Parallel $\mathscr{H}$-Matrices accelerated isogeometric boundary element method implementation applied to acoustics internal and external problems

Álvaro Campos Ferreira

Brasília, 27 de novembro de 2020

UNIVERSIDADE DE BRASÍLIA

FACULDADE DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA MECÂNICA

UNIVERSIDADE DE BRASÍLIA
Faculdade de Tecnologia
Departamento de Engenharia Mecânica

TESE DE DOUTORADO

Parallel $\mathscr{H}$-Matrices accelerated isogeometric boundary element method implementation applied to acoustics internal and external problems

Por

Álvaro Campos Ferreira

Relatório submetido como requisito parcial para obtenção
do grau de Doutor em Ciências Mecânicas

Banca Examinadora

Professor Dr Éder Lima de Albuquerque (orientador) _____

Professor Dr Marcus Vinicius Girão de Morais (coorientador) _____

Professor Dr Gabriel de Oliveira Ribeiro (membro externo) _____

Professor Dr Paulo Sollero (membro externo) _____

Professora Dra Marcela Rodrigues Machado (membro interno) _____

Brasília 27 de novembro de 2020

Álvaro Campos Ferreira

Parallel $\mathscr{H}$-Matrices accelerated isogeometric boundary element method implementation applied to acoustics internal and external problems / Álvaro Campos Ferreira. – Brasília, 27 de novembro de 2020-

130 p. : il. (algumas color.) ; 30 cm.

Supervisor: Professor Dr Éder Lima de Albuquerque (orientador)

Tese de Doutorado – Universidade de Brasília - UnB
Faculdade Tecnologia , 27 de novembro de 2020.

1. boundary element method. 2. acoustics. I. Professor Dr Éder Lima de Albuquerque (orientador). II. Universidade de Brasília. III. Faculdade de Tecnologia. IV. Parallel $\mathscr{H}$-Matrices accelerated isogeometric boundary element method implementation applied to acoustics internal and external problems

CDU 02:141:005.6

Acknowledgements

Gostaria de agradecer minha família e amigos cuja companhia foi essencial para finalizar o trabalho. Agradeço a minha irmã, Clarissa. Te amo, maninha. A minha companheira Adriana. Meu primo, Francisco. Meus amigos Túlio, Natália, Roberto e Marcos. Agradeço a Bárbara, amiga cientista. E meus pais.

Agradeço meu amigo Dante, que heroicamente montou e cedeu o computador que rodou todas as simulações neste trabalho, o cinza. Assim como o Filipe e eu passamos tardes implementando juntos, o cinza e eu passamos muitas noites juntos processando! Eu te garanto que ele teve pouco tempo de trégua! Agradeço Lucas Campos, atualmente professor da UFES, mas eterno colega de doutorado. Os colegas do Grupo de Dinâmica de Sistemas, Gino, Tiago, Laís e principalmente o nosso técnico Filipe, cujo suporte foi essencial para o trabalho. Ao grupo de estudos do método dos elementos de contorno da UnB, o UnBEM. Aos meus psicólogos ao longo desse doutorado, especialmente Ana Paula e Robson. Bloqueio de escrita é real mas não instransponível. Obrigado.

Resumo

Uma implementação paralela da formulação do método dos elementos de contorno isogeométrico acelerada pelas matrizes hierárquicas é apresentada neste trabalho. A implementação está disponível online em github.com/alvarocafe/BEM_base e contém testes baseados em problemas de acústica interna e externa para os quais soluções analíticas estão disponíveis.

A formulação descrita nesse trabalho utiliza curvas de Bézier obtidas de NURBS através de um procedimento de extração de Bézier. Arquivos de CAD com especificações abertas como IGES em geral utilizam curvas NURBS que podem ser utilizadas para a extração, mas um editor de NURBS em Julia é apresentado para construir os modelos utilizados nesse trabalho. É possível também obter os pontos de controle, pesos e ordem de curvas específicas NURBS e obter a representação como curvas de Bézier sem prejuizo em precisão ou continuidade. Uma vez que o domínio é representado como um retalho de curvas ou superfícies de Bézier, esse retalho compõe o contorno da representação direta do método dos elementos de contorno. O domínio consiste no volume apontado pelo vetor oposto ao vetor unitário normal no contorno. Cada curva de Bézier pode ser considerada como um elemento de contorno, com o cuidado de não se utilizar os pontos de controle como os pontos de colocação, pois eles podem e muitas vezes não se encontram no contorno, e sim construir pontos posicionados de forma conveniente na curva. Sendo as condições de contorno aplicadas a elementos individuais, o resultado é um sistema linear $N \times N$, sendo N o número de curvas de Bézier que compõe o contorno. A montagem do sistema é realizada através de matrizes hierárquicas por interpolação utilizando polinômios de Lagrange. Isso significa que as as matrizes de influência serão representadas como matrizes de baixo rank, especificamente, como um produto matricial de outras pequenas matrizes, chamadas blocos. Essa representação é conveniente pois a memória necessária para armazenar uma matriz é reduzida, de acordo com o rank dessa matriz. Utilizando esse método, a matriz de influência completa nunca é armazenada, uma vez que o sistema linear é resolvido utilizando o método dos mínimos resíduos generalizados. Esse procedimento permite que problemas maiores sejam resolvidos para uma mesma configuração de hardware.

A implementação é utilizada para resolver um problema inverso usando algoritmos genéticos para obter a configuração de um modelo axissimétrico tridimensional a partir da

informação do fluxo acústico em pontos discretos. A otimização foi utilizada para inferir a configuração de um trato vocal utilizando apenas 20 pontos de informação do fluxo acústico em uma linha reta entre a glote e a boca.

Um levitador acústico não resonante foi implementado experimentalmente e numericamente e a resposta acústica é comparada com imagens obtidas pelo método de Schlirien com boa concordância. O levitador utilizado é baseado no projeto TinyLev, que usa 72 transdutores ultrassônicos ao invés de falantes de Langevin para produzir a levitação. O levitador é modelado utilizando o BEM e uma bancada experimental é apresentada para providenciar imagens de Schlirien da onda acústica estacionária.

Palavras-chaves: acústica. método dos elementos de contorno. matrizes hierárquicas. métodos numéricos.

Abstract

A parallel implementation of the hierarchical matrices accelerated isogeometric boundary element method formulation is presented in this work. The implementation is available online in github.com/alvarocafe/BEM_base and contains tests based on internal and acoustic problems analytical solutions.

The formulation described in this work utilises Bézier curves obtained from NURBS through a Bézier extraction procedure. CAD files with open specifications such as IGES uses NURBS curves from which Bézier patches may be extracted, but a NURBS editor in Julia is presented to build the models used in this work. It's possible to obtain control points, weights and curve degrees such that there is no loss in precision or continuity of the curve. Once the domain is represented as a Bézier patch, this patch is used as the boundary of the direct boundary element method. The volume in the direction the unit normal vector to the boundary is the domain of interest. Each Bézier curve may be considered a boundary element, with the care to no use control points as collocation points, as they may reside outside of the domain, but to position the points conveniently on the curve. As the boundary conditions are applied on individual elements results in a $N \times N$ linear system, for N elements. The system is built using hierarchical matrices using interpolation by Lagrange polynomials. This means that the influence matrices are represented as low-rank, specifically as a matrix product of smaller matrices, called blocks. This representation is convenient as the memory necessary to store the matrix is reduced, accordingly to the its rank. Using this procedure, the full influence matrix is never stored, as the linear system is solved using the generalized minimal residual method. Such procedure allows larger problems to be solved for a given hardware configuration.

The implementation was used to solve an inverse method optimization using genetic algorithms to obtain the geometric configuration of a three-dimensional axissymetrical model using only acoustic information. The optimization was used to infer the configuration of a vocal tract using only 20 points of acoustic flux information, displayed in a straight line from the glottis to the mouth.

A non-resonant acoustic levitator model was also implemented and the resulting acoustic response is compared to Schlirien imaging showing good accordance. The levitator is based on the TinyLev project, which uses 72 ultrassonic transducers opposed to Langevin horns to produce acoustic levitation. The levitator is modeled using the BEM

and an experimental bench is presented to provide Schlirien imaging of the standing acoustic wave.

Key-words: acoustics. boundary element method. hierarchical matrices. numerical methods.

Summary

Figures

Tables

List of abbreviations and acronyms

ACA	Adaptive Cross-Approximation
BEM	Boundary Element Method
FEM	Finite Element Method
MRI	Magnetic Resonance Imaging
CAD	Computer-Aided Design
IGA	Isogeometric Analysis
NURBS	Non-uniform Rational B-Spline
GMRES	Generalized Minimal Residual method
BIE	Boundary Integral Equation
CBIE	Conventional Boundary Integral Equation
HBIE	Hypersingular Boundary Integral Equation
TM	Transfer matrix Method

List of symbols

$A, a, B, b, ...$	Letters in italic are scalars
$\mathbf{A}, \mathbf{a}, \mathbf{B}, \mathbf{b}, ...$	Letters in bold are vectors
Ω	Physical domain
Γ	Boundary of domain Ω
$\frac{d}{dt}$	Derivative with respect to t
$\frac{\partial}{\partial n}$	Partial derivative with respect to n
ϕ	Velocity potential
∇	Laplace operator
ρ	Specific mass
θ	Angle
ω	Angular frequency
λ	Wavelenght
δ	Dirac's delta
ξ	Parametric variable
β	Coupling constant
ϵ	Relative error

‘

1 INTRODUCTION

The study of acoustic scattering is a major area of active research as it relates to basic human communication and comfort. Acoustic models can also be used in many engineering applications, such as non-destructive control on structures. The design of reactive or dissipative mufflers in the admission or exhaustion of internal combustion engines is a concern in the automotive industry, for example. In petrochemical applications, the noise control is a necessity for the reduction of acoustic effects generated by the machinery or the flow inside the duct itself. The study of the vocal tract bounces in the domain of phonoaudiology, voice synthesis and in the teaching/coaching of lyrical singers and any professional that uses speech as a way to convey information. Passive noise control (PNC) is a strategy for the reduction in amplitude of unwanted noise using sound absorbing materials to dissipate the acoustic energy. The disposition of the sound absorbing material is subject to an array of restrictions and the acoustic properties of the problem are very important to obtain the best possible solution.

Many solutions are proposed to acoustic scattering problems, each presenting different strengths and weaknesses. Analytical solutions can be obtained by directly solving the Helmholtz equation, which governs the acoustic scattering phenomena. These analytical solutions are often difficult to obtain and are limited to special geometries (DHANDOLE; MODAK, 2007). Prototyping is expensive and time consuming, whereas numerical solutions are widely available, cheaper and softwares are getting increasingly easier to operate. Developing new numerical tools is essential to predict the acoustic field thorough the audible frequency range of new complex geometry scatterers (ESNORFF, 2008). Due to Moore's law, processing and storage capacity in computers have been doubling up each year since the 1970's, making computational solutions cheaper and faster than ever. Amongst the wide variety of numerical methods that can solve acoustic scattering problems, Finite Element Method (FEM) and Boundary Element Method (BEM) are the most prominent methods used in laboratories all around the world.

FEM is a powerful tool, but it demands the whole domain to be meshed. This difficulties the solving of infinite domain problems, once the best solution would require an infinite mesh around the scatterer. Of course, there are numerical ways to get around this limitation, but they often require complex meshing techniques to achieve a compatible model to infinite domain scattering. Kopuz et al (1995) present a comprehensive comparison between FEM and BEM in an interior acoustics problem.

The BEM is a powerful numerical method for solving acoustic scattering problems governed by the Helmholtz equation. BEM models tend to be competitive mainly due to ease of discretization. Using the BEM, only the boundary of the model needs to be discretized, effectively eliminating one dimension of the problem. In acoustic scattering problems, as with any problem with infinite domains, the BEM presents an even more appealing facet: there is no need to discretize the whole domain. The mathematical internal structure of the method already satisfies the Sommerfeld radiation condition, scatterers need only to have their boundary meshed, rather than the complete domain (WROBEL, 2001). Another advantage of the BEM is the evaluation of quantities outside of the boundary of the problem. Every quantity is evaluated in the boundary of the problem in the BEM, and quantities can be evaluated outside the boundary as post-processing and can be located anywhere inside the domain without any restrictions. In external problems, this means that any point out of the boundary of the scatterer can be evaluated.

1.1 Numerical methods and computation

Humans have been carrying out computations for many generations. There are bone markings for counting which are at least 10 thousand year old. The babylonian method for determining square roots is at least 6 thousand years old, they also did computations to approximate pi and determine Jupiters movement through the night sky by calculating the area under a time-velocity graph (OSSENDRIJVER, 2016).

In the pre-eletronic era, the methods for doing calculations have been the same for thousands of years. Kepler computed the movement of the planets with astonishing precision using no more than his mind, pen and paper. Abacus existed and other abacus-like tables, which were used for merchant trades and banking using roman and later arabic numerals, which ere historically invented in India. Pascal introduced a mechanical calculator called *the Pascalline*, which was not popularized by its high cost and the fact that it didn't perform the calculations faster than a person with pen and paper. Nevertheless, it inspired Leibniz to pioneer in the field of mechanical calculators with his *Stepped Reckoner*, which allowed to perform all four elementary operations on decimal characters. This type of mechanical calculators were expansive, but could greatly increase the speed of calculation and so it was produced and used in state agencies, business and military applications up until the rise of the electronic calculators and computers were introduced.

Charles Babbage later improved upon Leibinz concept and designed the *Difference Engine* which could perform operations on a step by step way which could solve problems and calculate analytical functions, namely, polynomial functions. Babbage later designed but never constructed his *Analytical Engine*, which would be considered a general purpose computer in today's nomenclature. This was a fully programable general purpose computer whose first program was written by mathematician Ada Lovelace (KING, 1843)

on her notes as translator of a sketch of the analytucal engine written by italian engineer Luigi Menabrea. Her notes expands on the descriptions given by Menabrea, which was mainly a mechanical description of the machine. Lovelace introduced abstractions to the machine and showed how programs would be loaded by using punched tapes, as was then used by fabric machinery to produce elaborate broidery. In this work, a program to calculate Bernoulli numbers on the analytical engine is shown, which came to be the first computer program to be published.

Even with this advances, long computations are laborious and even skilled mathematicians are capable of doing not so many computations in a day. They produced formulas which computed with precision very specific cases, which allow for simplifying hyphoteses. Ballistics and accounting applications required large number of computations to be performed, and mechanical calculators had limitations.

Engineers are eager to solve real life problems, which are often complex and must be specified mathematically. The resulting method for solving these problems involve storing large amount of digits as well as performing operations on them. By the end of the 1800's, many computation machine had already existed, most of them of mechanic nature. The use of electricity for computations started by then and by 1940's there were large computers for solving specific problems such as decrypting messages or accounting for census. These computers were very specialized and generally hard wired for the task it solved. After World War II, much of different computer architectures developed independently across Europe and the United States converged into the now standard Von Neumann architecture, composed of a processor, arithmetic and store units. Nowadays, these are called by the general terms Central Processing Unit (CPU) and memory. This adoption led to the development of general purpouse electronic computers. By the 1950's, electronic general purpouse computers existed, but they could only run one program at a time, which they often did 24/7 in a process called batch processing. Each program had the full capacity of the computer's speed and memory while it is running and each program is run until it completes sequentially. This worked well as the computers were very slow by today's standard and each program could run for several days or weeks. The development of the first commercial high level compiled language began in 1953 by IBM and is called FORTRAN (BACKUS, 1998). The name comes from FORmula TRANslation and its main use is for numerical computation. FORTRAN is still in use today for this very purpose and many legacy FORTRAN code is available.

By the 1960's, computers got fast enough that a time-sharing concept was created. Time-sharing meant that each program could access some of the computer's processing and memory, allowing for multiple programs to run at once. This also meant that many users could access the computer through teletypes or terminals, giving the impression of harnessing the full computer's power at any time. A precursor in this field was professor John Kemeny of University of Darthmouth, who co-created the BASIC programmign language and the Darthmouth Time-Sharing System (DTSS) in 1963. Up until them,

computer-time was very expensive and computer operation was limited to high level technical corporate or governmental jobs. At Darthmouth's campus, undergraduate students could use terminal's connected to a central mainframe computer to write their homework, create programs and run them at the same time 30 other students did the same. This software has an important role in computer history as it was the first operating system of home computers.

The evolution of hardware was very fast at this decade as NASA invested heavily on embedded computers for the Apolo mission. The invention of the solid state transistor allowed for much more reliable and low-cost computer. In 1969, Ken Thompson and Dennis Ritchie created Unix, a time-sharing operating system at Bell Labs. This operating system attracted a lot of attention, specially as it was mostly written in a high-level language. Some years later, they rewrote most of Unix in a language they created called C. This made the operating system completely portable to any machine that could compile C, which was very unusual and desired. At that time, most operating systems were written in Assembly language, which is fast but very difficult for human beings to understand.

In 1975, an US based company called MITS released the Altair 8800, the first home computer. At first, there was not much you could do with it, but, at an edition of a BASIC implementation magazine, which was especialized in detailing the porting of BASIC to different hardware, it was described how porting BASIC to this architecture could evoke a powerful home machine. Steve Wozniak, a young engineer working at Intel built his own home computer and called it Apple I. Its design was freely distributed in computer enthusiast meetings called "Home Brew". From this environment, home computing was born and from then on, mainframe computations and large computers became a niched activity, reserved for large-scale computations typical of large companies and the government. Engineering software which was initially developed for large mainframes was quickly brought to home computers.

Nowadays, computing has become an ubiquitous facet of life and a necessary activity in engineering. Most computers are still based on the x86 architecture although 32 and 64 bits machines are still common.

Most of the numerical computations done while solving the numerical problems of physics and engineering are carried out using double-precision 64 bits floating-point numbers. This representation consists of 1 sign, 11 exponent and 53 significant precision bits, which amount to 64 bits, which must be stored in memory. Many floating-point number operations can be performed by a modern CPU. The operations may be parallelized to many cores or even graphics cards, when the operations are indepependent of each other. The number of floating-point operations per second (FLOPS) is the main processing speed paramenter in supercomputers.

1.1.1 Parallel computing

The first supercomputers were mainly not very different from other computers of the era, mainly using a Von Neumann architecture. But Cray started a trend of using many off the shelf components to create multi-core processors which could make many computations at the same time. This architecture and computing strategy came to be known as parallel computing. Even as with many processors there was a lot more computing power available, parallel algorithms must be used to harness all of this power. This style of computing wasn't common place and usually the data is treated sequentially, with one step taken at a time, working with the same data, what is called sequential programs.

Parallel algorithms are not suited to all kinds of problems. In general, a problem can be performed in a parallel manner if there are tasks that can be performed simultaneously. For example, tossing coins and counting the amount of heads and tails can be made with a single coin and observer or many.

Memory access is also a limiting factor for computer processing. Data handling may reduce the performance of a numerical algorithm just as strongly as data processing. Processors got faster than traditional memory hardware and as a result, very fast and expensive memory blocks are built in GPUs and CPUs called *memory caches*. This type of memory is very fast and allows the processor to use all of its speed and clock frequency. Another performance bottleneck in data handling is using swap memory, which is physically written on the hard drive. This type of allocation is done when there's more memory required by the program than available RAM.

1.2 Isogeometric analysis

Regular BEM formulations use Lagrangian shape functions to approximate the boundary geometry of the problem and unknown functional variables. This formulation results in boundary meshes which can be obtained in meshing programs similar to the ones used to obtain FEM meshes. Computer-aided design (CAD) is based on a different approach, with most of the softwares opting for a spline based method. Non-uniform rational B-spline is the most used mathematical model for generating and representing surfaces in CAD softwares. This type of model is based in control points which determine the shape of the spline. A common methodology when generating a mesh for a FEM or BEM model is to describe the geometry of the problems using a CAD software and then use a meshing software to generate the required mesh.

Integration of computer-aided design (CAD) models with numerical methods is an active area in the computational engineering research. Spline-based CAD geometries can be used directly in numerical analysis, using the functions which describe the boundary of the model in the numerical method. This enables a strong interaction between CAD

softwares and numerical methods, reducing the meshing procedure and saving time and computational cost. The idea of using splines in BEM was already proposed and implemented by Cabral, Wrobel e Brebbia (1990). A new element formulation based on uniform cubic B-splines in a BEM formulation was proposed for problems governed by Laplace's equation. Use of spline functions can provide higher degrees of continuity along the geometric boundary of the regions. This allows for a representation of the geometrical shape that is generally much better than other interpolation functions. The results also suggest that, for the same level of accuracy, one can generally use far less B-splines elements than other kinds of elements (CABRAL; WROBEL; BREBBIA, 1991).

An Isogeometric Analysis using BEM (IGABEM) for acoustics was developed by Simpson et al. (2014) using T-splines. Exact geometrical representation of the IGABEM models yields superior accuracy compared to conventional Lagrangian discretization models. The method also uses the exact CAD geometry for the analysis and no mesh generation processes are needed. Several interior and exterior acoustic problems are presented, including an acoustic probe with complex geometry.

Peake, Trevelyan e Coates (2013) presented two formulations of isogeometric boundary element methods for two-dimensional Helmholtz problems. The formulation used NURBS functions to describe the geometry and the approximation of the potential function over the boundary of acoustic scatterers. This resulted in an increase accuracy of the method, as the boundary of the scatterer was better described by the NURBS function. Also, the NURBS functions provide the analytical geometry inherently and the geometry provided by CAD software can be analyzed with little or no need for meshing. An extended isogeometric boundary element method (eXtended IGABEM - XIBEM) formulation for three-dimensional acoustic medium-wave acoustic scattering problems are presented by Peake, Trevelyan e Coates (2015). NURBS functions are decomposed into its Bezier patches to represent the geometries of three-dimensional scatteres. This procedure is primarily motivated as it provides a set of patches that can be easily implemented into existing BEM codes that use Lagrangian elements, allowing a faster integration between CAD geometries and BEM analysis. A linear partition of unity expansion of plane waves is introduced to reduce the number of degrees of freedom needed for certain accuracy. Functions that describe the geometry of the problem are multiplied by families of plane waves to approximate the potential over the surface of the scatterer. It is shown that the XIBEM formulation requires far fewer degrees of freedom to achieve engineering accuracy for the same scattering problem.

