsymbol{d}_1^{(it)}.
\end{aligned}$$

It is known that $D(S^{(it)}) = \{\boldsymbol{d}_1^{(it)}, \boldsymbol{d}_2^{(it)}, \ldots, \boldsymbol{d}_{n+1}^{(it)}\}$ is a positive basis in $\mathbb{R}^n$, then $B^{(it)} = \{\boldsymbol{d}_1^{(it)}, \boldsymbol{d}_2^{(it)}, \ldots, \boldsymbol{d}_n^{(it)}\}$ forms a basis in $\mathbb{R}^n$ (Theorem 5.2.1). Therefore, $\{\boldsymbol{d}_1^{(it)}, \boldsymbol{d}_2^{(it)}, \ldots, \boldsymbol{d}_n^{(it)}\}$ is a linear independent set. Now, we prove that $B^{(it+1)} = \{\boldsymbol{d}_1^{(it+1)}, \boldsymbol{d}_2^{(it+1)}, \ldots, \boldsymbol{d}_n^{(it+1)}\}$ is also a linear independent set in $\mathbb{R}^n$.

Let $\beta_1, \beta_2, \ldots, \beta_n \in \mathbb{R}$

$$\sum_{i=1}^{n} \beta_i \boldsymbol{d}_i^{(it+1)} = 0 \Leftrightarrow \beta_1 \frac{n+1-n\alpha_j}{n+1} \boldsymbol{d}_1^{(it)} + \sum_{i=2}^{n} \beta_i \left(\boldsymbol{d}_i^{(it)} + \frac{\alpha_j}{n+1} \boldsymbol{d}_1^{(it)} \right) = 0$$

$$\Leftrightarrow \left(\beta_1 \frac{n+1-n\alpha_j}{n+1} + \sum_{i=2}^{n} \beta_i \frac{\alpha_j}{n+1} \right) \boldsymbol{d}_1^{(it)} + \sum_{i=2}^{n} \beta_i \boldsymbol{d}_i^{(it)} = 0 \Rightarrow \begin{cases} \beta_1 \frac{n+1-n\alpha_j}{n+1} + \sum_{i=2}^{n} \beta_i \frac{\alpha_j}{n+1} = 0 \\ \beta_2 = 0 \\ \vdots \\ \beta_n = 0 \end{cases}$$

$$\Rightarrow \beta_i = 0, \quad i = 1, \ldots, n, \quad \text{assuming } \alpha_j \neq 1 + \tfrac{1}{n}.$$

Since $B^{(it+1)} = \{\boldsymbol{d}_1^{(it+1)}, \boldsymbol{d}_2^{(it+1)}, \ldots, \boldsymbol{d}_n^{(it+1)}\}$ is a linear independent set in $\mathbb{R}^n$, $B^{(it+1)}$ is a basis in $\mathbb{R}^n$. Note that

$$\begin{aligned} -\sum_{i=1}^{n} \boldsymbol{d}_i^{(it+1)} &= -\frac{n+1-n\alpha_j}{n+1} \boldsymbol{d}_1^{(it)} - \boldsymbol{d}_2^{(it)} - \frac{\alpha_j}{n+1} \boldsymbol{d}_1^{(it)} - \cdots - \boldsymbol{d}_n^{(it)} - \frac{\alpha_j}{n+1} \boldsymbol{d}_1^{(it)} \\ &= -\frac{n+1-\alpha_j}{n+1} \boldsymbol{d}_1^{(it)} - \boldsymbol{d}_2^{(it)} - \cdots - \boldsymbol{d}_n^{(it)} \\ &= \boldsymbol{d}_{n+1}^{(it)} + \frac{\alpha_j}{n+1} \boldsymbol{d}_1^{(it)} = \boldsymbol{d}_{n+1}^{(it+1)}. \end{aligned}$$

Thus, using the Theorem 5.2.3 again, $\tilde{B}^{(it+1)} = B^{(it+1)} \cup \{-\sum_{i=1}^{n} \boldsymbol{d}_i^{(it+1)}\} = \{\boldsymbol{d}_1^{(it+1)}, \boldsymbol{d}_2^{(it+1)}, \ldots, \boldsymbol{d}_{n+1}^{(it+1)}\} = D(S^{(it+1)})$ forms a positive basis in $\mathbb{R}^n$. If iteration *it* performs a shrink step, assume without loss of generality that $\boldsymbol{x}_{best}^{(it)} = \boldsymbol{x}_1^{(it)}$. Therefore $\boldsymbol{x}_1^{(it+1)} = \boldsymbol{x}_1^{(it)}$ and $\boldsymbol{x}_i^{(it+1)} = \boldsymbol{x}_{best}^{(it)} + \alpha_s(\boldsymbol{x}_i^{(it)} - \boldsymbol{x}_1^{(it)}) = (1-\alpha_s)\boldsymbol{x}_1^{(it)} + \alpha_s \boldsymbol{x}_i^{(it)}, i = 2, \ldots, n+1$. Note that

$$\begin{aligned} \boldsymbol{c}^{(it+1)} &= \frac{1}{n+1} \sum_{i=1}^{n+1} \boldsymbol{x}_i^{(it+1)} = \frac{1}{n+1} \left(\boldsymbol{x}_1^{(it)} + \sum_{i=2}^{n+1} [(1-\alpha_s)\boldsymbol{x}_1^{(it)} + \alpha_s \boldsymbol{x}_i^{(it)}] \right) \\ &= \frac{1}{n+1} \left((n(1-\alpha_s)+1)\boldsymbol{x}_1^{(it)} + \alpha_s \sum_{i=2}^{n+1} \boldsymbol{x}_i^{(it)} + \alpha_s \boldsymbol{x}_1^{(it)} - \alpha_s \boldsymbol{x}_1^{(it)} \right) \\ &= \frac{1}{n+1} \left((n+1-(n+1)\alpha_s)\boldsymbol{x}_1^{(it)} + (n+1)\alpha_s \boldsymbol{c}^{(it)} \right) \\ &= \alpha_s \boldsymbol{c}^{(it)} + (1-\alpha_s)\boldsymbol{x}_1^{(it)}. \end{aligned}$$

Moreover,

$$\begin{aligned} \boldsymbol{d}_1^{(it+1)} &= \boldsymbol{x}_1^{(it+1)} - \boldsymbol{c}^{(it+1)} = \boldsymbol{x}_1^{(it)} - \alpha_s \boldsymbol{c}^{(it)} - (1-\alpha_s)\boldsymbol{x}_1^{(it)} \\ &= \alpha_s \boldsymbol{x}_1^{(it)} - \alpha_s \boldsymbol{c}^{(it)} = \alpha_s \boldsymbol{d}_1^{(it)} \end{aligned}$$

and, for $i = 2, \ldots, n+1$

$$\begin{aligned} \boldsymbol{d}_i^{(it+1)} &= \boldsymbol{x}_i^{(it+1)} - \boldsymbol{c}^{(it+1)} = (1-\alpha_s)\boldsymbol{x}_1^{(it)} + \alpha_s \boldsymbol{x}_i^{(it)} - \alpha_s \boldsymbol{c}^{(it)} - (1-\alpha_s)\boldsymbol{x}_1^{(it)} \\ &= \alpha_s(\boldsymbol{x}_i^{(it)} - \boldsymbol{c}^{(it)}) = \alpha_s \boldsymbol{d}_i^{(it)}. \end{aligned}$$

As we expected, the new directions only change the size. Thus, $D(S^{(it+1)})$ still forms a positive basis in $\mathbb{R}^n$. □

We showed that if S is a simplex in $\mathbb{R}^n$, then its set of directions $D(S)$ forms a positive basis in $\mathbb{R}^n$. Furthermore, we proved that the method suggested preserves the positive basis along the iterations, i.e, if $D(S^{(it)})$ forms a positive basis in $\mathbb{R}^n$, then $D(S^{(it+1)})$ also forms a positive basis in $\mathbb{R}^n$. The last proposition also showed that the step size is decreased when an iteration is unsuccessful.

The variant of the NM method suggested in thesis was used for the McKinnon function (5.5), considering the initial simplex as in described in (5.6), with $tol =$ 1e-04. The algorithm performed 60 iterations and converge to $\boldsymbol{x}^* =$ (9.69654e-05, −5.000e-01), where $f(\boldsymbol{x}^*) =$ -2.5000e-01. This result highlights the importance of the control of the geometry through the cosine measure used in this method.