1.3 Fast BEM

Historically, the BEM has not obtained much popularity for a very simple reason: it generates fully populated non-symmetric matrices. The linear system which describes

the problem can be easily solved using traditional methods, but the computational cost of storing the matrices is of order $O(N^2)$ and it needs $O(N^3)$ operations to solve the linear system by direct methods and $O(N^2)$ for iterative methods. It is not competitive with other numerical methods in its most basic form. For this reason, BEM models have been limited to a few thousands of degrees of freedom for almost twenty years. Some acceleration methods have been proposed to circumvent this problem.

The fast multi-pole method was the first one to be fully formulated, but its implementation is complex and laborious (LIU; NISHIMURA, 2006). There are many BEM formulations that use the fast multi-pole method to improve the efficiency of the models, but inflexibility is a major concern when considering improvements and variations on the formulation.

Large dense matrices resulting from integral equations have no explicit structure in general. It is possible, however, to find a permutation so that the permuted rows and columns contain blocks of low rank matrices (KURZ; RAIN; RJASANOW, 2002). This means that it is possible to approximate each of these blocks by low rank matrices using some linear algebraic operation. The adaptive cross-approximation (ACA) method is designed to exploit this characteristic to reduce the computational cost of the BEM. To find a suitable permutation, a cluster tree is constructed by recursively partitioning the collocation points according to some geometrical criterions. Low-rank approximations will be assigned to far interactions, while near interactions are described by matrices with higher rank (ROGUS, 2008). This is done by determining a hierarchical approximation of the large dense matrix which arises from the integral equations. Due to the nature of the integral equations of the Green's functions in the BEM formulation, the singularities are close to the diagonal of the matrix. Kurz and Rain (2002) presented an ACA BEM formulation to electromagnetic dynamics which shows great performance advantage over traditional BEM while maintaining engineering accuracy. The main advantage of the formulation proposed is that only original entries of the system matrix are used for the approximation, thus allowing the application into already-developed procedures for the generation of the BEM matrices after some minor modifications. Kurz et al (2007) presented a BEM formulation for electromagnetic problems accelerated by ACA and symmetry exploitation. The formulation used hierarchical matrices clustering to approximate the system matrix by a combination of lower rank matrices. The procedure to determine the approximated matrix is described and numerical examples are given, including an example with complex geometry from an industry application.

Brancati et al (2012) presented a BEM formulation accelerated by the ACA using an iterative solver for three-dimensional Helmholtz problems. The assembly time of the linear system of equations is accelerated by calculating only a few entries of the original matrix. The approximation is obtained by dividing the whole matrix into two blocks, the low rank block and the full rank block. This division is based on the size and distance between a group of collocation points and a group of boundary elements. A full rank block

is not approximated, it is represented entirely. Low rank blocks permit an approximation where only a few entries of the original block are required to represent the entire block. This approximation is based on an hierarchical procedure and the prescribed accuracy is set to determine how many entries of the original block are going to be used to obtain the low rank block. The process leading to the subdivision in sub-blocks and the classification into low and full rank blocks is based on a preliminary hierarchical partition of the matrix index set. Integrals of contiguous elements are almost identical, due to a single collocation point. The matrix is subdivided until a block allows the approximated representation. Mallardo et al (2012) compared experimental results with a BEM model accelerated with ACA of a three-dimensional Helmholtz problem in an aircraft cabin. Numerical and experimental results showed good agreement.

An IGABEM algorithm for elasticity problems accelerated by ACA is proposed by Marussig et al (2015). A strategy for the bisection of the domain described by Non-Uniform B-Splines (NURBS) functions is presented to apply the hierarchical matrices concept. The complete system matrix of equations is subdivided in low rank (far field) and full rank (near field) sub matrices, which are approximated using ACA. The geometry bisection uses the convex hull property of NURBS patches with respect to their control points.

Development in the University of Brasília (UnB) of the isogeometric analysis has produced a formulation of the IGABEM accelerated by the Adaptive Cross-Approximation (ACA+) (CAMPOS; ALBUQUERQUE; WROBEL, 2017). This formulation was initially implemented in C++, but has since been ported to Julia to benefit from its high level syntax and graphic libraries. The process of porting this implementation, revealed some drawbacks of the formulation using the ACA+, which led to the adoption of other strategies to the approximate the influence matrices.

1.4 Goals

This work has the main goal of developing and implementing an isogeometric boundary element method accelerated by hierarchical matrices for the Helmholtz equation. This implementation will be made available as free software under the GPL license[1]. The choice of open sourcing the implementation was made to ease the efforts of maintaning and introducing state of the art methods to engineering students. The readily availlability of the source code allows users to fully comprehend the techniques used to solve the problems, which is not possible with conventional proprietary solvers. The user of proprietary software cannot know the inner workings of it and has to rely on the documentation supplied. Aditionally, the user of a proprietary software cannot make modifications, add new funcionality or adjust the software to their particular use.

[1] now available at github.com/alvarocafe/BB

Other goals for this work are to:

- Solve an inverse problem to determine the geometry if a vocal tract from its acoustic response.
- Perform experimental and numerical analysis of the acoustic levitation phenomenom.
- Set up a Schlirien apparatus bench for acoustic wave imaging.

1.5 Chapter organization

This work is divided in 8 chapters, starting with this introductory one.

Chapter 2 establishes the basics of acoustics, the wave equation and solutions for commom cases such as planewaves and monopoles.

Chapter 3 presents the mathematical formulation for the BEM. The formulation is kept simple and concise for ease of understanding.

Chapter 4 is dedicated to the adaptive cross-approximation (ACA), interpolation by Lagrange polynomials and hierarchical clustering of matrices. This is the main acceleration technique used in this thesis, and the formulation is presented together with the algorithms used for the implementation.

Chapter 5 presents the isogeometric analysis in BEM (IGABEM) and shows the formulation used in the implementation.

Chapter 6 deals with the numerical implementation of the aforementioned formulations in the Julia programming language [2].

Chapter 7 is reserved to showcase numerical results obtained using the algorithms implemented. The first case is the solution of an inverse problem using genetic algorithms and BEM to obtain the geometrical configuration of a vocal tract from acoustic information. This problem is first solved using an analytical method called the transfer matrix and then using the BEM and FEM for validation of three-dimensional effects. The BEM was then used to produce the acoustic behavior of the many vocal tracts models used for each generation of the genetic algorithm, in the end producing individuals which are very similar to the initial model. The second is the study of a non-resonant acoustic levitator based on the TinyLev project, which uses 72 ultrassonic transducers to produce acoustic levitation. Both 2D 3D BEM model is built for the geometry of the levitator and an experimental apparatus is set in place to produce Schlirien imaging of the standing acoustic wave on which fringes levitation is possible.

Chapter 8 is the final chapter and brings conclusions of this work.

[2] https://julialang.org

2 ACOUSTICS

In this chapter, the wave equation for acoustics will be discussed. Sound is the phenomenon attributed to the sensory faculty of hearing. It is produced by variations in the field of pressure of a physical medium and is typically defined as an audible mechanical wave of pressure and displacement. The fundamentals of acoustics are introduced, including a definition of the acoustic pressure field, speed of sound and the wave equation. A description of the class of related problems and the resonance concept and scattering are presented.

2.1 Acoustics equations

The sensation of sound is produced by mechanical waves travelling a compressible media untill it interacts with the human body, more especifically, the auditory system. The study of such waves is called acoustics and the waves, acoustic waves. Disturbances in the field of pressure exists in other forms, for example, ultrassonic and infrassonic waves and high intensity waves, such as those produced by a jet stream. Acoustic waves propagates through physical media by way of a compression and expansion cycle of fluid particles. The balance between inertial and compression/expansion forces is the mechanism of acoustic wave propagation.

The movement of the air particles as the wave passes through is such that a cycle of compression and rarefaction, which is best represented by an harmonic function. The distance between the peak and the valley of the wave is its wavelength λ. The frequency is defined as $f = c/\lambda$, where c is the speed of propagation. The angular frequency is $\omega = 2\pi f$ and the wavenumber is

$$k = \frac{\omega}{c} = \frac{2\pi}{\lambda}. \tag{2.1}$$

The wavenumber may be interpreted as the number of peaks within one wavelength.

2.1.1 Continuum hypothesis

A domain Ω of boundary Γ is defined as the acoustic volume. This domain is a compressible physical medium composed of molecules, which excert forces on each other and chemically interact. A small region of length l is used to describe the characteristics of the problem. The number of molecules in this region is a lot smaller than in the domain Ω, but still maintains its physical properties. For air, the mean distance between two molecules is $\Lambda = 6 \times 10^{-5}$ centimeters, or one thousandth of a millimiter, also called *mean free path*. There is an upper limit to the frequency this model is able to correctly predict, and that is $f = 5 \times 10^6$ Hz. For frequencies higher than f, the continuum hyphotesis is no longer valid and the model falls. So, as long as the characteristic length of the smallest particle of the model is higher than the *mean free path*, i.e. $l > \Lambda$, and the upper frequency is lower than f, the continuum model is preserved.

2.1.2 Equations of state for perfect gases

A gas is said to be perfect when at each instant only a small proportion of molecules are sufficiently close to others for them to interact. The internal energy E for unit of mass is approximately a function of its temperature, $E(T)$, proportional to the mean translational, rotational, and vibrational energy of an isolated molecule. Potential energy from inter-molecular forces are negligible. If the volume $V(t)$ is constant, an increase in temperature dT demands a heat input equal to the internal energy increase $E'(T)dT$. For this reason, $E'(T)$ is named heat capacity and it relates heat input to temperature increase. When an acoustic element is compressed, neighbouring elements excert work on it, and that increases the internal energy E, and so the element temperature also increases (LIGHTHILL, 1978). The internal energy change per unity mass due to compression work is:

$$dE = p(-d\rho)^{-1}, \tag{2.2}$$

which can also be expressed as $c_v dT$,where c_v is the specific heat capacity. The pressure in an ideal fluid can be expressed as a function of the temperature and pressure as:

$$p = RT\rho, \tag{2.3}$$

where R is the gas constant for the gas, in this work, air. The internal energy may be expressed as either a function of the internal energy E or entropy S, as will be seen later in the linearisation procedure.

2.1.3 Conservation laws

The mass of the volumetric domain Ω bounded by $\Gamma = \partial\Omega$, is defined as

$$m = \int_{\Omega} \rho(\mathbf{x}, t) d\Omega(\mathbf{x}) \tag{2.4}$$

where $\rho(\mathbf{x}, t)$ is the domains density, or mass per unit of volume, $\mathbf{x}$ is the position vector, t is time. Mass flux is only possible if there's a passage of fluid particles through the boundary Γ,

$$\dot{m} = \int_{\Gamma} \rho(\mathbf{x}, t)\mathbf{v}(\mathbf{x}, t) \cdot \mathbf{n}(\mathbf{x}) d\Gamma(\mathbf{x}) \tag{2.5}$$

where $\mathbf{v}(\mathbf{x}, t)$ is the fluid velocity and $\mathbf{n}(\mathbf{x})$ is the outwards pointing unit normal vector of the boundary Γ at point $\mathbf{x}$. Taking the derivative of the total mass of the system, it should equal the mass flux,

$$\frac{d}{dt} \int_{\Omega} \rho(\mathbf{x}, t) d\Omega(\mathbf{x}) = - \int_{\Gamma} \rho(\mathbf{x}, t)\mathbf{v}(\mathbf{x}, t) \cdot \mathbf{n}(\mathbf{x}) d\Gamma(\mathbf{x}). \tag{2.6}$$

The minus signal on the right hand side implies the flux is pointing outwards the boundary Γ. If the volume Ω is considered to be constant in time and space, the left hand side of Eq. (2.6) becomes

$$\frac{d}{dt} \int_{\Omega} \rho(\mathbf{x}, t) d\Omega(\mathbf{x}) = \int_{\Omega} \frac{\partial}{\partial t} \rho(\mathbf{x}, t) d\Omega(\mathbf{x}). \tag{2.7}$$

The right hand side of Eq. (2.6) is rewritten by applying the divergence theorem. From now on, the parenthesis will be ommited for clarity.

$$\int_{\Gamma} \rho\mathbf{v} \cdot \mathbf{n} d\Gamma = \int_{\Omega} \nabla \cdot (\rho\mathbf{v}) d\Omega. \tag{2.8}$$

Using Eq. (2.8) in Eq. (2.7) gives:

$$\int_{\Omega} \left[\frac{\partial}{\partial t} \rho + \nabla \cdot (\rho\mathbf{v}) \right] d\Omega = 0, \tag{2.9}$$

which is valid for each point inside the domain Ω. This implies that

$$\frac{\partial}{\partial t} \rho + \nabla \cdot (\rho\mathbf{v}) = 0, \tag{2.10}$$

which is known as the continuity equation.

The equation of motion for such an ideal fluid simply states that body forces acting on the domain and surface forces on the boundary yield an acceleration of the mass. Surface forces are usually caused by surrounding particles, while body forces are long range such as gravity. The acceleration of the mass is the time rate of the variation of the domain integral of the internal product of the specific mass ρ and the velocity of the fluid $\mathbf{v}$, or particle momentum. The equilibrium equation is thus:

$$\frac{d}{dt} \int_{\Omega} \rho\mathbf{v} d\Omega = \int_{\Gamma} \mathbf{f}_{\Gamma} d\Gamma + \int_{\Omega} \mathbf{f}_{\Omega} d\Omega, \tag{2.11}$$

where $\mathbf{f}_\Gamma$ and $\mathbf{f}_\Omega$ are the surface and body forces per unit area and volume, respectively. Unless the frequency of the soudn is of the same order as g/c, where g is gravity acceleration, gravitacional effects will not affect the acoustics of the problem. In most cases, acoustic behavior is not influenced by gravitational forces, so body forces can be neglected.

Regions where the vorticity tensor is different from zero are characterized by the existence of vortex, vortex propagation, and boundary layer. This phenomena are present in jet streams and turbulent flow in general and are not the focus of this work. Many body forces are excerted by those vortexes which considerably change the internal energy of the system, so that other formulations that consider these changes must be used. The vorticity tensor is related to the viscocity of the system and if the viscocity effects are negligible for the study cases, there's no need to take the vorticity tensor into account.

For the propagation of sound originated from jet streams or other phenomena with a high vorticity, two main regions may be defined: close and far field propagation. Close field is the region where vorticity effects are strong and must be considered as body forces. Far field is characterized by having negligible vorticity effects and thus the vorticity tensor may be written out of the momentum and energy equations. The surface force tensor is then reduced to:

$$\int_\Gamma \mathbf{f}_\Gamma d\Gamma = -\int_\Gamma p\mathbf{n} d\Gamma = \int_\Omega \nabla p d\Omega, \tag{2.12}$$

and thus, Eq. (2.11) may be rewritten as:

$$\frac{d}{dt}\int_\Omega \rho\mathbf{v} d\Omega = \int_\Omega \nabla p d\Omega. \tag{2.13}$$

The left hand side of Eq. 2.13 may be rewritten as:

$$\frac{d}{dt}\int_\Omega \rho\mathbf{v} d\Omega = \int_\Omega \rho\frac{D\mathbf{v}}{Dt} d\Omega, \tag{2.14}$$

where:

$$\frac{D}{Dt} = \frac{\partial}{\partial t} + \mathbf{v}\cdot\nabla \tag{2.15}$$

is known as the material derivative. Equation 2.11 may now be rewritten as:

$$\int_\Omega \left(\rho\frac{D\mathbf{v}}{Dt} + \nabla p\right) d\Omega = 0 \tag{2.16}$$

which, as with Eq. (2.10), is valid throughout the domain as:

$$\rho\frac{D\mathbf{v}}{Dt} = -\nabla p. \tag{2.17}$$

2.1.4 Linearisation

Let an acoustic medium be defined by its pressure p, density ρ, temperature T, and velocity $\mathbf{v}$. Each of these quantities is defined as the sum of a mean value and a perturbation. The mean velocity is set to zero, so that there is no net flow through the domain. Therefore,

$$\begin{cases} p = \bar{p} + \tilde{p}, & |\tilde{p}| << \bar{p}, \\ \rho = \bar{\rho} + \tilde{\rho}, & |\tilde{\rho}| << \bar{\rho}, \\ \mathbf{v} = \tilde{\mathbf{v}}, \end{cases}$$

where $\bar{p}$ stands for the mean value of p and $\tilde{p}$ stands for a perturbation of this mean value at a specific time instant. Writing a pressure infinitesimal perturbation on terms of entropy S and specific mass ρ yields:

$$dp = \left(\frac{\partial p}{\partial \rho}\right)_S d\rho + \left(\frac{\partial p}{\partial S}\right)_\rho d\S \tag{2.18}$$

Let the infinitesimal perturbation on pressure be an isentropic compression, so that the following relationship holds:

$$dp = c^2 d\rho \tag{2.19}$$

where c is the speed of wave propagation. Equation (2.18) can be expanded by a Taylor series. Using only the linear terms from this expansion is known as linearisation and Eqs. (2.10) and (2.11) become:

$$\nabla^2\phi + k^2\phi = 0, \tag{2.20}$$

where $k = \omega/c$ is the wavenumber and $\tilde{\phi} = \phi e^{i\omega t}$ is an acoustic potential defined by

$$\tilde{p} = \rho\frac{\partial \phi}{\partial t} \tag{2.21}$$

where $\tilde{p}$ is the acoustic pressure and ρ is the density of the fluid. This equation will describe the acoustic pressure field for media with no mean flow ($u = 0 + \tilde{u}$) and no mean temperature gradient ($\nabla T = 0$), such as vehicular habitacles, concert halls, and other important acoustic configurations. To describe the acoustic pressure field of extreme conditions such as cavities with a strong temperature gradient and mean flow, the equation must be corrected to account for such conditions.

There are four main conditions to describe acoustic phenomena:

- $\mathbf{v} = 0$ and $\nabla T = 0$

In this case, the wave equation is described by the well known elementary form:

$$\frac{\partial^2 \tilde{p}}{\partial t^2} - c_0^2 \nabla^2 \tilde{p} = 0 \tag{2.22}$$

- $\mathbf{v} \neq 0$ and $\nabla T = 0$

  The resulting equation is called the convective wave equation:

$$\left(\frac{\partial}{\partial t} - \mathbf{v}.\nabla\right)^2 \tilde{p} - c_0^2 \nabla^2 \tilde{p} = 0 \tag{2.23}$$

- $\mathbf{v} = 0$ and $\nabla T \neq 0$

  This wave equation is associated with stratification effects:

$$\frac{\partial^2 \tilde{p}}{\partial t^2} - \nabla.(c_0^2 \nabla \tilde{p}) = 0 \tag{2.24}$$

- $\mathbf{v} \neq 0$ and $\nabla T \neq 0$

  This equation takes into account the effects of both mean flow and temperature gradient:

$$\left(\frac{\partial}{\partial t} - \mathbf{v}.\nabla\right)^2 \tilde{p} - \nabla.(c_0^2 \nabla \tilde{p}) = 0 \tag{2.25}$$

In this work, only the wave equation for uniform medium will be used. This equation describes wave propagation on an uniform medium of ideal fluid. The fluid is a perfect gas, and the wave equation which will be solved is the Helmholtz Equation (2.20).

2.2 Acoustic field

Acoustic field is the name of the acoustic pressure distribution in a given domain. The acoustics field describes the acoustic properties of the domain, wether it's an infinite domain or a finte domain. For each point in an acoustic field, the velocity potential is known and so the acoustic pressure. Some common cases are plane waves, monopoles and dipoles. Each of them will be described below.

Planewaves are sound waves for which the phase, velocity and pressure are the same on the same plane. Sound waves in ducts being driven by a piston may be described by planewaves if the wavelength is considerably larger than the diameter of the duct. An example for a planewave is shown in Figure 2.1.

The equation for the velocity potential that describes such behavior is:

$$\phi(\mathbf{x}, t) = \phi_{max} e^{i(\omega t - k\mathbf{d}\cdot\mathbf{x})}, \tag{2.26}$$

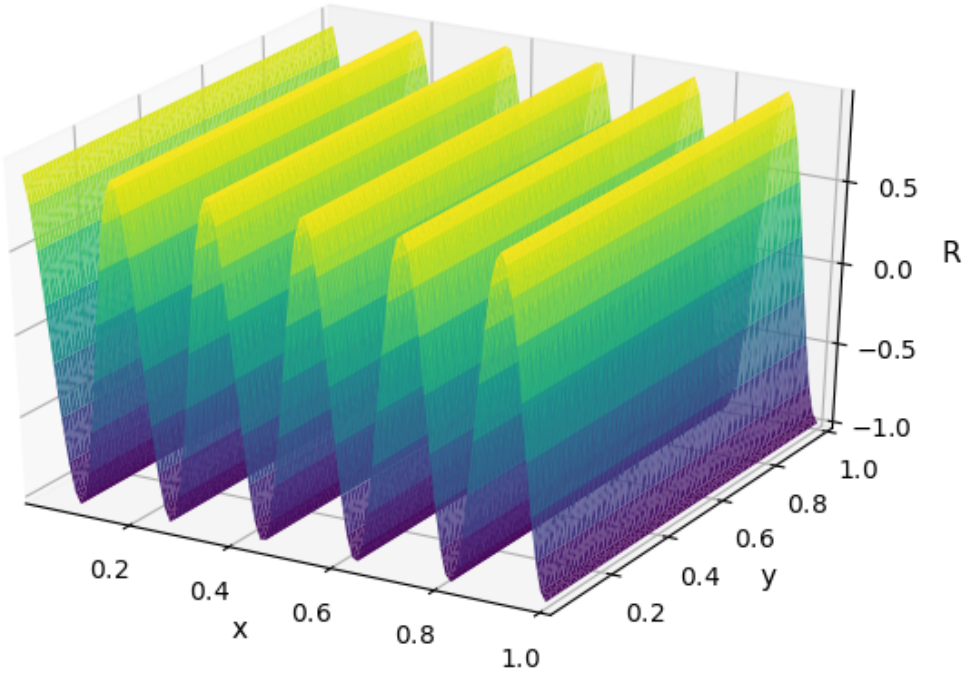

Figure 2.1 – Example of a planewave

at position $\mathbf{x}$, where k is the wavenumber, ω is the angular frequency and $\mathbf{d}$ is the direction of the planewave.