Theorem 5.4.1. *Let $D(S^{(it)})$ be a positive basis in $\mathbb{R}^n$. Assume that ∇f is Lipschitz continuous (with constant $L > 0$) in an open set containing all the poll points in $P^{(it)} = \{\boldsymbol{x}_i^{(it)} + \alpha_j \boldsymbol{d}_i,\ \boldsymbol{d}_i \in D^{(it)},\ i = 1, \ldots, n+1\}$. Assume that $\alpha^{(it)}$ is the coefficient used in the iteration it, $\alpha^{(it)} \in \{\alpha_r, \alpha_e, \alpha_{oc}, \alpha_{ic}\}$ and that $\mathtt{cm}(D(S^{(it)})) \geq \gamma > 0$. To simplify the notation, suppose that $\boldsymbol{x}_{worst}^{(it)} = \boldsymbol{x}^{(it)}$. Assuming that, at each iteration, $cm(D(S^{(it)})) \geq \max\{||d_worst - d||/||d||, d \in D(S^{(it)})\}$, if $f(\boldsymbol{x}^{(it)}) \leq f(\boldsymbol{x}^{(it)} + \alpha^{(it)}\boldsymbol{d}) + \rho(\|\boldsymbol{d}\|)$, i.e., the iteration it is unsuccessful, then*

$$\|\nabla f(\boldsymbol{x}^{(it)})\| \leq C\left(\frac{L}{2}\alpha^{(it)}\|\boldsymbol{d}\|^2 + \frac{\rho(\|\boldsymbol{d}\|)}{\alpha^{(it)}}\right).$$

Proof. Consider $h : \mathbb{R}^n \setminus \{0\} \to [0,1]$ such that $h(\boldsymbol{v}) = \max\limits_{\boldsymbol{d} \in D(S^{(it)})} \frac{\boldsymbol{v}^T \boldsymbol{d}}{||\boldsymbol{v}||||\boldsymbol{d}||}$.

Given a nonzero vector $\boldsymbol{w} \in \mathbb{R}^n$, $\mathtt{cm}(D(S^{(it)})) = min\{h(\boldsymbol{v}) : \boldsymbol{v} \neq 0\} \leq h(\boldsymbol{w}) = \max\limits_{\boldsymbol{d} \in D(S^{(it)})} \frac{\boldsymbol{w}^T \boldsymbol{d}}{||\boldsymbol{w}||||\boldsymbol{d}||} = \frac{\boldsymbol{w}^T \overline{\boldsymbol{d}}}{||\boldsymbol{w}||||\overline{\boldsymbol{d}}||}$, for some $\overline{\boldsymbol{d}} \in D(S^{(it)})$.

Thus, $0 < \mathtt{cm}(D(S^{(it)})) \leq \frac{\boldsymbol{w}^T \overline{\boldsymbol{d}}}{||\boldsymbol{w}||||\overline{\boldsymbol{d}}||} \Leftrightarrow 0 < \mathtt{cm}(D)||\boldsymbol{w}||||\overline{\boldsymbol{d}}|| \leq \boldsymbol{w}^T \overline{\boldsymbol{d}}$. In particular, for the negative gradient at a given point $\boldsymbol{x}^{(it)}$

$$\mathtt{cm}(D(S^{(it)}))||\nabla f(\boldsymbol{x}^{(it)})||||\overline{\boldsymbol{d}}|| \leq -\nabla f(\boldsymbol{x}^{(it)})^T \overline{\boldsymbol{d}}. \tag{5.10}$$

Since $f(\boldsymbol{x}^{(it)}) \leq f(\boldsymbol{x}^{(it)} + \alpha^{(it)}\boldsymbol{d}) + \rho(\|\boldsymbol{d}\|)$,, from the integral form of the Mean Value Theorem we obtain that

$$0 \leq f(\boldsymbol{x}^{(it)} + \alpha^{(it)}\boldsymbol{d}) - f(\boldsymbol{x}^{(it)}) + \rho(\|\boldsymbol{d}\|) = \int_0^1 \nabla f(\boldsymbol{x}^{(it)} + t\alpha^{(it)}\boldsymbol{d})^T \alpha^{(it)}\boldsymbol{d}\, dt + \rho(\|\boldsymbol{d}^{(it)}\|).$$