Monopoles are spherically symmetric source excitations of the acoustic medium. The source is considered to be a sphere of radius R_0 and all points on this sphere oscillate with the same velocity, phase and pressure. They can be used to represent most real life sources, for which the characteristic length of the model is of the same order as the wavelegth λ. This behavior may be obtained by solving the wave equation using spherical coordinates, and the solution for the velocity potential is (BRANCATI, 2010):

$$\phi(r,t) = \frac{AR_0^2}{r(1+ikR_0)} e^{i(\omega t - kr)}, \tag{2.27}$$

where r is the distance from the monopole, A is its amplitude and R_0 is the radius of the source. If $r \xrightarrow{\infty}$, the influence of the monopole is reduced to the air impedance. When $r >> \lambda$, the impedance is real and the phase is zero which represents planewaves. This is in accordance to the Sommerfeld solution for waves at large distances. An example of a monopole is given on Figure 2.2.

Dipoles may be represented by two monopoles close together with reverse phase. This produces a symmetric acoustic field, which can be seen on Figure 2.3.

Complex acoustic fields may be obtained by adding monopoles, which are called n-poles. The propagation of these acoustic waves are of interest as they may act as point sources of acoustic energy in internal and external problems in BEM. Monopoles are potential fundamental solutions for the wave equation, which will be presented later in this work.

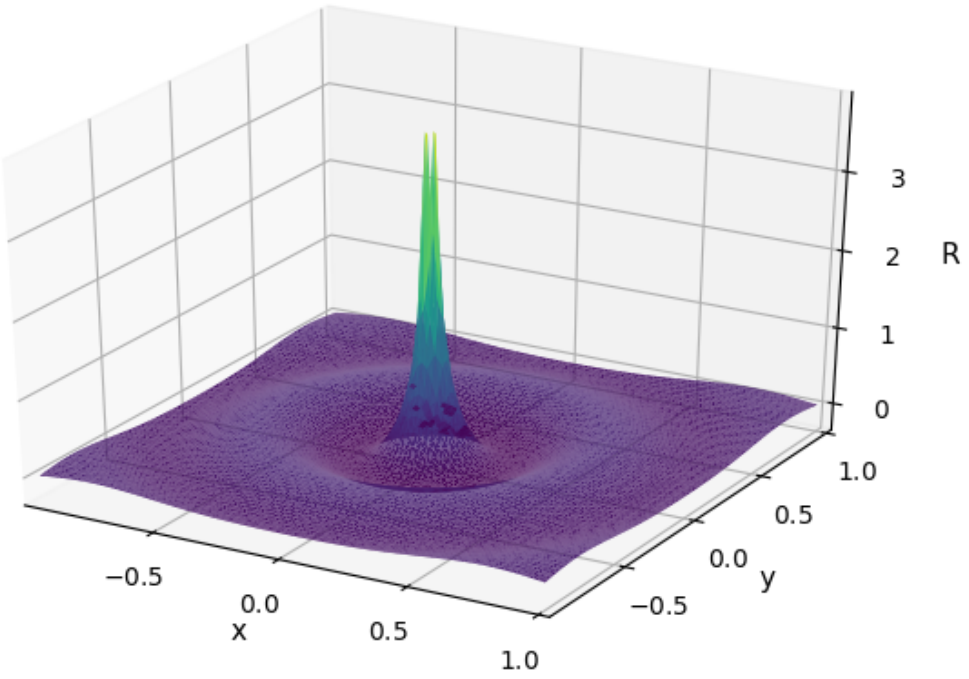

Figure 2.2 – Example of a monopole

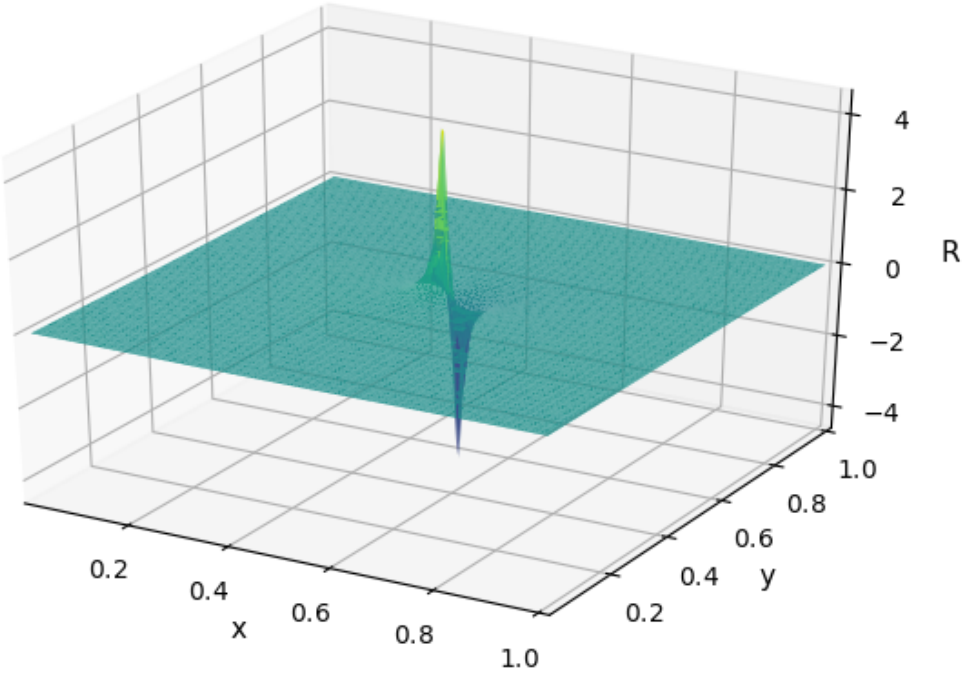

Figure 2.3 – Example of a dipole

2.3 Noise level

Not every acoustic wave can be perceived by humans. We are subjected to a hearing range just like every other animal. Humans can hear from 20-20000 Hz, which makes hearing the most wide range of perception in humans. Human can hear a lot, but other animals can hear more. The amplitude of the acoustic wave is related to the acoustic power. Acoustic power is measured in Db, which is a relative scale. It uses the lower bound of human hearing to create a logarithm scale. This bound is often called hearing threshold.

$$SPL = 20\log(\frac{P}{P_t}) \tag{2.28}$$

where P is the pressure of the acoustic wave and P_t is the threshold for human hearing, usually $P_t = 10^{-7}$ Pa.

3 THE BOUNDARY ELEMENT METHOD

This chapter introduces the boundary element method (BEM) formulation, some of the difficulties encoutered while solving the conventional boundary integral equation (CBIE) and the hypersingular boundary integral equation (HBIE) is presented for both Helmholtz and Laplace equations, while focusing on acoustics problems.

3.1 BEM formulation

Boundary integral equations describe the variables as functions only of values on the boundary Γ of the domain Ω, $\Gamma = \partial\Omega$. These kind of equations will describe a uniform medium very well. If a problem has characteristic diameter d, and if there is a large volume $V \approx d^3$, the description of the area will increase with the order of $A \approx d^2$. For a very big domain D^3 around a smaller, closed domain d^3, the volume of the problem will be propportional to D^3, but the surface area of interest may finally be d^2, and the field variable for any point in D^3 can be obtained using the BIE and the values on Γ. With this motivation in mind, consider an acoustic field described by the velocity potential ϕ and its gradient $\nabla\phi$. This field is bounded to a volumetric domain Ω.

The formulation used in this work is the direct boundary element method for the Helmholtz equation, described in more details by Dominguez (1993), Wrobel (2001) and Kirkup (2007).

The propagation of acoustic waves through a fluid medium Ω is described by the wave equation. When the motion is assumed to be traveling waves, the wave equation reduces to the Helmholtz equation and take the form shown in Eq. (3.1).

$$\nabla^2\phi + k^2\phi = 0 \tag{3.1}$$

where ϕ is a reduced velocity potential, $k = \omega/c$ is the wavenumber and ω is the angular frequency.

A test function ϕ^* is multiplied to Eq. (3.1), then the integral of the function in the domain Ω is carried out. If the result of the integral is set to zero, this procedure is

known as weighted residual, as shown in Eq. (3.2).

$$\int_{\Omega} \phi^*(\nabla^2\phi + k^2\phi)d\Omega = 0 \tag{3.2}$$

from the identity $\phi^*\nabla^2\phi = \nabla \cdot (\phi^*\nabla\phi) - \nabla\phi^*.\nabla\phi$, one obtain:

$$\int_{\Omega} \left[\nabla.(\phi^*\nabla\phi) - \nabla.(\phi\nabla\phi^*)\right] d\Omega = -\int_{\Omega} \phi \left(\nabla^2\phi^* + k^2\phi^*\right) d\Omega \tag{3.3}$$

The boundary integral equation for the problem can be found by starting from applying Green's second identity:

$$\int_{\Omega} \left[\nabla.(\phi^*\nabla\phi) - \nabla.(\phi\nabla\phi^*)\right] d\Omega = -\int_{\Gamma} \left(\phi\frac{\partial\phi^*}{\partial\mathbf{n}} - \phi^*\frac{\partial\phi}{\partial n}\right) d\Gamma \tag{3.4}$$

where Γ is the boundary of domain Ω. Substituting the right part of Eq. (3.3) in Eq. (3.4) gives:

$$\int_{\Omega} \phi \left(\nabla^2\phi^* + k^2\phi^*\right) d\Omega = \int_{\Gamma} \left(\phi\frac{\partial\phi^*}{\partial n} - \phi^*\frac{\partial\phi}{\partial n}\right) d\Gamma \tag{3.5}$$

If the weight function ϕ^* satisfies Eq. (3.6), then the function is said to be a fundamental solution of Eq. (3.1). It corresponds to the field generated by a unit concentrated harmonic source at point X' and wave number k.

$$\nabla^2\phi^*(X', x, k) + k^2\phi^*(X', x, k) = -\delta(x - X') \tag{3.6}$$

where X' is the source point, x is the field point and δ is the Dirac delta function. Introducing the property of Eq. (3.6) in Eq. (3.5) produces the boundary integral representation of the problem:

$$\phi(X') = \int_{\Gamma} \frac{\partial\phi(x)}{\partial n}\phi^*(X', x, k)d\Gamma - \int_{\Gamma} \phi(x)\frac{\partial\phi^*(X', x, k)}{\partial n}d\Gamma \tag{3.7}$$

The fundamental solution of the Helmholtz equation is physically defined as a point source perturbation of an infinite domain.

The fundamental solution and the normal derivative of the fundamental solution for the three-dimensional Helmholtz equation are

$$\phi^* = -\frac{e^{ikr}}{4\pi r}, \tag{3.8}$$

$$\frac{\partial\phi^*}{\partial n} = -\frac{e^{ikr}}{4\pi r}\left(ik - \frac{1}{r}\right)\frac{\partial r}{\partial n}, \tag{3.9}$$

where $r = |X' - x|$ is the distance between the source point X' and the field point x.

Taking X' to the boundary Γ in Eq. (3.7), in view of the behaviour of the fundamental solution when $X' \to x'$, $x' \in \Gamma$ produces Eq. (3.10).

$$c(x')\phi(x') = \int_\Gamma \frac{\partial \phi(x)}{\partial n}\phi^*(x', x, k)d\Gamma - \int_\Gamma \phi(x)\frac{\partial \phi^*(x', x, k)}{\partial n}d\Gamma \tag{3.10}$$

where the jump term is $c(x') = 1/2$ for smooth boundary on x'.

3.2 Incident waves and concentrated acoustics sources

In external and scattering problems, it is often necessary to include a concentrated acoustic source or an incident wave, which can both be included in the formulation in a similar manner.

Considering Eq. (3.18), one may include a concentrated acoustic source in point x_s, then

$$\begin{aligned}\gamma\phi(x') + \int_\Gamma \left[\frac{\partial \phi^*(x', x, k)}{\partial n} - \frac{\partial \bar{\phi}^*(x', x)}{\partial n}\right]\phi(x)d\Gamma \\ + \int_\Gamma \frac{\partial \bar{\phi}^*(x', x)}{\partial n}\left[\phi(x) - \phi(x')\right]d\Gamma \\ = \int_\Gamma \frac{\partial \phi(x)}{\partial n}\phi^*(x', x, k)d\Gamma + Q\phi^*(x_s, x', k),\end{aligned} \tag{3.11}$$

where Q is the acoustic source amplitude. This source will act as an acoustic monopole.

Another application might be to study the reflection patterns of the boundary from an incident acoustic wave. This is possible by including another term in Eq. (3.11).

$$\begin{aligned}&\gamma\phi(x') \\ &\quad + \int_\Gamma \left[\frac{\partial \phi^*(x', x, k)}{\partial n} - \frac{\partial \bar{\phi}^*(x', x)}{\partial n}\right]\phi(x)d\Gamma \\ &\quad + \int_\Gamma \frac{\partial \bar{\phi}^*(x', x)}{\partial n}\left[\phi(x) - \phi(x')\right]d\Gamma \\ &\quad = \int_\Gamma \frac{\partial \phi(x)}{\partial n}\phi^*(x', x, k)d\Gamma + Q\phi^*(x_s, x', k) + Ae^{ikdx'},\end{aligned} \tag{3.12}$$

where d is the unit direction of the wave ($|d| = 1$) and A is the incident wave amplitude.

3.3 Regularisation

When the element contains the source point x', the integration is said to be singular. A weak singular equation tends to infinity as r approaches 0. This effect and mitigation techniques are discussed for each element type.

The integration is performed mostly using the Gaussian quadrature. To perform the numerical integration, the Gauss-Legendre method is used. The results are obtained usually using 12 points for 2D elements (parametric variable ξ) and 6 for 3D elements (parametric variables ξ and η).

It's important to notice that the fundamental solution and its derivative (Eqs. (3.8) and (3.9), respectively) contain singularities that might be difficult to evaluate when r is small. Even though it is possible to use numerical integration to compute the singular integral which appear in the right hand side of Eq. (3.10), it has been shown that there are other approaches which might require less computing time, such as to use weakly singular forms of these integrals (GONG; DONG; BAI, 2017; ZHANG; QU; GU, 2013).

3.3.1 Singularity handling

There are many ways to handle the singular integrals which arise from the boundary integral equations (BIE) of the BEM. There are mainly two ways of handling the singularities: either by direct handling of the BIE and indirect handling. Direct handling deals with the original BIE and the handling is mainly made by changing numerical integration parameters, such as integration point position and weights. Indirect handling consists of using other forms of the BIE such as a weakly singular formulation, which is obtained analytically. Some examples of direct and indirect handling are given below.

TELLES (1987) presents a novel self-adaptive transformation to the traditional Gaussian quadrature which is found to greatly improve the accuracy within the near-singularity range.

The idea behind the proposed self-adaptive transformation is to lump the points towards the minimum source distance position without subdividing the element to generate new points. This is performed by applying a new coordinate transformation based on a complete third degree polynomial between the element and the source point: $\xi = a\lambda^3 + b\lambda^2 + c\lambda + d$. As an aditional property, both the first and second derivatives of the transformation are zero, i.e., $\frac{\partial \xi}{\partial \lambda} = 0$ and $\frac{\partial^2 \xi}{\partial \lambda^2} = 0$ at the singularity point. This means that the Jacobian of the transformation is zero at the singularity, further reducing its influence and facilitating the integration.

It moves the Gauss point positions towards the source point, using only information from the local coordinate of the element point closest to the source point and the minimum source to element distance. This means that, for each source point positioning, there is a new set of quadrature points that needs to be created. As most of the used parametric curves have clear physical node collocation points, described by parametric points, the implementation of this transformation is fairly straightforward and powerful.

Gong et al (2017) discusses nearly singular integrals which arise in the IGABEM when studying thin-body/coating structures. An exponential transformation is carried

out to weaken or remove the near singularities that appear in 2D/3D potential problems and presents numerical results of the procedure. The discretization used NURBS curves and the transformation is carried out in the parametric domain. This transformation is used by the author to directly evaluate the nearly singular integrals over isogeometric elements with high accuracy, even when the nearly singular points are very close to the boundary of the model.

A weakly singular form of the BIE may be obtained by expressing the jump term $c(x')$ from Eq. (3.10) as

$$c(x') = \gamma - \int_\Gamma \frac{\partial \phi^*(x', x, 0)}{\partial n} d\Gamma \tag{3.13}$$

where $\gamma = 1$ for infinite domains and 0 for finite domains. The integral on the right hand side of Eq. (3.13) contains the fundamental solution for the Helmholtz equation when there is no wave propagation ($k = 0$), which corresponds to a static form of the wave equation, namely, the Laplace equation. Therefore, it's possible to rewrite Eq. (3.13) as

$$c(x') = \gamma - \int_\Gamma \frac{\partial \bar{\phi}^*(x', x)}{\partial n} d\Gamma \tag{3.14}$$

where an overbar is added to distinguish the fundamental solution for the Laplace equation from the fundamental solution for the Helmholtz equation. The fundamental solution for the Laplace equation and its normal derivative for three-dimensions are

$$\bar{\phi}^* = -\frac{1}{4\pi r}, \tag{3.15}$$

$$\frac{\partial \bar{\phi}^*}{\partial n} = -\frac{1}{4\pi r^2} \frac{\partial r}{\partial n}, \tag{3.16}$$

where $r = |x' - x|$ is the distance between the source point x' and the field point x.

Substituting Eq. (3.14) in (3.10), one obtains

$$\begin{aligned} &\left(\gamma - \int_\Gamma \frac{\partial \bar{\phi}^*(x', x)}{\partial n} d\Gamma\right) \phi(x') \\ &\qquad = \int_\Gamma \frac{\partial \phi(x)}{\partial n} \phi^*(x', x, k) d\Gamma - \int_\Gamma \phi(x) \frac{\partial \phi^*(x', x, k)}{\partial n} d\Gamma \end{aligned} \tag{3.17}$$

From Eq. (3.17), its possible to reorganize the terms to obtain a BIE in which there is no strongly singular integrals:

$$\begin{aligned} \gamma \phi(x') + \int_\Gamma \left[\frac{\partial \phi^*(x', x, k)}{\partial n} - \frac{\partial \bar{\phi}^*(x', x)}{\partial n}\right] \phi(x) d\Gamma& \\ + \int_\Gamma \frac{\partial \bar{\phi}^*(x', x)}{\partial n} \left[\phi(x) - \phi(x')\right] d\Gamma& \\ = \int_\Gamma \frac{\partial \phi(x)}{\partial n} \phi^*(x', x, k) d\Gamma& \end{aligned} \tag{3.18}$$

Now, every integral is weakly singular at most and most numerical integration techniques can be used directly.

For the purpose of this work, one regularization handling strategy was chosen: the Telles transformation. Equation (3.18) is commonly known as weakly singular form of the boundary integral equation. Now, all three integrals are weakly singular and can be handled by numerical integration schemes directly. The use of the Telles transformation facilitates the integration as the quadrature points will be distributed in such a way that the integration will require less points. This effect will be shown in the following sections as different discretization of the boundary are shown.

3.3.2 Fictitious eigenfrequency difficulty

In external problems, fictitious eigenfrequencies may be encountered on BEM models. These are associated with the resonant frequencies of interior modes. This results in an expurious mode being detected on the external problem. When performing a frequency sweep to obtain a frequency response, a peak may appear at the frequency of an internal problem. The mode related to it has no physical meaning. Several solutions have been proposed to solve this problem, including the CHIEF or Schenck (2013) procedure and hypersingular boundary integral equations (HBIE) alongside the conventional BIE (CBIE), which uses the derivative of Eq. (3.10).

The CHIEF procedure, or Schenck method consists of adding new collocation points to the model to avoid this eigenfrequency difficulty. Although the method requires more points and, therefore, more computing time, the difference is very small.

The hypersingular formulation of the BIE uses the derivative of the BIE and it can increase the computational effort greatly as it requires that another set of integral equations, one which includes a hypersingular integral. Taking the derivative of the integral equation Eq. (3.12):

$$\begin{aligned}\bar{c}(x')\phi(x') = \int_\Gamma \frac{\partial\phi(x)}{\partial n}\frac{\partial\phi^*(x',x,k)}{\partial n}d\Gamma \\ - \int_\Gamma \phi(x)\frac{\partial^2\phi^*(x',x,k)}{\partial n^2}d\Gamma \\ + Q\frac{\partial\phi^*(x_s,x',k)}{\partial n} + A\frac{\partial(e^{ikdx'})}{\partial n}\end{aligned} \tag{3.19}$$

where $c(\bar{x'})$ is the derivative of the jump term.

The improved formulation is simply a linear combination of Eqs. (3.12) and (3.19). This combination can be written as shown in Eq. (3.20).

$$CBIE + \beta HBIE = 0 \tag{3.20}$$

where $\beta = i/k$ is the coupling constant. This formulation was shown to yield unique solutions at all frequencies and is referred to as Burton-Miller formulation for acoustic wave problems.

The improved formulation shown in Eq. (3.20) can still be used to remove the fictitious eigenfrequency difficulty, yielding Eq. (3.21), named from now on HBIER.

$$\begin{aligned}\phi(x') + \int_\Gamma \frac{\partial^2 \bar{\phi}^*(x',x)}{\partial n^2}[\phi(x) - \phi(x')]d\Gamma \\ + \int_\Gamma \left[\frac{\partial^2 \phi^*(x',x,k)}{\partial n^2} - \frac{\partial^2 \bar{\phi}^*(x',x)}{\partial n^2}\right]\phi(x)d\Gamma \\ = \int_\Gamma \frac{\partial \phi^*(x',x,k)}{\partial n}\frac{\partial \phi(x)}{\partial n}d\Gamma\end{aligned} \tag{3.21}$$

Now the regularized improved formulation can be obtained through

$$CBIER + \beta HBIER = 0. \tag{3.22}$$

Another alternative is to skip the frequencies of the interior modes when solving for exterior modes, which requires solving the interior problems before solving any exterior problem.

3.4 Discretization of the boundary integral equation in boundary elements

The discretization of the boundary of the domain is the process of transforming a continuous domain into a discrete one by the use of shape functions that describe the boundary. This description may be an approximation of the actual boundary of the domain, as with linear, quadratic or cubic interpolation of points which reside in the boundary, or a factual description of the boundary as with an isogeometric approach, either by using Bézier patches, B-Splines, NURBS or T-Splines. In this section, the most important discretization procedures are reviewed.

The BEM is a numerical method of solution of boundary integral equations, based on a discretization procedure. Two types of approximations are required for the application of the method: the first is geometrical and involves a subdivision of boundary Γ in N small segments, such that $\Gamma \approx \sum_{i=1}^{N} \Gamma_i$. This approximation can be viewed in Figure 3.1. The second is an approximation of the variation of the velocity potential and its normal derivative within each element.