By multiplying (5.10) by $\alpha^{(it)}$ and adding it to the above inequality, we get

$$
\begin{aligned}
\mathtt{cm}(D(S^{(it)}))||\nabla f(\boldsymbol{x}^{(it)})||||\overline{\boldsymbol{d}}||\alpha^{(it)} &\leq \int_0^1 \nabla f(\boldsymbol{x}^{(it)}+t\alpha^{(it)}\boldsymbol{d})^T\alpha^{(it)}\boldsymbol{d}\,dt - \nabla f(\boldsymbol{x}^{(it)})^T\alpha^{(it)}\overline{\boldsymbol{d}} + \rho(\|\boldsymbol{d}\|)\\
&= \int_0^1 \left(\nabla f(\boldsymbol{x}^{(it)}+t\alpha^{(it)}\boldsymbol{d}) - \nabla f(\boldsymbol{x}^{(it)})\right)^T\alpha^{(it)}\boldsymbol{d}\,dt - \nabla f(\boldsymbol{x}^{(it)})^T\alpha^{(it)}\overline{\boldsymbol{d}}\\
&\quad + \nabla f(\boldsymbol{x}^{(it)})^T\alpha^{(it)}\boldsymbol{d} + \rho(\|\boldsymbol{d}\|)\\
&\leq \int_0^1 \|\nabla(f(\boldsymbol{x}^{(it)}+t\alpha^{(it)}\boldsymbol{d}) - \nabla f(\boldsymbol{x}^{(it)})\|\,\|\alpha^{(it)}\boldsymbol{d}\|\,dt + \alpha^{(it)}\|\nabla f(\boldsymbol{x}^{(it)})\|\|\boldsymbol{d}-\overline{\boldsymbol{d}}\|\\
&\quad + \rho(\|\boldsymbol{d}\|)\\
&\leq (\alpha^{(it)})^2 L\|\boldsymbol{d}\|^2\int_0^1 t\,dt + \alpha^{(it)}\|\nabla f(\boldsymbol{x}^{(it)})\|\|\boldsymbol{d}-\overline{\boldsymbol{d}}\| + \rho(\|\boldsymbol{d}\|)\\
&= \frac{L}{2}(\alpha^{(it)})^2\|\boldsymbol{d}\|^2 + \alpha^{(it)}\|\nabla f(\boldsymbol{x}^{(it)})\|\|\boldsymbol{d}-\overline{\boldsymbol{d}}\| + \rho(\|\boldsymbol{d}\|),
\end{aligned}
$$

where the third and the fourth inequalities follow from the Cauchy-Schwartz inequality and from the Lipschitz continuity of the gradient of f, respectively. Thus, assumig $\mathtt{cm}(D(S^{(it)})) \geq \frac{\|\boldsymbol{d}-\overline{\boldsymbol{d}}\|}{\|\overline{\boldsymbol{d}}\|}$ we obtain

$$
\|\nabla f(\boldsymbol{x}^{(it)})\| \leq C\left(\frac{L}{2}\alpha^{(it)}\|\boldsymbol{d}\|^2 + \frac{\rho(\|\boldsymbol{d}\|)}{\alpha^{(it)}}\right),
$$

where $C = \mathtt{cm}(D(S^{(it)}))\|\overline{\boldsymbol{d}}\| - \|\boldsymbol{d}-\overline{\boldsymbol{d}}\|$. □

5.5 Numerical results

In this section, we present some numerical experiments of the proposed convergent variant of the NM method on polygonal meshes for solving the two-dimensional Helmholtz's equations with homogeneous Dirichlet boundary conditions (5.1). We compare this method with the classical NM method, which is implemented in MATLAB as the function `fminsearch`. Both methods were used to solve the minimisation problem (5.9), providing the computational boundary condition values to be imposed in the DG method.

The numerical results were obtained considering the same conditions as in Section 3.3 and a polygonal mesh $\mathscr{T}_h$ with $h = 9.34\text{e-}01$ as the one presented in that section. Moreover we consider a set of $R^k = 5$ points on the boundary $\partial\Omega$ for each element T^k with common edge e^k with the computational boundary $\partial\Omega_h$. The results obtained for the DG method combined with NM algorithm with step size control are compared with the classical DG method and with the DG method combined with the classical NM method.

We consider the same initial point (which is a null vector) for the modified NM presented in this thesis and for the function `fminsearch`. Moreover, we present the results considering a tolerance $tol = 1\text{e-}04$ and $tol = 1\text{e-}06$ for the stopping criterion in both methods. We consider that stopping conditions for the NM method with step size control are satisfied when $|f(\boldsymbol{x}_{worst}^{(it+1)}) - f(\boldsymbol{x}_{worst}^{(it)})| < tol$ or $\|\boldsymbol{x}_{worst}^{(it+1)} - \boldsymbol{x}_{worst}^{(it)}\| < tol$ and we consider the forcing function $\rho(t) = 0.01t^2$.

In Table 5.1 we report the error evaluated at a set of P points on the physical boundary $\partial\Omega$ and the number of iterations performed by each method, considering $tol = 1\text{e-}04$. We present global error

obtained with the computational boundary condition Eq. (5.2) given by each method in Table 5.2, considering the same tol value.

Moreover, considering $tol = 1\text{e-}06$, in Tables 5.3 and 5.4 we report the error evaluated at a set of P points on the physical boundary $\partial\Omega$ and the number of iterations performed by each method, and the global error obtained with the computational boundary condition given by each method, respectively.

The results obtained in our simulations by the function `fminseach` and by NM method with step size control for polynomials of degree N, with $N = 1,2,3,4$, both combined with the DG method, show a decrease in the global error and in the error evaluated in the points on the $\partial\Omega$ in relation to the errors obtained with the classical DG method. Moreover, the convergent variant of the NM method suggested in this thesis requires less iterations than the MATLAB function `fminsearch`.

In Appendix A, we include a comparison of the methods presented in this thesis: the classical DG method, the DG-ROD method, the DG-`fminsearch` and the DG-NM method with step size control.

Table 5.1 Error evaluated at a set of P points on the physical boundary $\partial\Omega$: $E_\infty(\partial\Omega) = ||u_h(\boldsymbol{P};\boldsymbol{b}) - g(\boldsymbol{P})||_\infty$ and number of iterations it performed by the method, with $tol = 1\text{e-}04$.

Method	$N=1$		$N=2$		$N=3$		$N=4$	
	$E_\infty(\partial\Omega)$	it	$E_\infty(\partial\Omega)$	it	$E_\infty(\partial\Omega)$	it	$E_\infty(\partial\Omega)$	it
DG	2.66e-02	1	1.21e-01	1	1.54-01	1	1.63e-01	1
DG-$fmin$	2.09e-02	327	9.54e-02	802	8.25e-02	2604	1.01e-01	2292
DG-NM_SSC	2.09e-02	325	9.55e-02	715	9.48e-02	1232	1.01e-01	2235

Table 5.2 Global error: $||u - u_h(\cdot,\boldsymbol{b})||_\infty$, with $tol = 1\text{e-}04$.

Method	$N=1$	$N=2$	$N=3$	$N=4$
DG	8.37e-02	1.24e-01	1.07e-01	1.25e-01
DG-$fmin$	8.35e-02	1.10e-01	6.77e-02	9.74e-02
DG-NM_SSC	8.35e-02	1.10e-01	7.38e-02	9.75e-02

Table 5.3 Error evaluated at a set of P points on the physical boundary $\partial\Omega$: $E_\infty(\partial\Omega) = ||u_h(\boldsymbol{P};\boldsymbol{b}) - g(\boldsymbol{P})||_\infty$ and number of iterations it performed by the method, with $tol = 1\text{e-}06$.

Method	$N=1$		$N=2$		$N=3$		$N=4$	
	$E_\infty(\partial\Omega)$	it	$E_\infty(\partial\Omega)$	it	$E_\infty(\partial\Omega)$	it	$E_\infty(\partial\Omega)$	it
DG	2.66e-02	1	1.21e-01	1	1.54e-01	1	1.63e-01	1
DG-$fmin$	1.92e-02	1743	8.59e-02	2320	8.10e-02	4202	9.54e-02	6647
DG-NM_SSC	1.97e-02	966	8.64e-02	1546	8.10e-02	3736	9.77e-02	3510

Table 5.4 Global error: $||u - u_h(\cdot,\boldsymbol{b})||_\infty$, with $tol = 1\text{e-}06$.