Three types of elements will be described in the next section: constant, linear and quadratic elements. Each discretization can be seen in Figure 3.2.

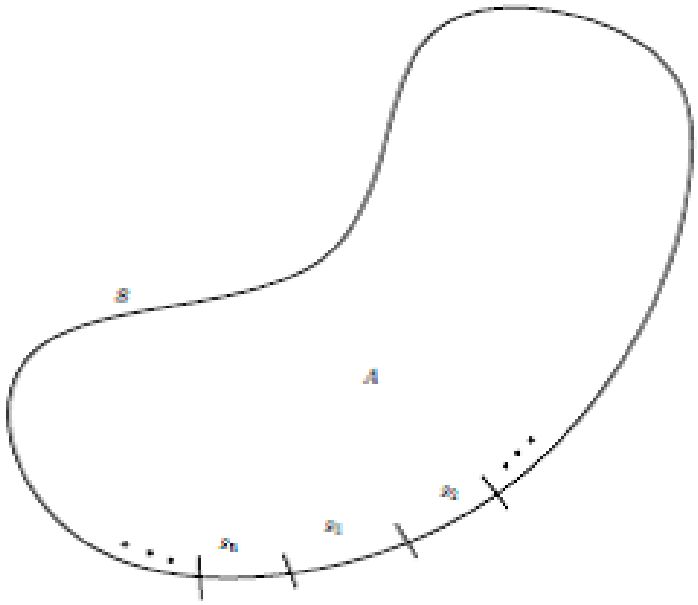

Figure 3.1 – Discretization of the boundary into elements

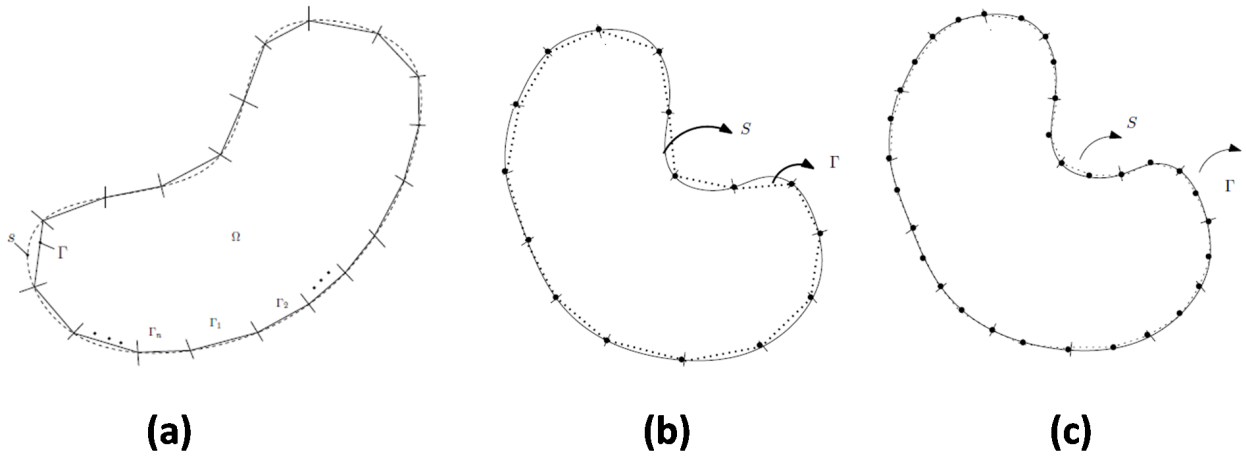

Figure 3.2 – (a) Constant, (b) linear and (c) quadratic elements

3.4.1 Constant elements

The simplest possible approximation is the constant one, which assumes that both ϕ and $\partial\phi/\partial\mathbf{n}$ are constant within each element and equal to their value at the midpoint. Introducing this approximations, one obtains

$$c_j\phi_j = \sum_{i=1}^{N} \frac{\partial\phi_i}{\partial\mathbf{n}} \int_{\Gamma_i} \phi^*(x', x)d\Gamma - \sum_{i=1}^{N} \phi_i \int_{\Gamma_i} \frac{\partial\phi^*(x', x)}{\partial\mathbf{n}} d\Gamma \tag{3.23}$$

where ϕ_i and $\partial\phi_i/\partial\mathbf{n}$ are the values of ϕ and $\partial\phi/\partial\mathbf{n}$ at the node located in the midpoint of element i. In the case of constant elements, the number of nodes is equal to the number of elements.

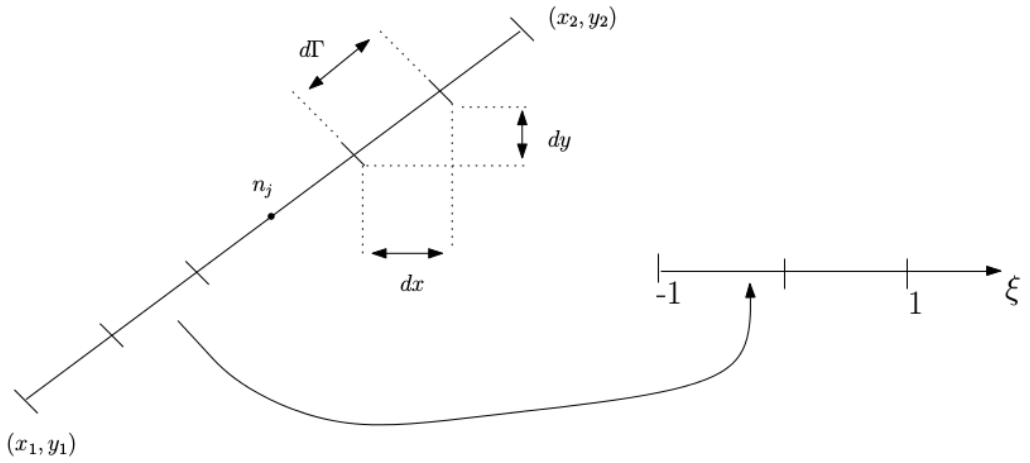

Figure 3.3 – Global and local coordinate systems for constant elements

Let

$$H_{ij} = \left(\int_{\Gamma_i} \frac{\partial \phi^*}{\partial \mathbf{n}} d\Gamma \right) \tag{3.24}$$

with the special case where $i = j$ being written as

$$H_{ii} = \frac{1}{2} + \left(\int_{\Gamma_i} \frac{\partial \phi^*}{\partial \mathbf{n}} d\Gamma \right) \tag{3.25}$$

and

$$G_{ij} = \left(\int_{\Gamma_i} \phi^* d\Gamma \right) \tag{3.26}$$

Equation (3.23) takes the following form

$$\sum_{i=1}^{N} H_{ij} \phi_i = \sum_{i=1}^{N} G_{ij} \frac{\partial \phi_i}{\partial \mathbf{n}} \tag{3.27}$$

Using a collocation technique to all nodal points along the boundary gives a system of equations which can be written in matrix form as

$$\mathbf{H}\phi = \mathbf{G}\mathbf{q}, \tag{3.28}$$

where ϕ and $\mathbf{q}$ are vectors containing the nodal values of the potential and its normal derivative; $\mathbf{H}$ and $\mathbf{G}$ are square $N \times N$ matrices of influence coefficients.

The numerical integration can be done by using Gauss-Legendre quadrature. For that goal, a change of variables must be made, from the start to the end of the element to [-1,1]. This change can be made by introducing a coordinate change controlled by a shape function. Consider the straight element shown in Figure 3.3.

To perform the numerical integration, the domain of the integral must be [-1,1], so a local coordinate system is imposed with new coordinate $\xi \in [-1, 1]$. This procedure can be done by using shape functions

$$x = N_1(\xi)x_1 + N_2(\xi)x_2, \tag{3.29}$$

$$y = N_1(\xi)y_1 + N_2(\xi)y_2, \tag{3.30}$$

where N_1 and N_2 are the linear shape functions which will describe the geometry of the boundary. Suitable shape functions are

$$N_1 = \frac{1}{2}(1 - \xi) \tag{3.31}$$

and

$$N_2 = \frac{1}{2}(1 + \xi). \tag{3.32}$$

To carry out the coordinate change, a first step is taking the derivative of the boundary with respect to ξ

$$\frac{d\Gamma}{d\xi} = \frac{L}{2}, \tag{3.33}$$

where $L = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$ is the total length of the element. Now, the integral equation can be written in the domain of ξ. The elements of matrices $\mathbf{G}$ (A.18) and $\mathbf{H}$ (A.17) are described below, respectively.

$$G_{ij} = \int_{-1}^{1} \phi^*(\xi, k)\frac{d\Gamma}{d\xi}d\xi \tag{3.34}$$

$$H_{ij} = \int_{-1}^{1} \frac{\partial\phi^*}{\partial\mathbf{n}}(\xi, k)\frac{d\Gamma}{d\xi}d\xi \tag{3.35}$$

3.4.2 Linear elements

The geometry can also be described by a first degree polynomial, with two nodes at each extremity of the element. This formulation is isoparametric, which means that the same shape functions used to describe the geometry of the boundary is used to interpolate the variables throughout the element. Linear elements contain two nodes per element, so for each element,

$$\phi = N_1(\xi)\phi_1 + N_2(\xi)\phi_2 \tag{3.36}$$

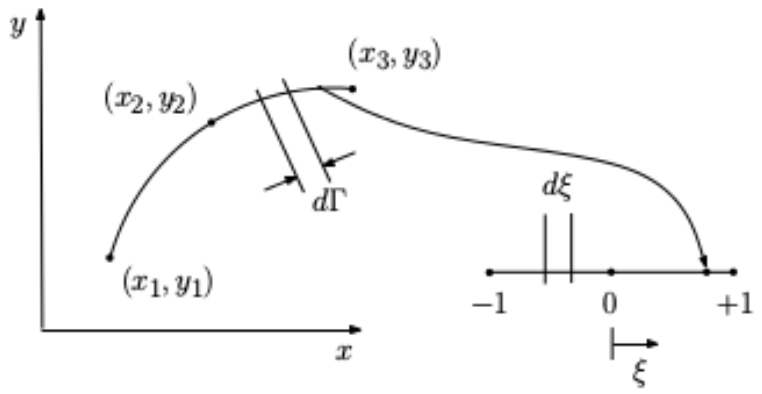

Figure 3.4 – Global and local coordinate systems for quadratic elements

$$\frac{\partial \phi}{\partial \mathbf{n}} = N_1(\xi)\frac{\partial \phi_1}{\partial \mathbf{n}} + N_2(\xi)\frac{\partial \phi_2}{\partial \mathbf{n}} \tag{3.37}$$

Because there are two nodes at each element, the discretization of integral equation Eq. (3.10) becomes

$$c(x')\phi(x') = \sum_{j=1}^{N}\sum_{a=1}^{2}\left[\int_{\Gamma_j} N_a \phi^*(\xi,k)\frac{d\Gamma}{d\xi}Jd\xi\frac{\partial \phi_a(x)}{\partial \mathbf{n}} - \int_{\Gamma_j} N_a \frac{\partial \phi^*(\xi,k)}{\partial \mathbf{n}}\frac{d\Gamma}{d\xi}Jd\xi\phi_a(x)\right], \tag{3.38}$$

where $J = \sqrt{\left(\frac{dy_1}{d\xi}\right)^2 + \left(\frac{dy_2}{d\xi}\right)^2}$ is the Jacobian of the coordinate transformation.

3.4.3 Quadratic elements

Other shape functions can be used to describe boundary Γ, such as second degree polynomials. If these kind of polynomials are used, three points of the boundary are needed to describe the geometry

$$x = N_1(\xi)x_1 + N_2(\xi)x_2 + N_3(\xi)x_3, \tag{3.39}$$

$$y = N_1(\xi)y_1 + N_2(\xi)y_2 + N_3(\xi)y_3, \tag{3.40}$$

where N_1 and N_2 and N_3 are the quadratic shape functions which will describe the geometry of the boundary. Suitable shape functions are

$$N_1 = \frac{\xi}{2}(1-\xi), \tag{3.41}$$

$$N_2 = (1-\xi)(1+\xi), \tag{3.42}$$

$$N_3 = \frac{\xi}{2}(1+\xi), \tag{3.43}$$

then

$$c(x')\phi(x') = \sum_{j=1}^{N}\sum_{a=1}^{3}\left[\int_{\Gamma_j} N_a\phi^*(\xi,k)\frac{d\Gamma}{d\xi}Jd\xi\frac{\partial\phi_a(x)}{\partial\mathbf{n}} - \int_{\Gamma_j} N_a\frac{\partial\phi^*(\xi,k)}{\partial\mathbf{n}}\frac{d\Gamma}{d\xi}Jd\xi\phi_a(x)\right], \tag{3.44}$$

where $J = \sqrt{\left(\frac{dy_1}{d\xi}\right)^2 + \left(\frac{dy_2}{d\xi}\right)^2 + \left(\frac{dy_3}{d\xi}\right)^2}$ is the Jacobian of the coordinate transformation.

3.5 Values at domain points

Once the values of the unknowns are obtained in the boundary, the value of the velocity potential in any point in the domain is obtained using

$$\phi_j = \sum_{i=1}^{N}\frac{\partial\phi_i}{\partial\mathbf{n}}\int_{\Gamma_i}\phi^*(x',x)d\Gamma - \sum_{i=1}^{N}\phi_i\int_{\Gamma_i}\frac{\partial\phi^*(x',x)}{\partial\mathbf{n}}d\Gamma, \tag{3.45}$$

where ϕ_j is the value of the velocity potential for internal point j.

Equation (3.45) is used to obtain the velocity potential in any external point, which is the external BEM version of internal points. If the elements are presented in a clock-wise direction, the normal of the boundary curve points into the curve and the BEM domain is external to the boundary. Variable ϕ_j can be evaluated for any point j located outside of boundary Γ using Eq. (3.45).

3.6 Solving the linear system of equations

Once the boundary conditions of the problem are applied, the problem stated in Eq. (3.28) can be rearranged in the form

$$\mathbf{Ax} = \mathbf{b}, \tag{3.46}$$

where $\mathbf{A}$ is a matrix containing all coefficients, $\mathbf{x}$ is a vector containing all unknowns of the problem and $\mathbf{b}$ is the 'load' vector. This system is solved using a standard direct solver. A reverse arrangement is used to obtain the original system so that the potential ϕ and the flux $\mathbf{q}$ can be assigned to each node.

The system of equations described in Eq. (3.47) can be solved directly using gaussian elimination or LU decomposition. Another alternative is to solve the system using iterative methods such as the generalized minimal residual method (GMRES).

3.6.1 GMRES and block matrices

To solve the BIE using a block matrix representation of the influence matrices, an iterative solution strategy such as GMRES is necessary. This is because one does not have access to every term of the matrix to perform traditional direct methods such as Gauss elimination and LU decomposition.

The GMRES is an iterative method. The method actually solves the system by trial and error, estimating the error of the matrix vector product. This makes it suitable to solve a problem in which the matrix is stored in a low-rank manner. To see how this works, consider the matrix product given by:

$$\mathbf{A}\mathbf{x} - \mathbf{b} = 0. \tag{3.47}$$

In this representation, it's possible to express the product of an approximated $\tilde{\mathbf{x}} \approx \mathbf{x}$. This product will result in a non-null value, which the GMRES will minimize.

$$\mathbf{A}\tilde{\mathbf{x}} - \mathbf{b} = \epsilon. \tag{3.48}$$

4 ISOGEOMETRIC ANALYSIS USING BEM

This chapter is dedicated to the study of Isogeometric Analysis, commonly abreviated as IGA. In section 4.3, the concept of Bézier curves and the mathematical formulation of NURBS is shown. Section 4.5 describes the implementation of NURBS in BEM, to obtain an IGA BEM (Isogeometric Analysis using the Boundary Element Method). The idea behind Isogeometric Analysis is to use the same geometrical description of the problem for both the design and the numerical analysis. This integration between computer-aided design (CAD) and numerical methods such as the BEM can greatly improve the speed and ease for the end-user to obtain considerably accurate models, for it reduces the necessity of meshing the geometry into Lagrangian shape functions, for example. Most CAD programs can generate Non-uniform rational B-splines (NURBS) geometrical descriptions of the boundary of the problems and this work will focus on this kind of mathematical object.

4.1 Bézier curves

The mathematics of the NURBS can be derived from Bézier curves. Bézier curves were put in place by Pierre Bézier to develop computer-aided models of automobiles as an indirect and intuitive way of specifying and controling the parameters of curves computationally. This curves substituted other parametric representations used in the industry (PIEGL; TILLER, 1997).

Bézier curves are obtained by a concatenation of linear interpolation of the control points. To illustrate how a Bézier curve may be obtained, the procedure to obtain a cubic Bézier curve is carried out. Let $\mathbf{b}$ be the collection of $n+1$ points such that for $i = 0, 1, 2, ..., n$, $\mathbf{b^i} \in \mathbb{R}^d$, where the dimension $d = 2$ or 3 corresponds to points in two or three-dimensional spaces. In this example, four points are used:

$$\mathbf{b} = \left\{ \begin{array}{c} \mathbf{b^0} \\ \mathbf{b^1} \\ \mathbf{b^2} \\ \mathbf{b^3} \end{array} \right\} = \left[\begin{array}{c} 0\ 0 \\ 0\ 4 \\ 4\ 4 \\ 4\ 0 \end{array} \right] \tag{4.1}$$

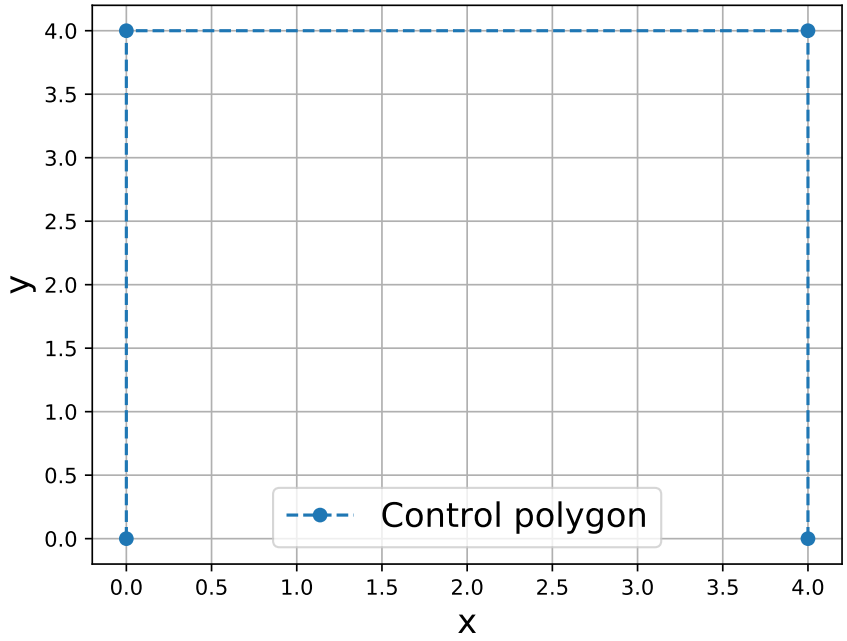

Figure 4.1 – Four control points

These points are called *control points*.

Let $t \in [0, 1]$ be a linear interpolation parameter between each two consecutive points. The control points for the Bézier curve, $\mathbf{b}$, will be referenced by $\mathbf{b^i}_k$, so that new points created by the k-th interpolation may be named $\mathbf{b^i_{k+1}}$. Therefore,

$$\mathbf{b^i_{k+1}}(t) = \mathbf{b^i_k} \cdot (1 - t) + \mathbf{b^{i+1}_k} \cdot t. \tag{4.2}$$

Now, each point at the interpolated curve $C(t)$, is interpolated again. For example, when $t = 0.3$, there will be a point on a third of the way from each consecutive point. These points are also interpolated consecutively to obtain the configuration showed in Fig 4.2.

The last point from the interpolation will be on the Bézier curve. Therefore, a complete Bézier curve may be constructed using this successive linear interpolation algorithm for any number of control points. This process is known as the de Casteljau's algorithm. It's clear that the curve is built using only linear interpolation between the control points. The last point, which would correspond to the control points $\mathbf{b}^3$ is actually the point on the Bézier curve.

This algorithm may be written as a single function which will return the last value for the curve of an arbitrary degree, defined by the number of control points.

Because the points which describe the geometry are not contained in the curve, this type of representation is known as isogeometric as opposed to an isoparametric representation in which the points used to describe the geometry are contained in the curve.