Method	$N=1$	$N=2$	$N=3$	$N=4$
DG	8.37e-02	1.24e-01	1.07e-01	1.25e-01
DG-$fmin$	8.64e-02	1.02e-01	6.78e-02	9.83e-02
DG-NM_SSC	8.50e-02	1.03e-01	6.78e-02	9.65e-02

Chapter 6

Conclusion

Our applications of interest consists in analysing the incidence and reflection of light on the cornea, therefore curved boundary domain arise naturally in our domain of interest. Motivated by this fact, in this thesis, we suggest two approaches to deal with the decrease in accuracy of the discontinuous Galerkin finite element method (DG) in a domain Ω with curved boundary. These approaches was suggested in the context of solving the Helmholtz's equation with Dirichlet boundary conditions. We consider that the physical domain Ω is approximated by a polygonal computational domain Ω_h. The DG method allows to obtain a polynomial solution u_h that approximates the solution u of the original problem. This solution is defined, in each triangle T^k of the mesh considered in Ω_h, by a polynomial u_h^k that satisfies the boundary conditions on $\partial\Omega_h$.

The first method proposed, named DG-ROD, is based on a polynomial reconstruction of the boundary condition imposed on the computational domain Ω_h, where the associated coefficients are determined such that the reconstructions adequately satisfy the boundary condition imposed on the physical domain Ω. In this way, we obtain the polynomial boundary condition π^{*k} that is close to the numerical solution u_h^k but allows to correct the error obtained by the approximation of $\partial\Omega$ by $\partial\Omega_h$. This polynomial reconstruction is based on an iterative method that considers two independent black-boxes: the resolution of the differential equation by the classical DG method (where the boundary conditions are defined on Ω_h) and the ROD reconstruction process on triangles with vertices on the boundary of the physical domain $\partial\Omega$. By analyzing the numerical tests presented, we verified that the DG-ROD method, unlike the classical DG method, allows to obtain high order in domains with curved boundary.

In order to avoid the iterative process inherent in the DG-ROD method, we proposed another alternative to overcome the difficulties in the boundary treatment when dealing with curved boundary domains. This approach consist in solving a minimisation problem to determine the boundary condition to be imposed on the computational domain Ω_h. In this method, the boundary condition values are determined such that the error between the exact solution and the numerical one, both evaluated at a set of point $\boldsymbol{P}$ on the physical boundary Ω_h, is minimised. To solve this unconstrained optimisation problem we used a variant of the Nelder-Mead (NM) method. In the modified NM algorithm proposed, all the transformations of the classical NM method can be performed. Moreover, the geometry of the simplex is controlled by requiring a lower bound for the cosine measure of the possible directions on each iteration.

Both methods suggested in this thesis do not require the generation of curved meshes to adjust the boundary, nor complex nonlinear transformations, which contributes for computational efficiency and simplifies the numerical schemes. Another advantage of these methods is the simplicity of the representation of the boundary: it is not required to know the analytical expression of the boundary (it is sufficient to know a set of points $\boldsymbol{P}$ on the boundary) and no orthogonal projection of the elements is performed.

The treatment of curved boundary domains has been a subject of interest since most real problems take place on arbitrary geometries. Thus, the perspectives of the research following the work of the present thesis include a generalisation of the DG-ROD method to other boundary conditions to $\mathscr{B}(u,\boldsymbol{x}) = \alpha(\boldsymbol{x})u(\boldsymbol{x}) + \beta(\boldsymbol{x})\nabla u(\boldsymbol{x}) \cdot \hat{\boldsymbol{n}} - g(\boldsymbol{x}) = 0, \quad \boldsymbol{x} \in \partial\Omega;$. Moreover, since our domain of interest aims to mimic the cornea and it is composed of collagen fibrils, we pretend to extend the method for curved interfaces, mimicking the interfaces between the collagen fibrils of the corneal stroma.

Another direction of future research, following the results presented in this thesis, is to analyse the convergence of the variant of the Nelder-Mead method and to use this algorithm to solve other optimisation problems.

References

[1] Araújo, A., Barbeiro, S., Bernardes, R., et al. (2022a). A mathematical model for the corneal transparency problem. *J.Math.Industry*, 12.

[2] Araújo, A., Barbeiro, S., and Santos, M. (2022b). On the Helmholtz's equation model for light propagation in the cornea. In *Proceedings of the 2nd International Conference on Image Processing and Vision Engineering - Imaging4OND*, pages 265–268. INSTICC, SciTePress.

[3] Bassi, F. and Rebay, S. (1995). Accurate 2D Euler computations by means of a high order discontinuous finite element method. In Deshpande, S. M., Desai, S. S., and Narasimha, R., editors, *Fourteenth International Conference on Numerical Methods in Fluid Dynamics*, pages 234–240. Springer Berlin Heidelberg.

[4] Bassi, F. and Rebay, S. (1997). High-order accurate discontinuous finite element solution of the 2D Euler equations. *Journal of Computational Physics*, 138:251–285.

[5] Brøndsted, A. (1983). *An introduction to convex polytopes*. Graduate Texts in Mathematics 90, Springer-Verlag.

[6] Cockburn, B. and Solano, M. (2012). Solving Dirichlet boundary-value problems on curved domains by extensions from subdomains. *SIAM Journal on Scientific Computing*, 34:497–519.

[7] Conn, A. R., Scheinberg, K., and Vicente, L. N. (2009). *Introduction to Derivative-Free Optimization*. SIAM, Philadelphia, PA, USA.

[8] Costa, R., Clain, S., Loubère, R., and Machado, G. J. (2018). Very high-order accurate finite volume scheme on curved boundaries for the two-dimensional steady-state convection–diffusion equation with Dirichlet condition. *Applied Mathematical Modelling*, 54:752–767.

[9] Doutch, J., Quantock, A. J., Smith, V. A., and Meek, K. M. (2008). Light transmission in the human cornea as a function of position across the ocular surface: Theoretical and experimental aspects. *Biophysical Journal*, 95:5092–5099.

[10] Evans, L. C. (1998). *Partial differential equations*. American Mathematical Society Providence, second edition.

[11] Farrell, R. A., Freund, D. E., and Mccaly, R. L. (1990). Research on corneal structure. *Johns Hopkins APL Technical Digest*, 11(1-2):191–199.

[12] Geuzaine, C. and Remacle, J.-F. (2009). Gmsh: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities. *International Journal for Numerical Methods in Engineering*, 79(11):1309–1331.

[13] Ghalati, M. K. (2017). *Numerical Analysis And Simulation Of Discontinuous Galerkin Method For Time-Domain Maxwell's Equations*. PhD thesis, Universidade de Coimbra, Departamento de Matemática da Faculdade de Ciências e Tecnologia.

[14] Givens, G. H. and Hoeting, J. A. (2013). *Computational Statistics*. John Wiley & Sons, second edition.

[15] Hesthaven, J. and Warburton, T. (2008). *Nodal Discontinuous Galerkin Methods: Algorithms, Analysis, an Applications*. Springer-Verlag, New York.

[16] Jin, J. (2010). *Theory and Computation of Electromagnetic Fields*. Wiley–IEEE Press, first edition.

[17] Krivodonova, L. and Berger, M. (2006). High-order accurate implementation of solid wall boundary conditions in curved geometries. *Journal of Computational Physics*, 211:492–512.

[18] Lagarias, J. C., J. A. Reeds, M. H. W., and Wright, P. E. (1998). Convergence Properties of the Nelder-Mead Simplex Method in Low Dimensions. *SIAM Journal of Optimization*, 9(1):112–147.

[19] Lagarias, J. C., Poonen, B., and Wright, M. H. (2012). Convergence of the Restricted Nelder–Mead Algorithm in Two Dimensions. *SIAM Journal on Optimization*, 22(2):501–532.

[20] McKinnon, K. (1998). Convergence of the Nelder-Mead Simplex method to a nonstationary point. *SIAM Journal of Optimization*, 9(1):148–158.

[21] Meek, K. M. and Knupp, C. (2015). Corneal structure and transparency. *Progres in Retinal and Eye Research*, 49:1–16.

[22] Moiola, A. (2021). Scattering of time-harmonic acoustic waves: Helmholtz equation, boundary integral equations and BEM. Lecture notes for the "Advanced numerical methods for PDEs" class, University of Pavia, Department of Mathematics.

[23] Nelder, J. A. and Mead, R. (1965). A Simplex Method for Function Minimization. *The Computer Journal*, 7(4):308–313.