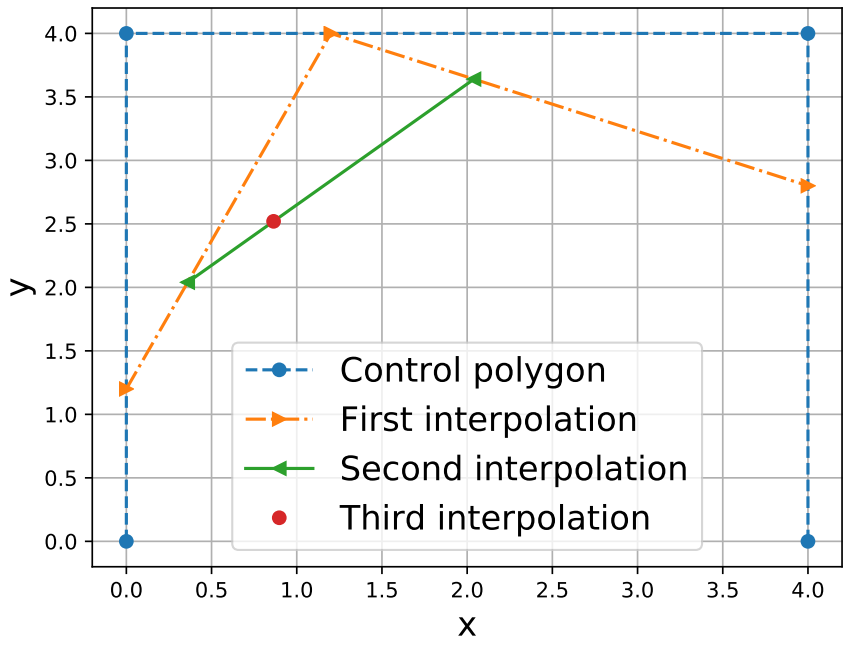

Figure 4.2 – Successive linear interpolations between the control points

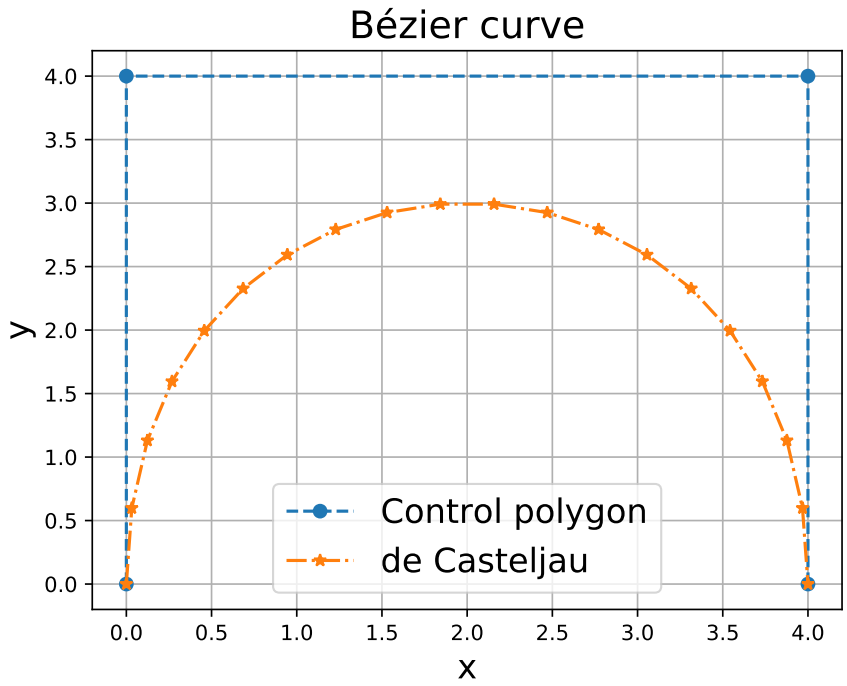

Figure 4.3 – Bézier curve constructed using De Casteljau's algorithm

Some characteristics of the Bézier curve can be observed from Figure 4.3. First of all, one notices that point $\mathbf{b}^2$ does not belong to the Bézier curve, while points $\mathbf{b}^0$ and $\mathbf{b}^3$ does. This is important to know, not all control points will belong to the curve, only the first and last points. Another characteristic is that the Bézier curve is contained into the region between the polygon formed by the blue line segments. The Bézier curve can be obtained using

$$\mathbf{P}(t) = \sum_{i=0}^{n} \mathbf{b^i} J_{n,i}(t), 0 \leq t \leq 1 \tag{4.3}$$

where the coefficients $\mathbf{b^i}$ are the control points, which form the control polygon, and $J_{n,i}(t)$ is the Bernstein basis,

$$J_{n,i}(t) = \frac{n!}{i!(n-i)!} u^i (1-u)^{n-i}. \tag{4.4}$$

The number of control points minus one is the degree of the polynomial of the Bézier curve, i.e., $p = n + 1$ or $p = k - 1$. The first and last control points are the initial and final points of the curve. The tangent vector at the beginning and ending of the curve is equal to the segments of the control polygon. The curve is always within the complex boundary of the control points.

Each point in the Bézier curve is calculated as an weighted sum of all of the control points. Change in any control point affects the curve globally.

The derivatives of a Bézier curve can be obtained by applying the following equations

$$\frac{\mathbf{dP}}{dt}(t) = \sum_{i=0}^{n} \mathbf{b^i} J'_{n,i}(t), 0 \leq t \leq 1, \tag{4.5}$$

$$\frac{dJ_{n,i}}{dt}(t) = \frac{i - nt}{t(1-t)} J_{n,i}. \tag{4.6}$$

4.2 Bézier patches and continuity

Bézier curves may be connected by their endpoints into a structure known as a Bézier patch. Continuity is guaranteed by joining the endpoints of two Bézier curves, so that the last control point of the first curve is the first of the second curve. This continuity is called C^0, continuity of position. The continuity of velocity, i.e., of the derivative of the curve at the junction is C^1 continuity. This means that the slope of the Bézier curves are the same at the shared control point. C^2 is the continuity of the second derivative of the curve. How this continuity is achieved is related to the position of the control points and the parametric space of each curve, as described below.

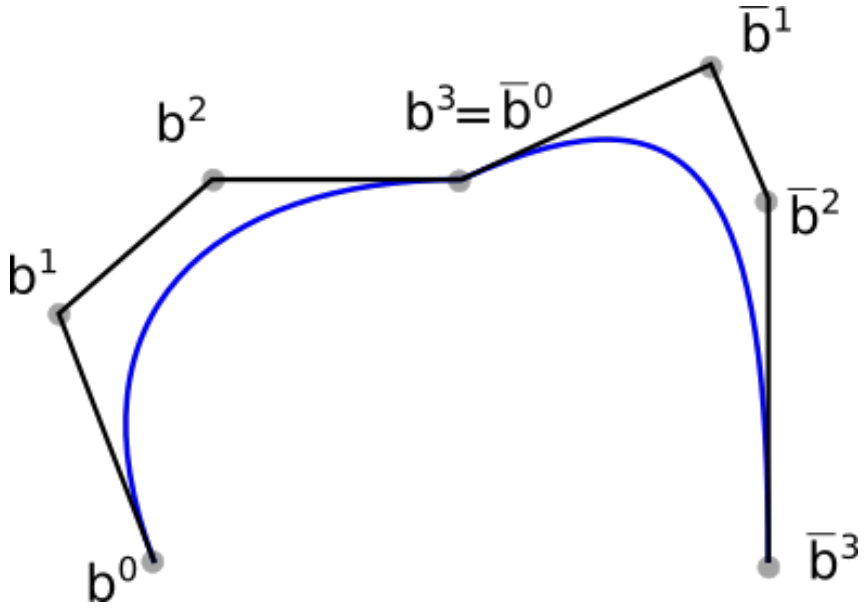

Figure 4.4 – Bézier patch of two curves with C^1 continuity

The most general case of Bézier patches is joining curves of different degrees and parametric spaces together. To develop the conditions necessary to guarantee C^0 and C^1 for this patch, joining curves of the same degree and parametric spaces will be performed.

4.2.1 Joining curves of the same degree and different parametric intervals

Let A and B be cubic Bézier curves to be joined with C^1 continuity. Each curve is defined by 4 control points, $\mathbf{b}_A = (\mathbf{b}_0, \mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3)$ for the first curve and $\bar{\mathbf{b}}_B = (\bar{\mathbf{b}}_0, \bar{\mathbf{b}}_1, \bar{\mathbf{b}}_2, \bar{\mathbf{b}}_3)$ for the second. The parametric space will be $u_a \leq u \leq u_b$, so that $u_b - u_a = \delta$ for the first curve and $u_b \leq u \leq u_c$ with $u_c - u_b = \delta'$. The parametric space for the patch is $u_a \leq u \leq u_c$.

To impose C^0 continuity, the last control point of curve A is coincident with the first control point of curve B, as shown in Figure 4.4.

But the slope of the two Bézier curves may still not be the same at the join point for both curves. To determine the relationship between the position of the control points and C^1 continuity between two Bézier curves of the same degree, Eq. (4.5) will be used at $t = u_b$, i.e., at the join point:

$$\frac{d\mathbf{P}}{dt}(t = u_b) = \frac{d\bar{\mathbf{P}}}{dt}(t = u_b) \tag{4.7}$$

Writing the derivative in terms of $t' = \frac{t-u_a}{u_b-u_a}$, which gives $\frac{dt'}{dt} = \frac{1}{b-a} = \frac{1}{\delta}$. Applying this to Eq. (4.7) gives:

$$\frac{d\mathbf{P}}{dt}(t) = \frac{1}{\delta}\frac{d\mathbf{P}}{dt}(t') \tag{4.8}$$

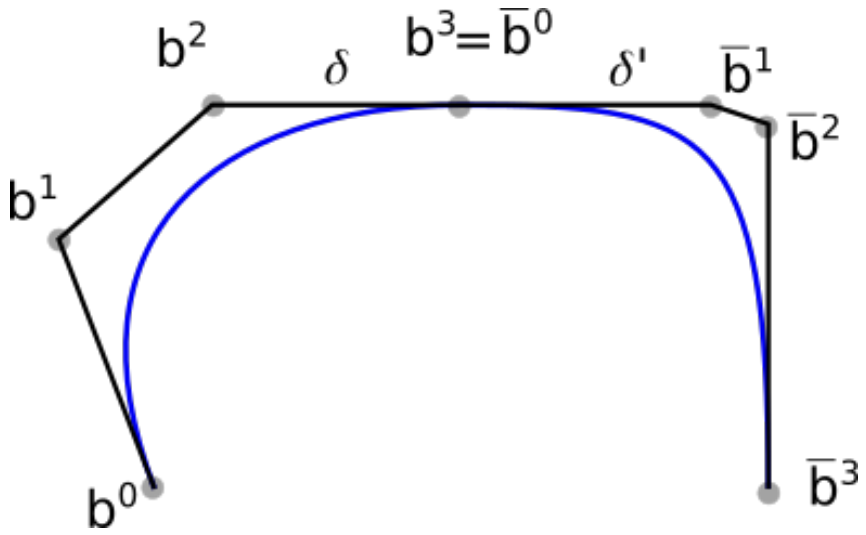

Figure 4.5 – Bézier patch of two curves of the same degree with C^1 continuity

Applying the same process for curve B and applying Eq. (4.8), the following relationship between control points of the curves to guarantee C^1 continuity:

$$\mathbf{b}_3 = \frac{\delta'}{\delta + \delta'}\mathbf{b}_2 + \frac{\delta}{\delta + \delta'}\bar{\mathbf{b}}_1, \tag{4.9}$$

which shows that the condition for continuity is that the control points before and after the join point must be in a straight line. The proportions between the distances of consecutive points must also be δ over δ', as shown in Figure 4.5.

4.2.2 Joining curves of different degrees and same parametric intervals

Two curves of different degrees and, therefore, number of control points can be achieved in the same manner. Consider a cubic and quadratic Bézier curves,

$$\frac{d\mathbf{P}}{dt}(t = u_b) = \frac{d\bar{\mathbf{P}}}{dt}(t = u_b) \tag{4.10}$$

$$\frac{d\mathbf{P}}{dt}(t) = \frac{1}{\delta}\frac{d\mathbf{P}}{dt}(t') \tag{4.11}$$

$$\mathbf{b}_3 = \frac{n}{m + n}\mathbf{b}_2 + \frac{m}{m + n}\bar{\mathbf{b}}_1, \tag{4.12}$$

which shows that the condition for continuity is that the control points before and after the join point must be in a straight line. The proportions between the distances of consecutive points must also be δ over δ', as shown in Figure 4.6.

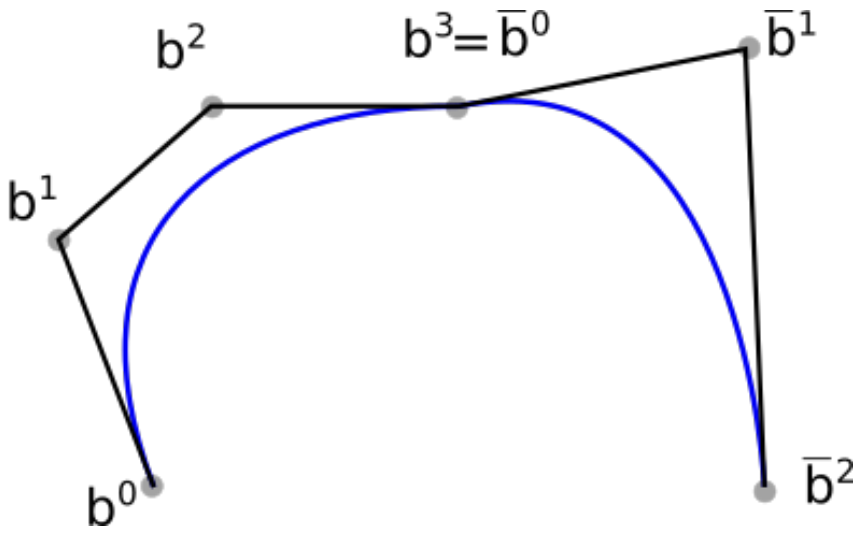

Figure 4.6 – Bézier patch of two curves of the same degree with C^0 continuity

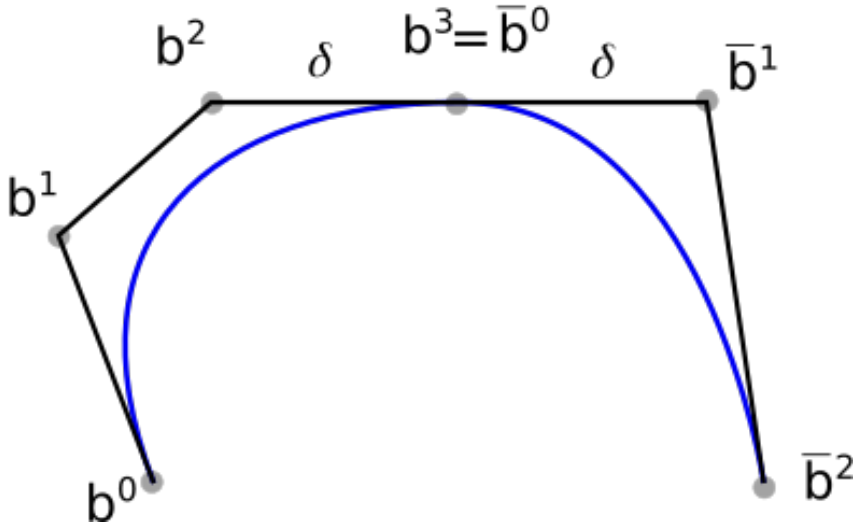

Figure 4.7 – Bézier patch of two curves of the same degree with C^1 continuity

4.2.3 Joining curves of different degrees and parametric intervals

Two curves of different degrees and, therefore, number of control points can be achieved in the same manner. Consider a cubic and quadratic Bézier curves,

$$\frac{d\mathbf{P}}{dt}(t = u_b) = \frac{d\bar{\mathbf{P}}}{dt}(t = u_b) \tag{4.13}$$

$$\frac{d\mathbf{P}}{dt}(t) = \frac{1}{\delta}\frac{d\mathbf{P}}{dt}(t') \tag{4.14}$$

$$\mathbf{b}_3 = \frac{m\delta'}{m\delta' + n\delta}\mathbf{b}_2 + \frac{n\delta}{m\delta' + n\delta}\bar{\mathbf{b}}_1, \tag{4.15}$$

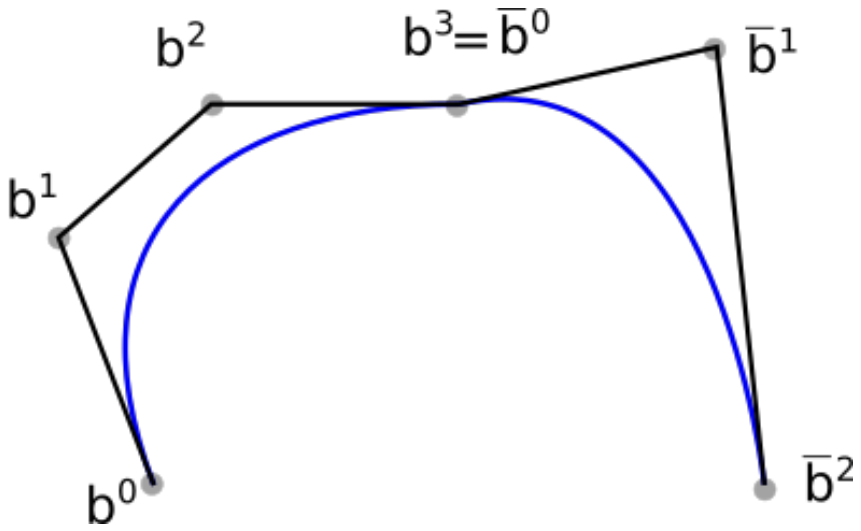

Figure 4.8 – Bézier patch of two curves of the same degree with C^0 continuity

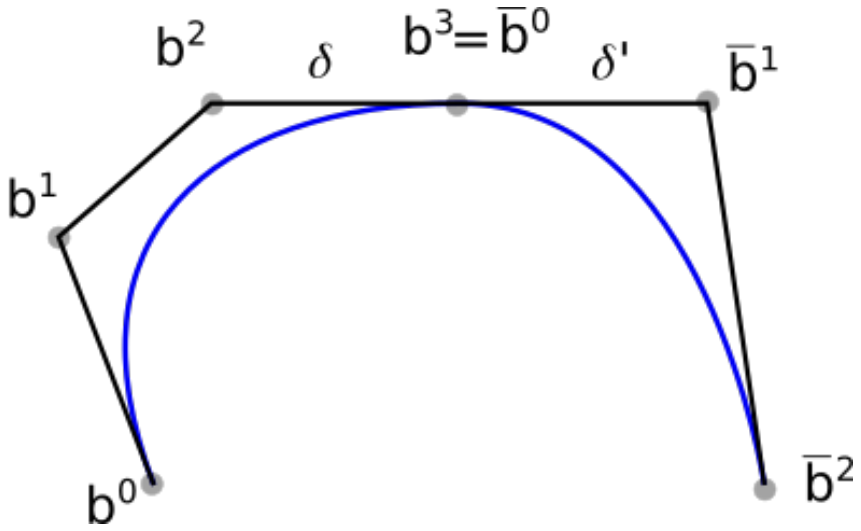

Figure 4.9 – Bézier patch of two curves of the same degree with C^1 continuity

which shows that the condition for continuity is that the control points before and after the join point must be in a straight line. The proportions between the distances of consecutive points must also be δ over δ', as shown in Figure 4.9.

4.3 B-Splines and NURBS

B-splines are a generalization of Bézier curves. It consists of one or more Bézier curve segments and a continuity mechanism between them. This curve is dependent on some interpolation or approximation scheme to define the relationship between the curve and control points. This scheme is provided by the basis function, if the Bernstein basis is used, the result is a Bézier curve, but there are other alternatives for basis functions and there are some limitations which arises from using the Bernstein basis, mainly:

1. The order of the resulting polynomial that defines the curve is dependent on the number of control points

2. The Bernstein basis has a global behaviour, which means that any change in a control point is felt throughout the curve, so that local change is impossible

In the B-Spline, each control point, also called DeBoor point as to differentiate from Bézier control points, defines and influences only one curve segment. The curve can be described as

$$\mathbf{T}(t) = \sum_{i=1}^{n+1} \mathbf{d}_i N_{i,k}(t) = \mathbf{d}^T \mathbf{N}(t) \ , \ t_{min} \leq t \leq t_{max}, 2 \leq k \leq n+1, \tag{4.16}$$

where $\mathbf{d}$ are the DeBoor points and $N_{i,k}$ are the basis functions of order k and degree $p = k - 1$. This functions are not defined explicitly and are calculated recursively using

$$N_{i,k}(t) = \begin{cases} 1, \text{ if } u_i \leq t \leq u_{i+1} \\ 0, \text{ otherwise} \end{cases} . \tag{4.17}$$

This can also be written as

$$N_{i,k}(t) = \frac{(t - u_i)N_{i,k-1}(t)}{u_{i+k-1} - u_i} + \frac{(u_{i+k-1} - t)N_{i+1,k-1}(t)}{u_{i+k} - u_{i+1}}, \tag{4.18}$$

where the values of u_i are from the known vector $\mathbf{U} = \{u_1 u_2 ... u_k\}$. In this vector, the relationship $u_i \leq u_{i+1}$ is always obeyed.

Once B-Splines have been introduced, it's possible to define a curve by a knot vector $\mathbf{U} = \{u_1 u_2 ... u_k\}$, a set of control points $\mathbf{B}$ and a set o rational basis functions $\mathbf{R} = \{R_1 R_1 ... R_k\}$ as

$$C(t) = \sum_{k=1}^{n} \mathbf{P_k} R_k(t). \tag{4.19}$$

These basis functions are defined as

$$R_k(t) = \frac{w_k N_k(t)}{W(t)}, \tag{4.20}$$

where $W(t)$ is a weight function defined by

$$W(t) = \sum_{B=1}^{n} w_b N_b(t) \tag{4.21}$$

and w_b is the weight corresponding to the bth basis function.

4.4 Bézier extraction from NURBS

It's possible to describe every NURBS curve as a patch of Bézier curves. The process of obtaining this Bézier patch representation is called Bézier extraction (BORDEN et al., 2011). There are some benefits to formulating the analysis based on Bézier curves

other than NURBS curves, but in the context of this work, this was done mainly to establish a well-defined element from which the collocation points will be defined. NURBS and B-Splines are representation of Bézier curve patches, blended in a continuous way, this approach didn't introduce any errors to the final curve and its continuity. A similar approach using Bézier decomposition has already been applied succesfully (PEAKE; TREVELYAN; COATES, 2013; PEAKE; TREVELYAN; COATES, 2015).

4.4.1 Knot insertion

Knot insertion is the process of adding a new knot into a knot vector, which may be done with no geometric or parametric properties of the curve. Let $\mathbf{U} = \{u_1 u_2 ... u_k\}$ to which a new knot will be inserted. The new knot $\bar{u} \in \{u_i, u_{i+1}\}$ with $i > p$ will define $m = n + 1$ control points, given by:

$$\bar{\mathbf{P}}_A = \begin{cases} \mathbf{P_1}, & A = 1, \\ \alpha_A \mathbf{P}_A + (1 - \alpha_A)\mathbf{P}_{A-1}, & 1 < A < m, \\ \mathbf{P}_n, & A = m, \end{cases} \tag{4.22}$$

where

$$\alpha_A = \begin{cases} 1, & 1 \leq A \leq k - p, \\ \frac{\bar{u} - u_A}{u_{A+p} - u_A}, & k - p + 1 \leq A \leq k, \\ 0, & A \geq k + 1. \end{cases} \tag{4.23}$$

Each time a new knot is inserted, the continuity of the basis function is reduced by one. Multiple knots may be inserted, however, by choosing the knew knots according to Eqs. (4.22) and (4.23) the continuity of the curve as a whole is preserved.

4.4.2 The Bézier extraction operator

Bézier patch may be extracted from a NURBS curve using the Bézier extraction operator. The first step is to perform a Bézier decomposition on the B-Spline, which is repeating all interior knots of a knot vector $\mathbf{U}$ until each has a multiplicity of $p + 1$. This results in a Bézier patch with repeated control points at each join. In this work, the Bézier decomposition is performed until a multiplicity of p is achieved, so that neighboring Bézier elements will share control points. It's important to note that this process has almost no effect on computational cost, as we are projecting to a smooth, continuous basis.