[24] Price, C., Coope, I., and Byatt, D. (2002). A Convergent Variant of the Nelder–Mead Algorithm. *Journal of Optimization Theory and Applications*, 113(1):5–19.

[25] Reed, W. H. and Hill, T. (1973). Triangular mesh methods for the neutron transport equation. *Los Alamos Report LA-UR-73-479*.

[26] Regis, R. (2016). On the properties of positive spanning sets and positive bases. *Optimization and Engineering*, 17(1):229–262.

[27] Stocker, P. (2017). Plane wave-based approximation methods for the helmholtz equation. Master's thesis, Universität Wien.

[28] Strang, G. and Berger, A. E. (1973). The change in solution due to change in domain. In *Partial differential equations*, pages 199–205.

[29] Warburton, T. (2006). An explicit construction for interpolation nodes on the simplex. *J. Engineering Math.*, 56:247–262.

[30] Yin, J., Xu, L., Xie, P., Zhu, L., Huang, S., Liu, H., Yang, Z., and Li, B. (2021). A curved boundary treatment for discontinuous Galerkin method applied to Euler equations on triangular and tetrahedral grids. *Computer Physics Communications*, 258:107549.

[31] Yosida, K. (1995). *Functional Analysis*. Springer-Verlag, 6th edition.

[32] Zhang, X. (2016). A curved boundary treatment for discontinuous Galerkin schemes solving time dependent problems. *Journal of Computational Physics*, 308:153–170.

Appendix A

Comparison of methods

In Table A.1, we report the global error $E_\infty = ||u - u_h(\cdot, \boldsymbol{b})||_\infty$ evaluated at the grid points of a polygonal mesh $\mathscr{T}_h$ with $h =$9.34e-01, the error evaluated at a set of P points on the physical boundary and the number of iterations it performed by each method, with $tol =$1e-04 and $tol =$1e-06. We compare the classical DG method, the DG-$fminsearch$ method (DG-$fmin$), the DG-NM with step size control (DG-NM_SSC) and the DG-ROD method. We consider a null vector as initial point for all methods.

Table A.1 Comparison of the methods considering $R^k = 5$ and a polygonal mesh $\mathscr{T}_h$ with $h =$9.34e-01.

tol	N	Error	DG	DG-$fmin$	DG-NM_SSC	DG-ROD
$1e-04$	1	E_∞	8.37e-02	8.35e-02	8.35e-02	8.37e-02
		$E_\infty(\partial\Omega)$	2.66e-02	2.09e-02	2.09e-02	2.66e-02
		it	1	327	325	2
	2	E_∞	1.24e-01	1.10e-01	1.10e-01	7.90e-02
		$E_\infty(\partial\Omega)$	1.21e-01	9.54e-02	9.55e-02	6.44e-02
		it	1	802	715	4
	3	E_∞	1.07e-01	6.77e-02	7.38e-02	1.82e-03
		$E_\infty(\partial\Omega)$	1.54e-01	8.25e-02	9.48e-02	1.19e-03
		it	1	2604	1232	12
	4	E_∞	1.25e-01	9.74e-02	9.75e-02	1.29e-03
		$E_\infty(\partial\Omega)$	1.63e-01	1.01e-01	1.01e-01	9.95e-04
		it	1	2292	2235	38
$1e-06$	1	E_∞	8.37e-02	8.64e-02	8.50e-02	8.37e-02
		$E_\infty(\partial\Omega)$	2.66e-02	1.92e-02	1.97e-02	2.66e-02
		it	1	1743	966	2
	2	E_∞	1.24e-01	1.02e-01	1.03e-01	7.90e-02
		$E_\infty(\partial\Omega)$	1.21e-01	8.59e-02	8.64e-02	6.43e-02
		it	1	2320	1546	7
	3	E_∞	1.07e-01	6.78e-02	6.78e-02	1.88e-03
		$E_\infty(\partial\Omega)$	1.54e-01	8.10e-02	8.10e-02	1.24e-03
		it	1	4202	3736	21
	4	E_∞	1.25e-01	9.83e-02	9.65e-02	1.32e-03
		$E_\infty(\partial\Omega)$	1.63e-01	9.54e-02	9.77e-02	1.12e-05
		it	1	6647	3510	77