To illustrate the extraction procedure, we will introduce a cubic B-Spline, which can be in projected space, from which the Bézier patch will be extracted. This curve is adapted from BORDEN et al. (2011), the knot vector is $\mathbf{U} = [0, 0, 0, 0, 1, 2, 3, 4, 4, 4, 4]$ and the DeBoor points $\mathbf{d}$ are shown in Figure 4.10.

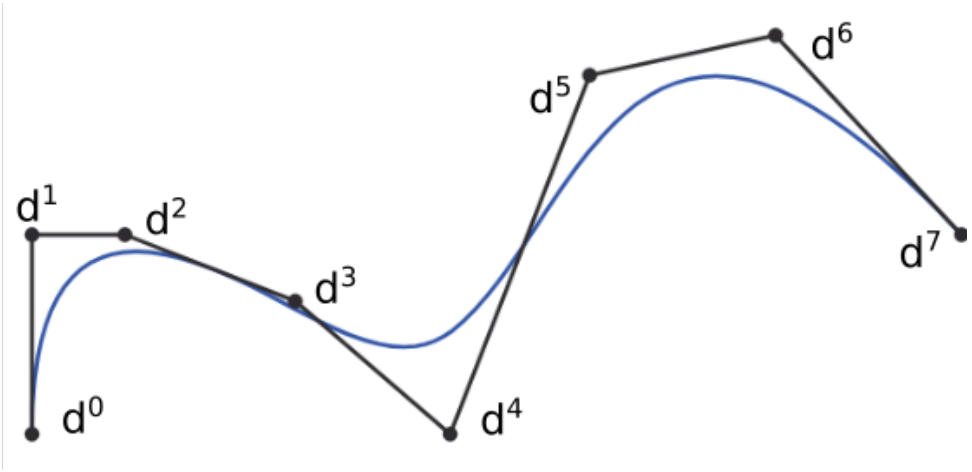

Figure 4.10 – B-Spline with its DeBoor points.

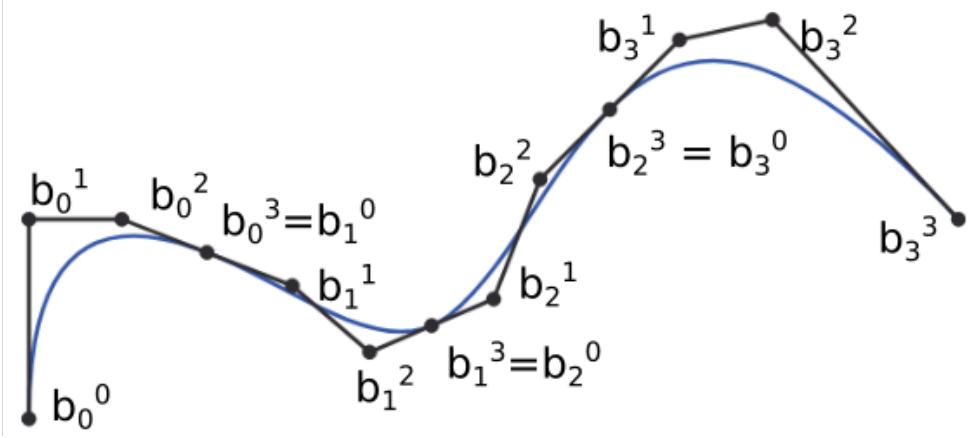

Figure 4.11 – Bézier patch with control points extracted from the B-Spline.

The Bézier decomposition is applied by repeated knot insertion on all interior knots until their multiplicity is p, which is to say until the multiplicity of each knot is equal to the degree of the curve. As Eqs. (4.22) and (4.23) are used to compute the new knots, the continuity of the curve is unchanged, eventhough the resulting basis is decomposed into a set of C^0 Bézier elements. Each element corresponds to a knot span in the original knot vector, which is $\mathbf{U} = [0,0,0,0,1,1,1,2,2,2,3,3,3,4,4,4,4]$ for the decomposed curve, shown in Figure 4.11.

This process may be performed using the Bézier extraction operator $\mathbf{C}$, which will be performed on an arbitrary B-Spline curve with knot vector $\mathbf{U} = [u_0, u_1, ..., u_{n+p}]$ and control points $\mathbf{P} = P_i$, $i = 0, 1, 2, ..., n$. The knots that have to be inserted to produce the Bézier decomposition are $[\bar{u}_1, \bar{u}_1, ..., \bar{u}_m]$. Define α_A^j, $A = 0, 1, ..., n + j - 1$, to be the A_{th} alpha from Eq. (4.23), then the extraction operator $\mathbf{C}^j \in \mathbb{R}^{(n+j-2)\times(n+j-1)}$:

$$\mathbf{C}^j = \begin{bmatrix} \alpha_0 & 1-\alpha_1 & 0 & ... & & & 0 \\ 0 & \alpha_1 & 1-\alpha_2 & 0 & ... & & 0 \\ 0 & 0 & \alpha_2 & 1-\alpha_3 & 0 & ... & 0 \\ ... & & & & & & \\ 0 & ... & & & 0 & \alpha_{(n+j-2)} & 1-\alpha_{(n+j-1)} \end{bmatrix}.$$

From Eq. (4.22), the set of new control points $\bar{\mathbf{P}}^2$ will be created from the first step $\bar{\mathbf{P}}^1 = \mathbf{P}$ will be:

$$\bar{\mathbf{P}}^{j+1} = (\mathbf{C}^j)^T \bar{\mathbf{P}}^j. \tag{4.24}$$

The Bézier elements will be obtained by using the set of control points $\bar{\mathbf{P}}^{m+1}$, so by setting $\bar{\mathbf{P}}^b = \bar{\mathbf{P}}^{m+1}$,

$$\bar{\mathbf{P}}^b = \mathbf{C}^T \bar{\mathbf{P}}^j, \tag{4.25}$$

the final representation of the control points is defined. So, from Eq. (4.16) and given the set of Bernstein basis functions defined by the final knot vector $\mathbf{B}_A(t)$, $A = 1, 2, ..., n + m - 1$, one obtains:

$$T(t) = (\mathbf{P}^b)^T \mathbf{B}(t) = (\mathbf{C}^T \mathbf{P}^T)\mathbf{B}(t) = \mathbf{P}^T \mathbf{C}\mathbf{B}(t) = \mathbf{P}^T \mathbf{N}(t), \tag{4.26}$$

but since $\mathbf{P}$ is arbitrary, a new basis and linear operator have been constructed such that:

$$\mathbf{N}(t) = \mathbf{C}\mathbf{B}(t), \tag{4.27}$$

which is called the *Bézier extraction operator*. All the information required to perform the extraction is the knot vector, i.e., the process does not depend on the control points or basis functions. Therefore, this procedure may be applied directly to NURBS curves.

Applying Eq. (4.4.2) to the B-Spline defined before, one obtains:

$$\begin{Bmatrix} N_1 \\ N_2 \\ N_3 \\ N_4 \\ N_5 \\ N_6 \\ N_7 \end{Bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & \frac{1}{2} & \frac{1}{4} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{7}{12} & \frac{2}{3} & \frac{1}{3} & \frac{1}{6} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{6} & \frac{1}{3} & \frac{2}{3} & \frac{2}{3} & \frac{2}{3} & \frac{1}{3} & \frac{1}{6} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{6} & \frac{1}{3} & \frac{2}{3} & \frac{7}{12} & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{4} & \frac{1}{2} & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{Bmatrix} B_1 \\ B_2 \\ B_3 \\ B_4 \\ B_5 \\ B_6 \\ B_7 \\ B_8 \\ B_9 \\ B_{10} \\ B_{11} \\ B_{12} \\ B_{13} \end{Bmatrix}$$

This is the global extraction operator, but in practice this matrix don't have to be computed entirely, as a localized extraction operator may be used to obtain the following element extraction operators:

$$\begin{Bmatrix} N_1 \\ N_2 \\ N_3 \\ N_4 \end{Bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & \frac{1}{2} & \frac{1}{4} \\ 0 & 0 & \frac{1}{2} & \frac{7}{12} \\ 0 & 0 & 0 & \frac{1}{6} \end{bmatrix} \begin{Bmatrix} B_1 \\ B_2 \\ B_3 \\ B_4 \end{Bmatrix},$$

$$\begin{Bmatrix} N_2 \\ N_3 \\ N_4 \\ N_5 \end{Bmatrix} = \begin{bmatrix} \frac{1}{4} & 0 & 0 & 0 \\ \frac{7}{12} & \frac{2}{3} & \frac{1}{3} & \frac{1}{6} \\ \frac{1}{6} & \frac{1}{3} & \frac{2}{3} & \frac{2}{3} \\ 0 & 0 & 0 & \frac{1}{6} \end{bmatrix} \begin{Bmatrix} B_4 \\ B_5 \\ B_6 \\ B_7 \end{Bmatrix},$$

$$\begin{Bmatrix} N_3 \\ N_4 \\ N_5 \\ N_6 \end{Bmatrix} = \begin{bmatrix} \frac{1}{6} & 0 & 0 & 0 \\ \frac{2}{3} & \frac{2}{3} & \frac{1}{3} & \frac{1}{6} \\ \frac{1}{6} & \frac{1}{3} & \frac{2}{3} & \frac{7}{12} \\ 0 & 0 & 0 & \frac{1}{4} \end{bmatrix} \begin{Bmatrix} B_7 \\ B_8 \\ B_9 \\ B_{10} \end{Bmatrix},$$

$$\begin{Bmatrix} N_4 \\ N_5 \\ N_6 \\ N_7 \end{Bmatrix} = \begin{bmatrix} \frac{1}{6} & 0 & 0 & 0 \\ \frac{7}{12} & \frac{1}{2} & 0 & 0 \\ \frac{1}{4} & \frac{1}{2} & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{Bmatrix} B_{10} \\ B_{11} \\ B_{12} \\ B_{13} \end{Bmatrix}.$$

It can be seen that the localized extraction matrices from Eqs. (4.4.2) to (4.4.2) are obtained from Eq. (4.4.2). Equation (4.4.2) now can be written as:

$$\mathbf{N}^e(t) = \mathbf{C}^e\mathbf{B}^e(t). \tag{4.28}$$

4.5 Isogeometric Analysis using NURBS and BEM

Once the geometry has been defined using a NURBS, it is possible to create a discretization methodology for the BEM. Consider a single NURBS function consisting of the control points, weights and knot vector. This function is decomposed into E non-overlapping elements, which are rational Bézier patches of order p. Each patch is created by mapping the NURBS into rational Bézier functions used for the geometry representation.

The boundary integral equation can be written as

$$\phi = \int_\Gamma \frac{\partial \phi}{\partial \mathbf{n}} \phi^* d\Gamma - \int_\Gamma \phi \frac{\partial \phi^*}{\partial \mathbf{n}} d\Gamma. \tag{4.29}$$

The analytical geometry on each element Γ_e is given by

$$\Gamma_e = \{R^e_{i,k}(t) : t \in (0,1)\}. \tag{4.30}$$

The behaviour of the velocity potential ϕ_e and its normal derivative over element Γ_e are also mapped using $R^e_{i,k}(t)$, as

$$\phi^e(t) = \sum_{i=1}^{p} R^e_{i,k}(t)\bar{\phi^e_i}, \tag{4.31}$$

$$q^e(t) = \sum_{i=1}^{p} R^e_{i,k}(t)\bar{q^e_i}. \tag{4.32}$$

The elements consist of a grid of $(p+1)\times(p+1)$ control potentials, $\bar{\phi^e_t}$ and $R^e_t(t)$ are their associated rational Bézier functions. The boundary integral equation is now written as

$$\phi = \int_\Gamma \sum_{i=1}^{p} R^e_{i,k}(t)\bar{q^e_i}\phi^* d\Gamma - \int_\Gamma \sum_{i=1}^{p} R^e_{i,k}(t)\bar{\phi^e_i}\frac{\partial \phi^*}{\partial \mathbf{n}} d\Gamma \tag{4.33}$$

As $\bar{\phi^e_i}$ and $\bar{q^e_i}$ are nodal values, they can be written out of the integral, which yields

$$c\phi = \sum_{i=1}^{p} \bar{q^e_i} \int_\Gamma R^e_{i,k}(t)\phi^* d\Gamma - \sum_{i=1}^{p} \bar{\phi^e_i} \int_\Gamma R^e_{i,k}(t)\frac{\partial \phi^*}{\partial \mathbf{n}} d\Gamma \tag{4.34}$$

The boundary is parametrized by t:

$$c\phi = \sum_{i=1}^{p} \bar{q^e_i} \int_{t_{min}}^{t_{max}} R^e_{i,k}(t)\phi^* \frac{d\Gamma}{dt} dt - \sum_{i=1}^{p} \bar{\phi^e_i} \int_{t_{min}}^{t_{max}} R^e_{i,k}(t)\frac{\partial \phi^*}{\partial \mathbf{n}}\frac{d\Gamma}{dt} dt \tag{4.35}$$

where $\frac{d\Gamma}{dt} = \sqrt{(\frac{dx(t)}{dt})^2 + (\frac{dy(t)}{dt})^2}$ is the Jacobian needed to perform the domain change.

Each basis function is not-null only in an unique interval. The influence of the control point is defined by this intersectant interval and is contained between t_i and t_{i+1}. It is possible, then, to reduce the boundary integral equation to the non-null intervals, which define the influence domain:

$$c\phi = \sum_{i=1}^{p} \bar{q^e_i} \int_{t_i}^{t_{i+k}} R^e_{i,k}(t)\phi^* \frac{d\Gamma}{dt} dt - \sum_{i=1}^{p} \bar{\phi^e_i} \int_{t_i}^{t_{i+k}} R^e_{i,k}(t)\frac{\partial \phi^*}{\partial \mathbf{n}}\frac{d\Gamma}{dt} dt \tag{4.36}$$

To evaluate the integrals numerically, another change is necessary. The integration interval must be regularized to $[-1, 1]$ to perform the Gaussian quadrature:

$$c\phi = \sum_{i=1}^{p} \bar{q^e_i} \int_{-1}^{1} R^e_{i,k}(t)\phi^* \frac{d\Gamma}{dt}\frac{dt}{d\xi} d\xi - \sum_{i=1}^{p} \bar{\phi^e_i} \int_{-1}^{1} R^e_{i,k}(t)\frac{\partial \phi^*}{\partial \mathbf{n}}\frac{d\Gamma}{dt}\frac{dt}{d\xi} d\xi \tag{4.37}$$

where $\frac{dt}{d\xi} = \frac{t_{i+k}-t_i}{2}$.

4.5.1 Collocation points

Greville abscissa is the point in the domain above which the control point has maximum influence and is defined as the mean of d nodes.

$$\lambda_i = \frac{1}{d}(t_{i+1} + t_{i+2} + ... + t_{i+d}) \tag{4.38}$$

These points are unique, located at the boundary and there are as many as control points. When the curve is smooth, they are good candidates for collocation points, but when there are discontinuities, such as corners, a position shift of the first and last points is necessary

$$\lambda_1 = \lambda_1 + \beta(\lambda_2 - \lambda_1), \tag{4.39}$$

$$\lambda_n = \lambda_n + \beta(\lambda_n - \lambda_{n-1}), \tag{4.40}$$

where $\beta = 0.5$ yields the best results.

5 FAST BEM

In this chapter, the topics of hierchical matrices and adaptive cross-approximation method for approximating integral equation matrices in a low rank form is presented. In section 5.1 a brief introduction to the Hierarchichal matrices and their properties is given. In sections 5.2 and 5.2.1, low rank matrices and their approximation properties are considered. Then, the adaptive cross-approximation method is presented.

5.1 $\mathscr{H}$-Matrices

In this section, the topic of hierarchical matrices is introduced. This is the procedure used to subdivide the original matrix from the integral equation into two groups: *low rank* and *full rank* blocks. The low rank blocks are approximated using a low rank approximation, such as the ACA. Full rank blocks are not approximated and correspond to the solution of the integral equation. The division of the original matrix into clusters will obbey geometrical considerations.

5.1.1 Hierarchical clustering

Low rank approximations can be obtained for low rank matrices, significantly reducing the computational cost of storing and manipulating large matrices. But influence matrices obtained from integral equations have no explicit structure in general. However, low rank blocks can be observed in the BEM influence matrices considering that the integrals of contiguous elements due to a single collocation point are almost identical, especially for high density meshes (BRANCATI, 2010). Some blocks of the influence matrix may present a low rank, which means these blocks are suitable for approximation.

In this work, the influence matrices are subdivided into two blocks *low rank* and *full rank* blocks. The subdivision is based upon a geometrical criterion of the discretised mesh. This division is done by storing the indices of contiguous nodes and elements in a *cluster tree*, creating a basis for the block subdivision and for building the *block tree*. Once this procedure is finished, low rank blocks are determined using an *admission criterion*, based on geometrical considerations.

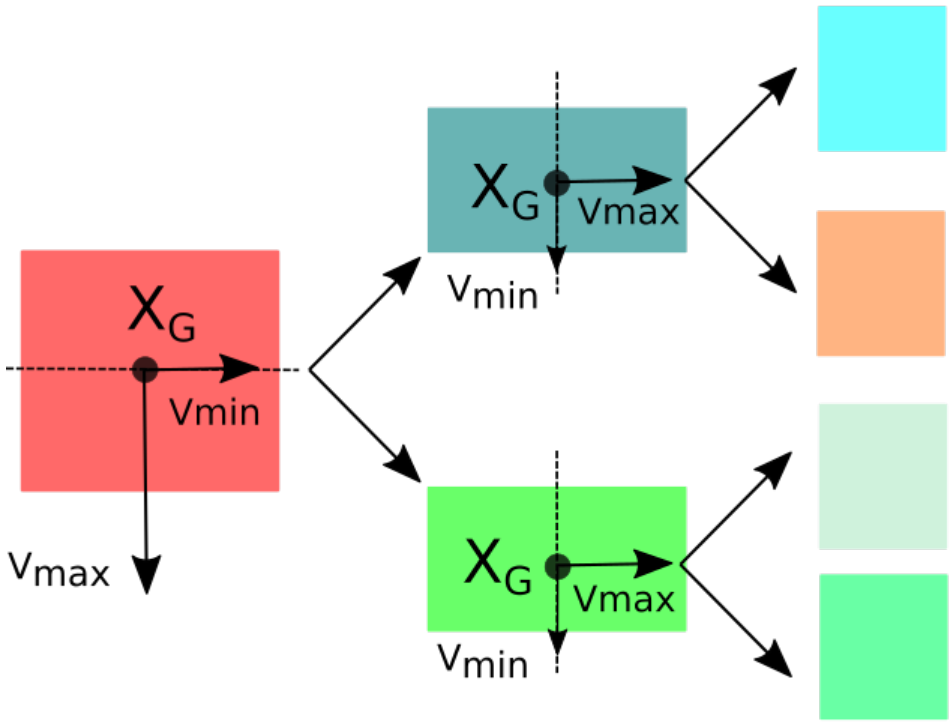

Figure 5.1 – Binary tree

5.1.2 Building the cluster tree

The first step when building the cluster tree is to determine how the subdivision will occur. A first try would be to determine the centroid of the geometrical shape of the model. Considering a collection of N points consisting of $x(i) = [x_1(i), x_2(i), x_3(i)]$, with $i = 1, 2, 3, \ ... \ N$, the centroid $\mathbf{XG} = (XG_1, XG_2, XG_3)$ can be defined as the mean of the values of $\mathbf{x}$.

$$\mathbf{XG}_i = \frac{1}{N}\sum_{j=1}^{N}\mathbf{x}_i(j). \tag{5.1}$$

Now that the centroid has been found, the covariance C of x is determined.

$$\mathbf{C} = \frac{1}{N}\sum_{j=1}^{N}(\mathbf{x}_1(j) - \mathbf{XG}_1)(\mathbf{x}_2(j) - \mathbf{XG}_2)(\mathbf{x}_3(j) - \mathbf{XG}_3). \tag{5.2}$$

Then, a eigenvalue extraction will be performed.

$$\lambda\mathbf{v} = \mathbf{C}\mathbf{v}, \tag{5.3}$$

where λ are the eigenvalues and $\mathbf{v}$ is the eigenvector matrix. One may choose the maximum value of the eigenvalues and select the corresponding eigenvector.

$$\mathbf{v}_{max} = \mathbf{v}(k), \tag{5.4}$$

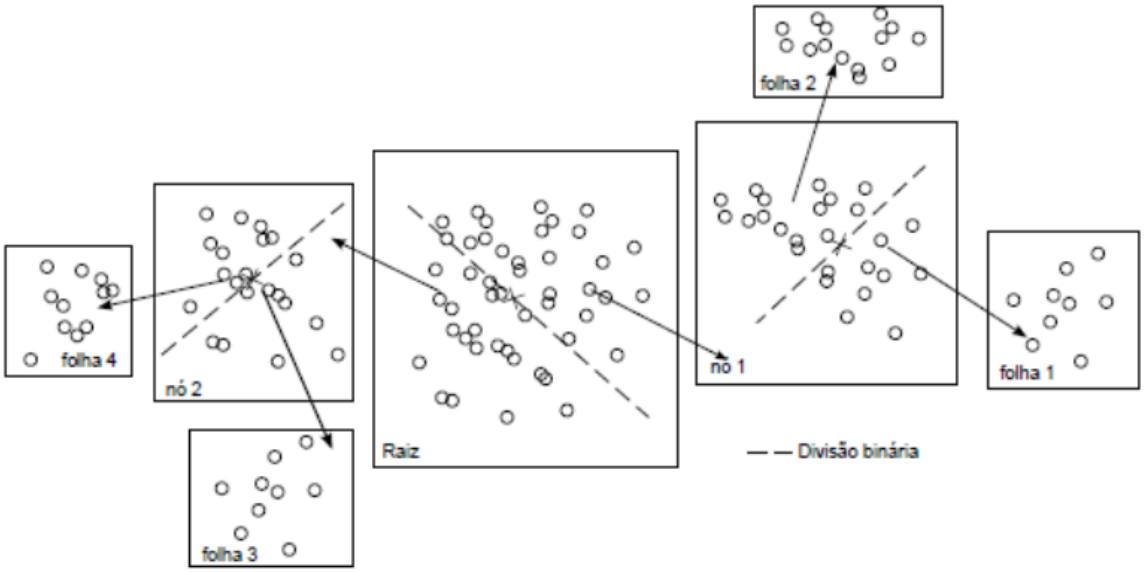

Figure 5.2 – Cluster tree construction

where k is the indice of the maximum eigenvalue.

The direction of this eigenvector is the direction of maximum covariance. One may separate the points perpendicular to this eigenvector from the center of mass by performing the dot product shown as follows.

$$\mathbf{p}_i = (\mathbf{x}_i - \mathbf{XG}) \cdot \mathbf{v}_{max} \tag{5.5}$$

This new variable $\mathbf{p}$ is used to determine whether point i is incorporated in cluster (Cl_1) or cluster (Cl_2). This creates a separation plane which is orthogonal to the eigenvector of the covariance. This plane is situated in the centroid of the geometrical shape of the model. The definition of clusters 1 and 2 are given by:

$$\begin{cases} \text{if } \mathbf{p}_i > 0 \text{ then } \mathbf{x}_i \in Cl_1 \\ \text{if } \mathbf{p}_i < 0 \text{ then } \mathbf{x}_i \in Cl_2 \end{cases} \tag{5.6}$$

This process will be applied recursevely to clusters 1 and 2 until both contain some (small and independent of N) n_{min} prescribed value number of points or less. Every cluster except the last one is named branch. The last one is called a leave. Figure 5.2 shows an schematic of this procedure.

5.1.3 Cluster pairing

To divide the influence matrix, the maximum number of elements that can fit a leaf is defined. The matrix which contains the geometric information about the elements

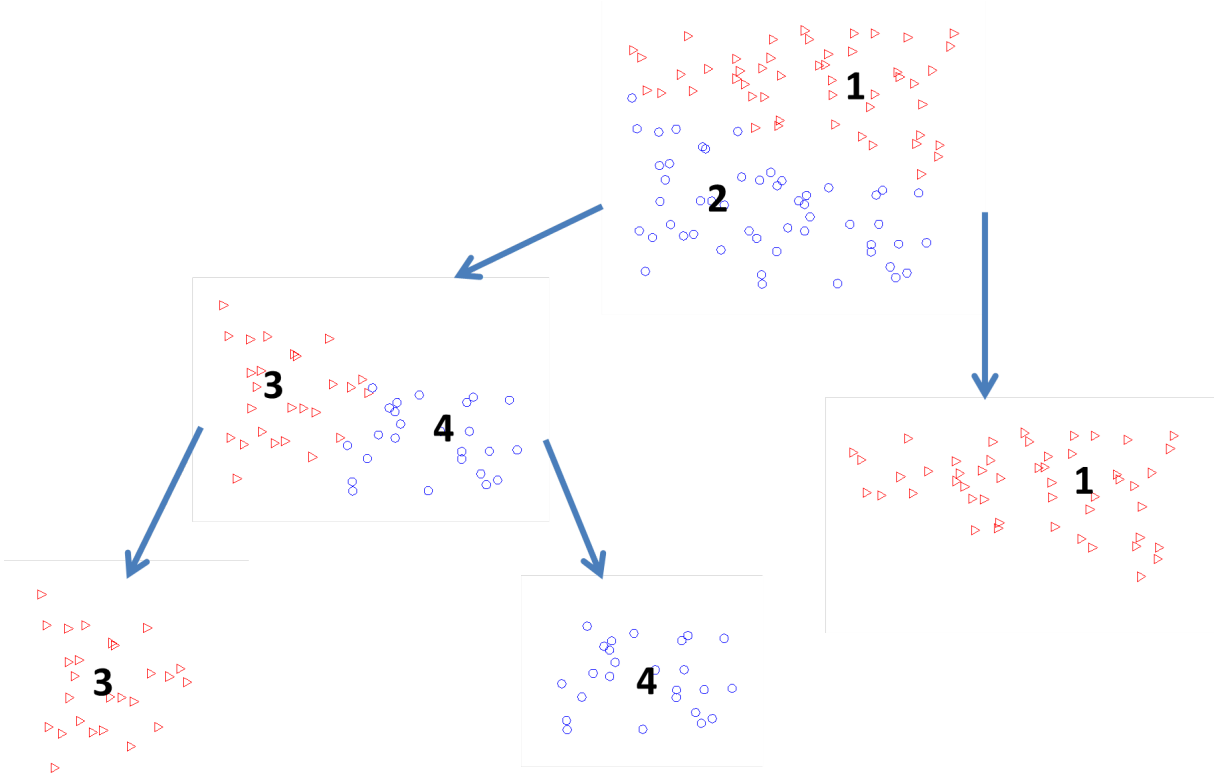

Figure 5.3 – Binary division of randomly distributed points

of the model is used to obtain matrix *Tree*, which contains the indices of the branches and leaves of the *cluster tree*.

Even with the subdivision of the original matrix, it is still possible to pair different leafs of the tree in an efficient manner, so that the biggest possible blocks form the complete matrix.

An example of the division of points is carried out in a set of 100 points distributed in the interval $(x, y) \in [0, 1]$. The division is shown in Figure 5.3.

As we can see from Figure 5.3, the influence between blocks 1 and 2 will probably not be approximated, but the influence between 3 and 1 might. An algorithm for this assessment was developed and consists of three comparisons of the properties of the cluster tree. The center of mass of the block is determined. Now each block is compared. If the distance between the center of mass is greater than an admission criteria, the influence between the blocks is approximated.

If the approximation is successful, the information is stored of which rows and columns are already calculated by a low rank approximation.

5.2 Low rank matrices

The rank of a matrix $\mathbf{A}$ is the number of linear independent lines or columns from $\mathbf{A}$. This also describes the dimension of the vector space generated by its columns or rows. The rank of a matrix is one of its fundamental characteristics. It is possible to describe a

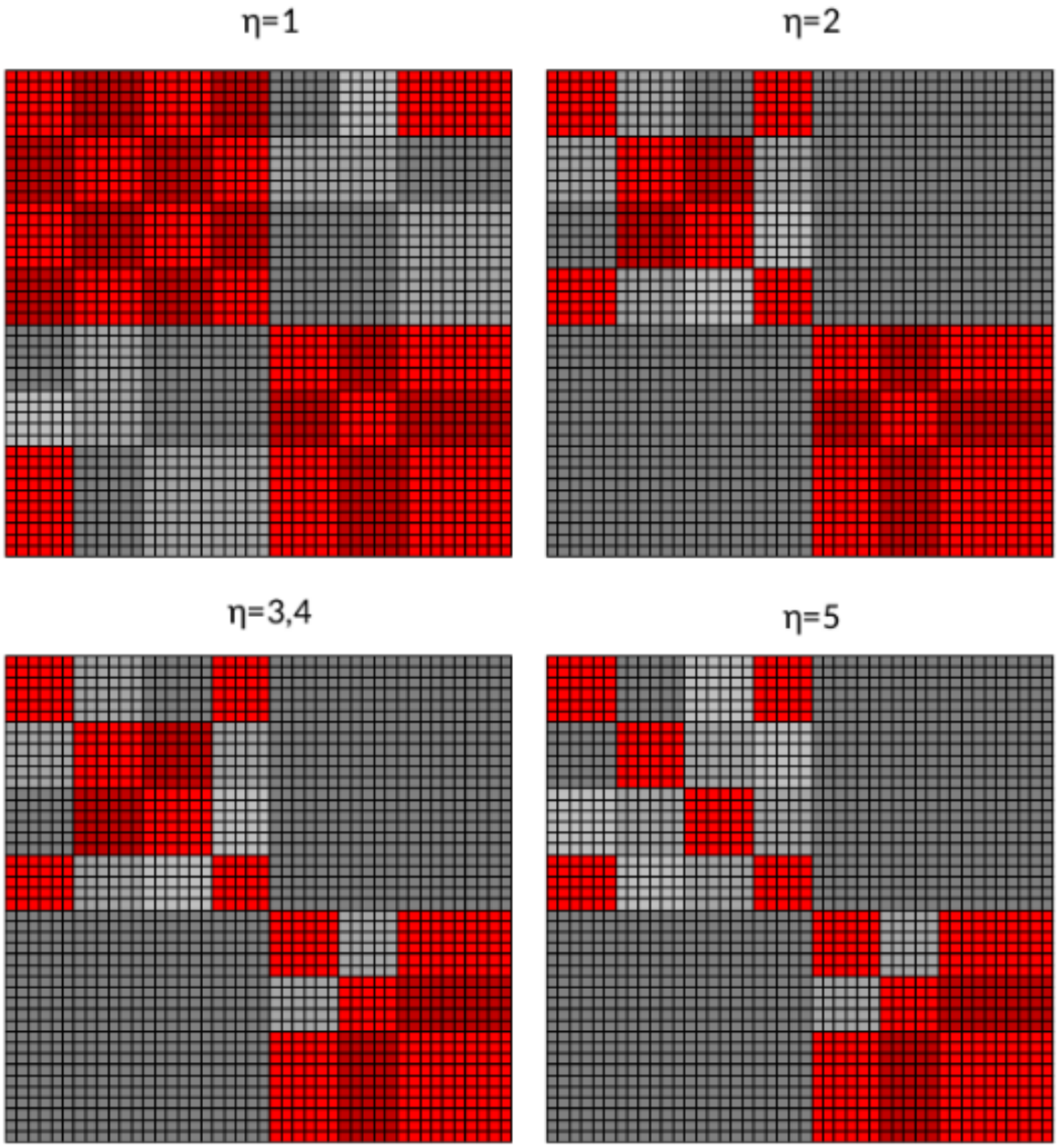

Figure 5.4 – Examples of different admission values, red blocks are approximated

matrix $\mathbf{A} \in \mathbb{R}^{n \times m}$ of rank k using the product of two vectors $\mathbf{U} \in \mathbb{R}^{n \times k}$ and $\mathbf{V} \in \mathbb{R}^{k \times m}$ as

$$\mathbf{A} = \mathbf{UV}, \tag{5.7}$$

$$\mathbf{Ax} = \mathbf{U}(\mathbf{Vx}), \tag{5.8}$$

called outer-product form.

The matrix-vector multiplication of the BEM can be written as shown in equation (5.8). The number of operations necessary for the multiplication $\mathbf{Ax}$ is $2mn$, while the multiplication $\mathbf{U}(\mathbf{Vx})$ requires $2k(m + n - 1)$ operations.

Matrices that have a rank that is relatively small compared with their dimensions

are one of the basic structures for the efficiency of hierarchical matrices (BEBENDORF, 2008). If matrix $\mathbf{A}$ is square, i.e. $\mathbf{A} \in \mathbb{R}^{nxn}$, then the matrix-vector multiplication shown in Eq. (5.8) needs $4k(n - 1/2)$ operations to be evaluated. Matrix $\mathbf{A}$ is considered low rank when the condition in equation (5.9) is satisfied.

$$k \leq \frac{n}{2} \tag{5.9}$$

Equation (5.9) shows the sufficient condition for which the representation shown in equation (5.7) is advantageous. When the rank of a matrix is considerably lower than their dimension, then the matrix is considered to be a low rank matrix.

5.2.1 Low rank approximation (LRA)

The BEM generates fully populated non-symmetric matrices. This results in high computational costs for large scale models, both in storage and computation. The influence matrices which results from integral equations have no explicit structure in general. This means that the matrices are not necessarily low rank matrices. It is possible, however, to approximate portions of this matrix as low rank matrices. This process is known as a hierarchical low rank approximation. This approximation is defined as a minimization problem, in which the approximated matrix contains a lower rank than the original matrix.

Let $\mathbf{A} \in \mathbb{R}^{n \times n}$ be a given matrix having the entries obtained by BEM. Let $\tilde{\mathbf{A}} \in \mathbb{R}^{nxn}$ be an approximation of $\mathbf{A}$ such that the error between $\mathbf{A}$ and $\tilde{\mathbf{A}}$ is described by equation 5.10.

$$||\mathbf{A} - \tilde{\mathbf{A}}||_F \leq \epsilon ||\mathbf{A}||_F \tag{5.10}$$

where $||*||_F$ is the Frobenius norm of variable $*$ and ϵ is the precision of the approximated matrix.

The approximated matrix $\tilde{\mathbf{A}}$ can be obtained using several methods, including the adaptive cross approximation. As an example, the singular value decomposition will be used to ilustrate the process of approximating a matrix $\mathbf{A}$ to $\tilde{\mathbf{A}}$. The singular value decomposition is a factorization that can be perfomed in any real or complex matrix. This decomposition allows the matrix to be described as the product of three other matrices.

$$\mathbf{A} = \mathbf{USV}' \tag{5.11}$$

where $\mathbf{U} \in \mathbb{R}^{nxn}$ is an unitary matrix, $\mathbf{V}'$ is the transpose of matrix $\mathbf{V} \in \mathbb{R}^{nxn}$ and $\mathbf{S} \in \mathbb{R}^{nxn}$ is a diagonal matrix with non-negative real numbers on the diagonal, known as the singular values of matrix $\mathbf{A}$. If the original matrix $\mathbf{A}$ has rank k, matrix $\mathbf{S}$ has k singular values. It is possible to approximate matrix $\mathbf{A}$ using less singular values, which would result in an approximation with lesser rank. The singular value decomposition will

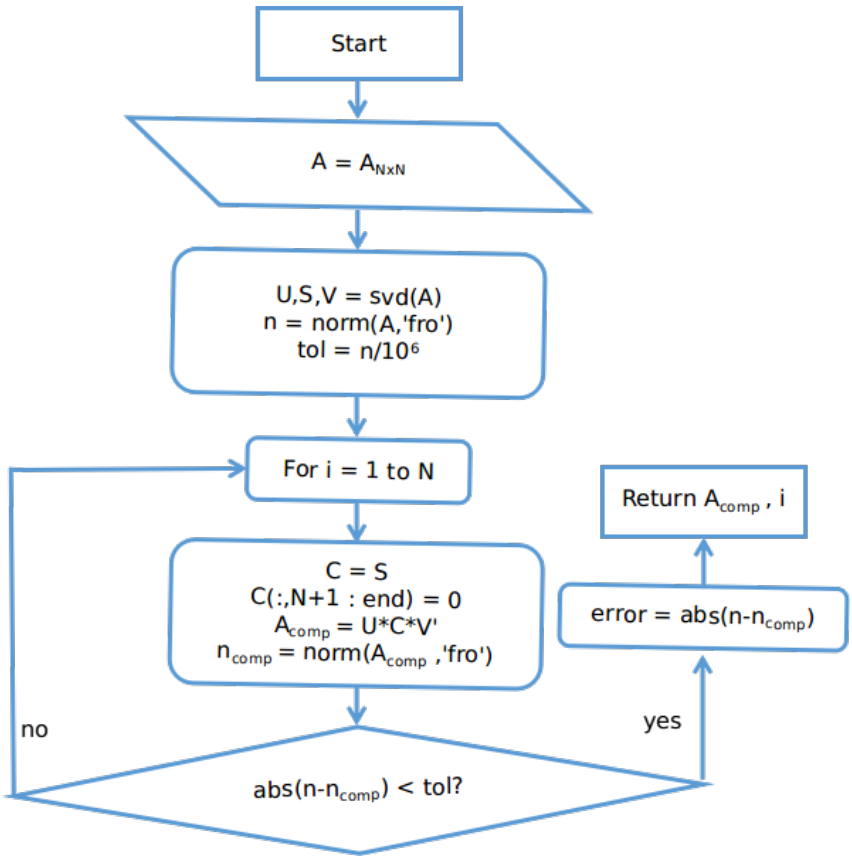

Figure 5.5 – Flowchart for the SVD approximation algorithm

yield the lowest rank approximation for a given error, but the cost of the computation is very high and it is not suitable for large problems.

The SVD algorithm is shown in Figure 5.5. This algorithm uses a function for SVD on matrix $\mathbf{A}$ which outputs matrices $\mathbf{U}, \mathbf{V}$ and $\mathbf{S}$. The Frobenius norm for matrix is obtained. An approximated matrix is obtained by successively adding more terms of the singular values and performing the multiplication with matrices $\mathbf{U}$ and $\mathbf{V}'$. The approximated matrix is compared to the original matrix and, if the absolute error is smaller than a tolerance value, the rank of the approximation is given.

Now that the concept of a low rank approximation of a matrix has been presented, the adaptive cross-approximation can be introduced. The idea behind the method is to approximate the matrix using a single line and column of the original matrix. This new matrix is then tested either compared to the original matrix or to a better approximation of it. This procedure will be further detailed in section 5.3.

The matrices used in this section are not assumed to be low rank matrices. One of the advantages of the adaptive cross-approximation is that if the matrix cannot be approximated by low rank representations, the original matrix is produced.

The main idea behind the adaptive cross-approximation procedures is to use entries from the original matrix to approximate the general form of the matrix. This is done by multiplicating specific lines and columns of the original matrix to approximate the matrix.

5.3 The ACA algorithms

Two different algorithms are presented to approximate the original matrix. The first one is the fully pivoted adaptive cross-approximation algorithm, and it provides an approximation of the original matrix but it needs the original matrix in order to create the approximation. This is not ideal once the original matrix will still have to be stored. Hence, another algorithm is presented, using just a few entries of the original matrix to approximate it. This is the partially pivoted adaptive cross-approximation algorithm.

5.3.1 The ACA fully pivoted algorithm

The fully pivoted adaptive cross-approximation algorithm is presented below. Extensive mathematical research has been developed for the convergence and accuracy of the method, and the following algorithms resulted from those efforts (HACKBUSH, 1999; BORM; GRASEDYCK; HACKBUSH, 2003; BEBENDORF, 2000; BEBENDORF; RJASANOW, 2003).

The first consideration of this algorithm is that the original matrix is stored and can be accessed at any time. The approximated matrix is compared to the original matrix in order to determine the error in the approximation.

The main idea behind the fully pivoted adaptive cross-approximation is that the main behaviour of a low rank matrix can be obtained using only a few entries of the original one. The first step is to select a line and a column from the original matrix. This is done by choosing the line with the maximum absolute value in the first column and the column with the maximum absolute value in the first line. The selected column is then divided by the maximum value of its entries. The diadic product between them is the first approximation of the original matrix.

This approximation is subtracted from the original matrix. The Frobenius norm of the difference is compared with the Frobenius norm of the original matrix times the tolerance of the method. If the first norm has a greater value than the tolerance times the norm of the original matrix, the process is repeated, only now using the difference matrix and not the original to select the line and column.

The algorithm for the ACA fully pivoted method is shown in Figure 5.6.

5.3.2 The ACA partially pivoted algorithm

The partially pivoted adaptive cross-approximation algorithm uses only a few entries of the original matrix to construct the approximation. In this procedure, the first line and column of the original matrix are chosen to start the approximation. The local absolute maximum row value is determined and is used to divide the column. A first

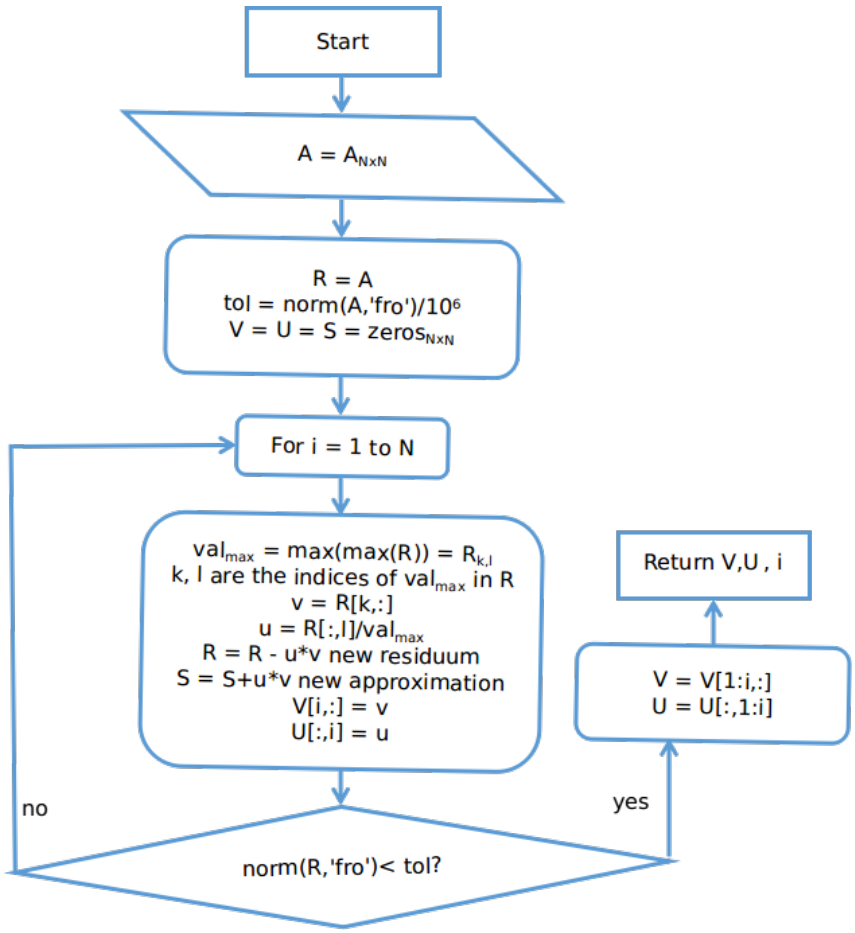

Figure 5.6 – Flowchart for the ACA fully pivoted approximation algorithm

approximation of the matrix is obtained. Now a new line and column is added until the Frobenius norm of the multiplication of the latest row and column has an absolute value lower than some tolerance. Using this algorithm, the complete matrix is not calculated unless no approximation is possible to obtain the required precision. In this case, the full matrix will be calculated with no precision loss.

The approximation will determine the rank k adaptively to precision ϵ. At each new step n, the matrix described by the dyadic product $\mathbf{a}_n\mathbf{b}_n$ is obtained. One way of estimating the error is to consider the new rank 1 approximation obtained in step $n = k+1$

$$||\mathbf{A} - \tilde{\mathbf{A}}^k||_F \leq ||\mathbf{A} - \tilde{\mathbf{A}}^{k-1}||_F \approx ||\tilde{\mathbf{A}}^k - \tilde{\mathbf{A}}^{k-1}||_F \leq ||\mathbf{a}_n\mathbf{b}_n||. \tag{5.12}$$

The relative error can be estimated as

$$\epsilon = \frac{||\mathbf{a}_n\mathbf{b}_n||}{||\tilde{\mathbf{A}}^1||}. \tag{5.13}$$

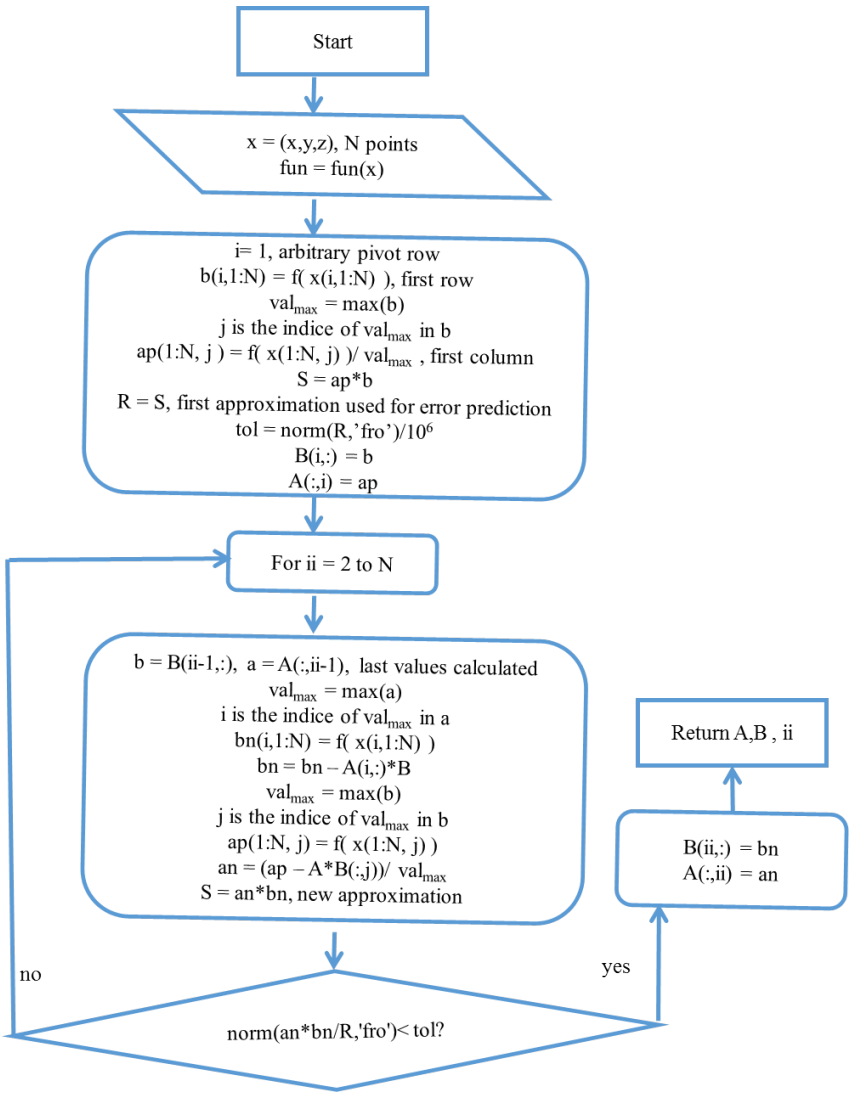

Figure 5.7 – Flowchart for the ACA partially pivoted approximation algorithm

5.3.3 ACA+

The approximated matrix $\tilde{A}$ can be obtained using several methods, including the adaptive cross approximation (ACA) and the singular value decomposition. In previous works (MORGADO, 2017), the ACA was used to approximate the influence matrices, but the authors concluded another implementation of matrix approximation was deemed necessary. This was the result of increasing the number of lines and columns on a procedure dubbed ACA+ (CAMPOS,), which solves representation problems for Dirichlet boundary conditions, but increased the complexity of the algorithm and resulted in higher processing time for building the influence matrix.

This process can be seen in Figure 5.8, in the first step, the first line and column of the matrix are calculated.

It's possible to notice how this procedure would take longer than building the whole matrix, but consumes less memory. Even though not every element of the matrix is stored, some of them are calculated multiple times.

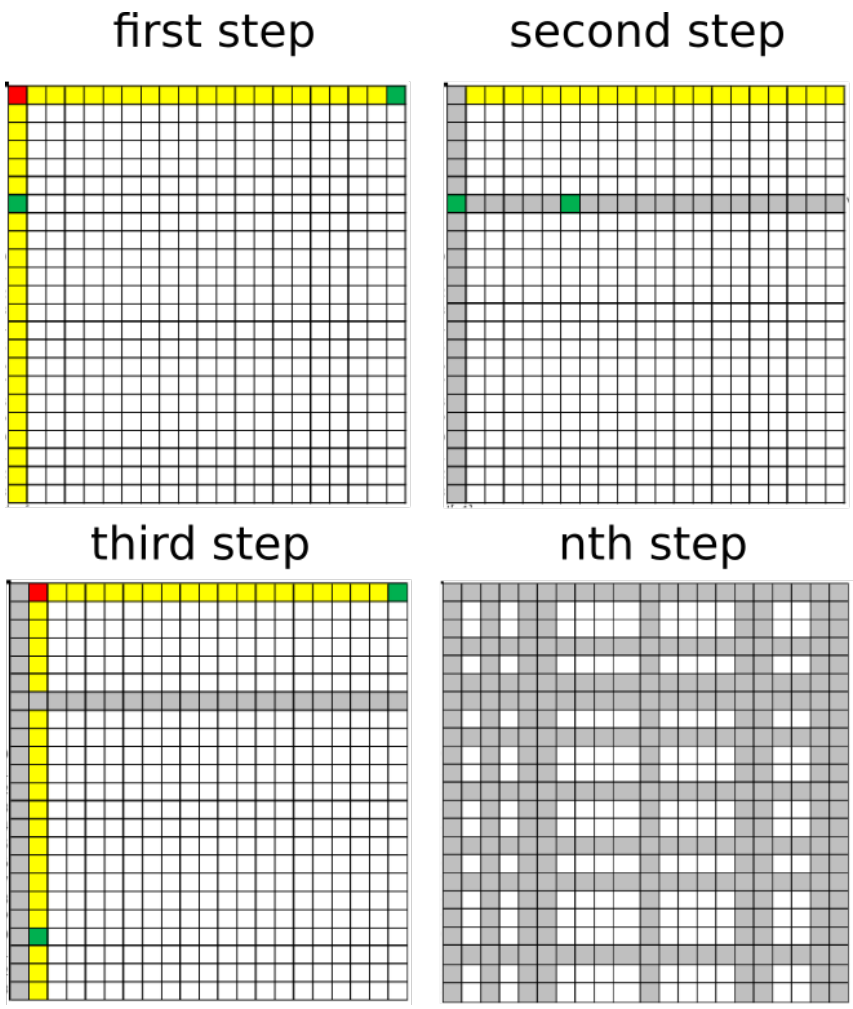

Figure 5.8 – Step by step building of the approximated matrix using the ACA

5.3.4 Numerical considerations of the ACA algorithms

As an example, the Helmholtz equation fundamental solution, given by Eq. (3.8), is used to create a square matrix which is approximated using the singular value decomposition and both versions of the adaptive cross-approximation algorithm

$$\phi^* = -\frac{e^{ikr}}{4\pi r}. \tag{5.14}$$

A two dimensional grid of points is created, forming a square of side $L = 1$. The points of the grid will be considered the field points and a fixed source point is located at $(x_d, y_d) = (0.5, 0.5)$. The number of points necessary to describe the acoustic field generated by this unitary source is N^2, where N is the number of points of the square matrix. By adjusting the value of N, its possible to simulate the kind of approximation that will be performed in the influence matrices of the BEM. The configuration is shown in Figure 5.9.

The processing time to compute approximated matrices using the methods described earlier and the number of values stored for each method are obtained. The results

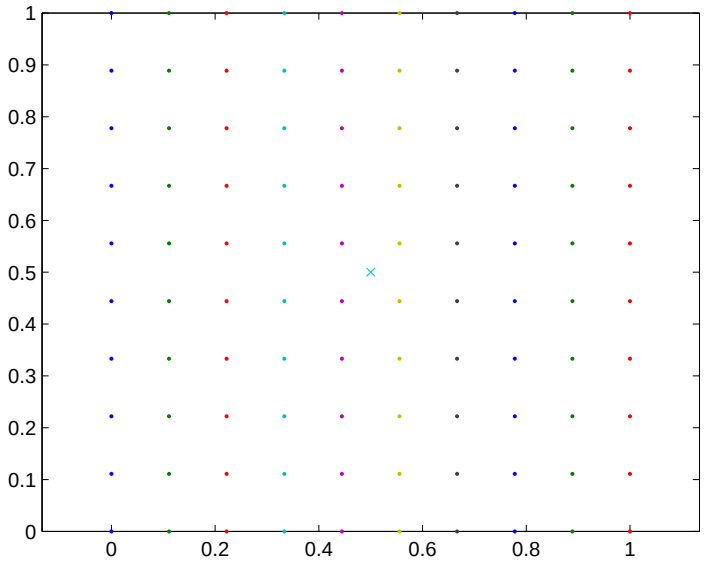

Figure 5.9 – Field points and source point for the algorithms tests

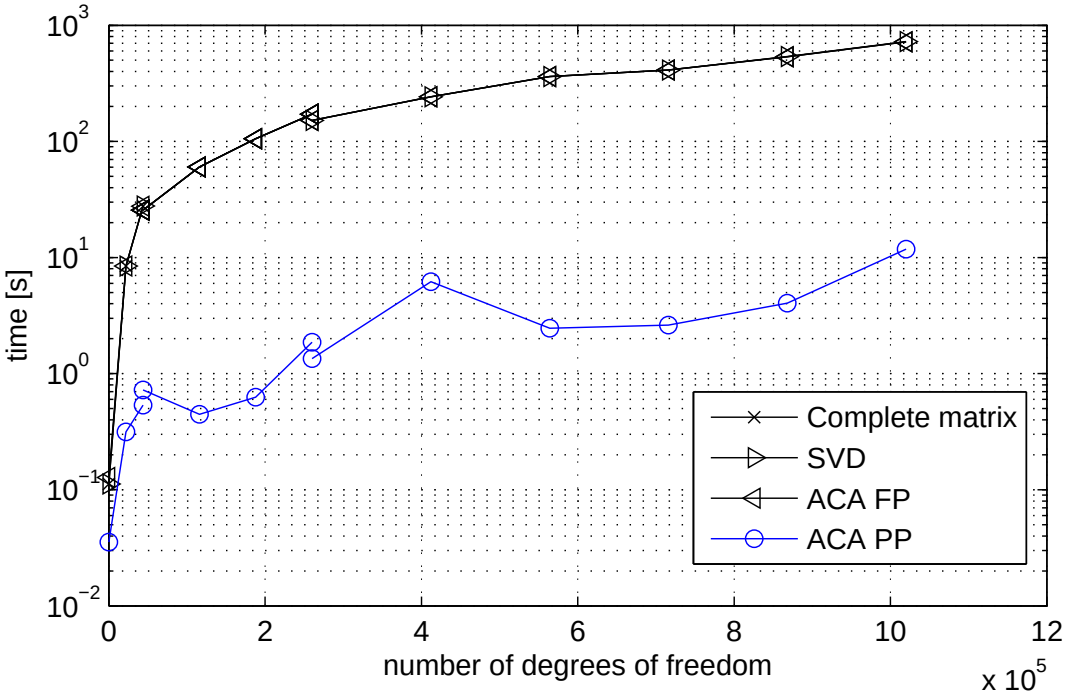

Figure 5.10 – Processing time comparison for the computation of the test matrix and its approximations by SVD, ACA fully pivoted and ACA partially pivoted

are shown in Figures 5.10 and 5.11.

Another curve was added in Figure 5.11 showing the values of $Nlog(N)$, where N^2 is the number of elements of the complete matrix. Its possible to observe that the SVD algorithm was the best at reducing the number of elements necessary to describe the matrix. The ACA fully pivoted algorithm kept close to $Nlog(N)$ and the ACA partially pivoted was the least efficient of the three. However, the speed gained in not having to compute the complete matrix before approximating must be considered. The processing

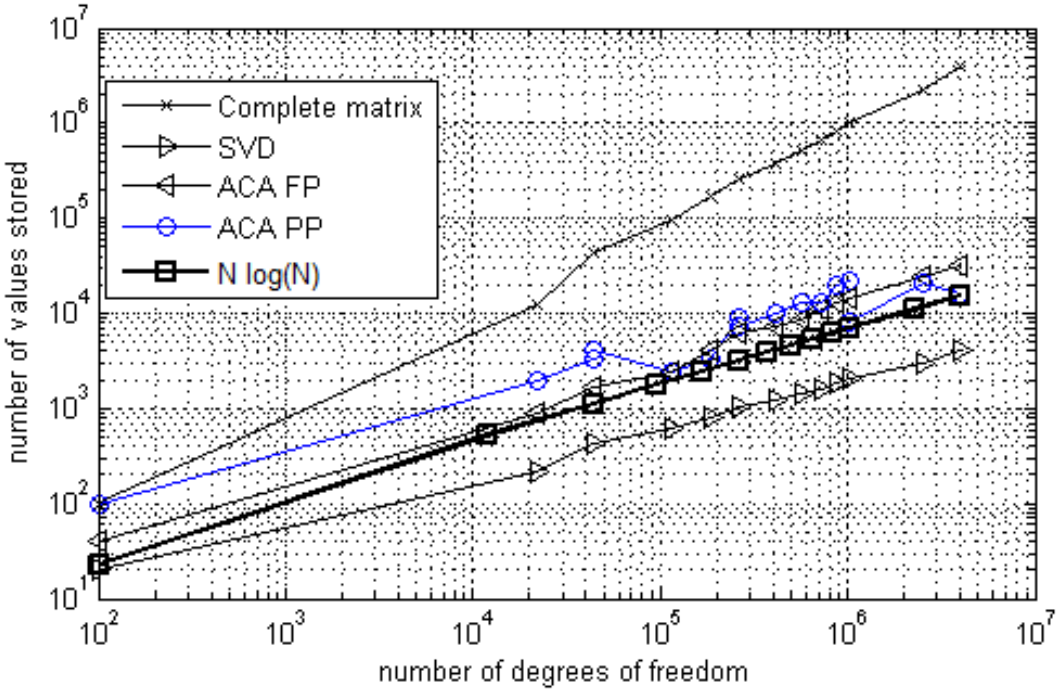

Figure 5.11 – Number of values stored comparison for the computation of the test matrix and its approximations by SVD, ACA fully pivoted and ACA partially pivoted

time shown in Figure 5.10 is decisive to assess the efficiency of this approximation. At the same time reducing memory needed to describe the problem and fast computation of this matrix.

This result was consistent for different values of wavenumber k, which yield different behaviours for the matrix. The behavior of this matrix for different values of $k = 1, 8, 16, 24$ is shown in Figure 5.12.

5.4 Approximation by Lagrange polynomials with Chebyshev nodes

Another way to approximate the terms of the matrix when the block is admissible is to use a polynomial interpolation. There are different choices for how to obtain a separable expression adequate to this approximation. The author chose a polynomial interpolation, in particular Lagrange polynomials with Chebyshev nodes (NEUMANN, 2017). A separable expression in $X \times Y$ is any function $\aleph(x, y)$ which can be written in the form

$$\aleph^{(r)}(x, y) = \sum_{\nu=1}^{r} \phi_{\nu}^{(r)}(x)\psi_{\nu}^{(r)}(y)$$

for

$$x \ \in X, \ y \in Y, \tag{5.15}$$

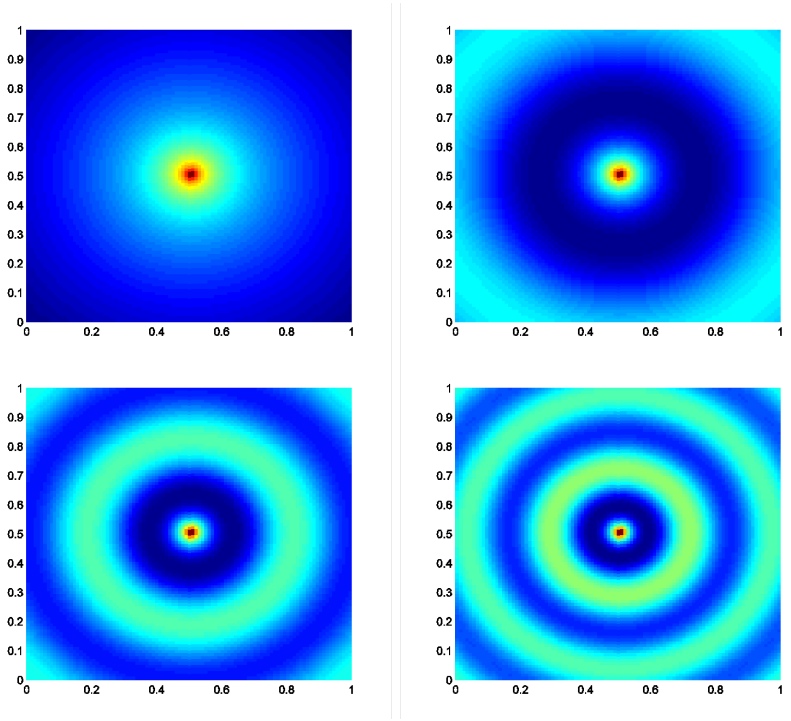

Figure 5.12 – Matrix behavior for wavenumbers $k = 1, 8, 16, 24$.

where $\phi_\nu^{(r)}$ and $\psi_\nu^{(r)}$ are arbitrary functions dependent only on X and Y, respectively. The rank r is called the separation rank of $\aleph(x, y)$ and also the number of terms in the approximation.

An approximation by Lagrange polynomials is obtained by using Chebyshev nodes distributed in the domain of the function and applying the Lagrange polynomial to these nodes (NEUMANN, 2017). The r Chebyshev nodes for a domain bounded by (a, b) are given by:

$$y_k := \frac{1}{2}\left[(a+b) + (b-a)\cos\left(\frac{(2k-1)\pi}{2r}\right)\right],$$

for

$$1 \leq k \leq r. \tag{5.16}$$

The Lagrange polynomial is defined as

$$L_k(y) := \prod_{j=1, j\neq k}^{r} \frac{y - y_j}{y_k - y_j}$$

for

$$1 \leq k \leq r, \tag{5.17}$$

and, therefore, an interpolating function for the separable expression $\aleph$ is

$$\aleph(x, y) \approx \sum_{k=1}^{r} \aleph(x, y_k) \cdot L_k(y). \tag{5.18}$$

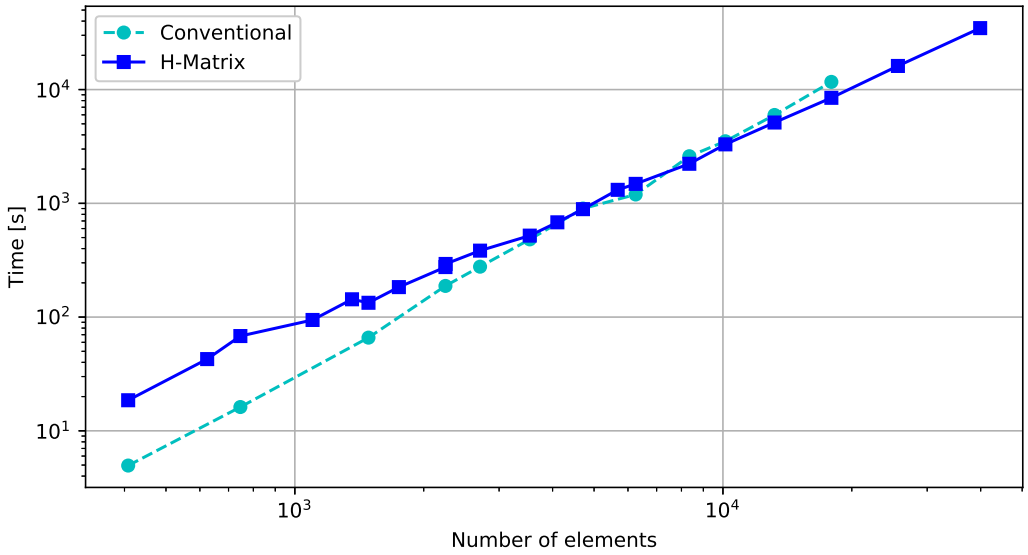

Figure 5.13 – Processing time to build and solve the BEM model using $\mathscr{H}$-Matrices with polynomial interpolation

This new implementation reduced the processing time as well as memory allocation for models with more than $\approx$ 5000 degrees of freedom, as can be seen in Figure 5.13. The different slopes on the processing time curves indicates that the implementation is suited to solve large scale models.

At first try, SharedArrays were used to build the influence matrices using the conventional BEM (NICOLAIDIS, 2018), but this approach was unable to represent the Arrays used in the $\mathscr{H}$-Matrices. The Threads primitive was more adequate to simultaneously build these matrices. As each block of the influence matrix is independent of any other, Multithreading parallelization was applied to the matrix building process. This means that each block of the matrix, wether it is admissible for approximation or not, will be evaluated on a different thread. The parallelization procedure for Multithreading is straightforward in Julia, as most low-level memory management is dealt with by the language, and the user only has to call high-level functions [1]. The parallel implementation was from 2 to 1.2 times faster, depending on the number of degrees of freedom.

[1] Parallel Computing - The Julia Language - https://docs.julialang.org/en/v1/manual/parallel-computing/

6 NUMERICAL IMPLEMENTATION

Software development went through a big rupture in the 1990's when the internet made worldwide collaboration possible by people who shared nothing but their interest in certain software. Software development methodologies and practices changed accordingly, going from a waterfall model in which the steps of development are compartimentized and each team communicates to higher up administrators who kept a strict long term schedule to more decentralized, short-termed, self-organizing collaboration which came to be known as agile development. The difference between those methodologies and its effects on academic software development will be discussed in this chapter.

6.1 Software development

By the beggining of the XVIII century, Charles Babbage designed the Analytical Machine. This was a general computer which could be programmed, and the first program written and published for it was done by Ada Lovelace. This mechanical machine worked with leverages and gears, and thus would be programmed by physically moving the parts of the system. Early eletronic computers worked similarly, by reppluging the computers processor and memory to perform computations. By the 1950's, most programming was performed using machine language, which was processed directly by the processor. A program called Assembler was created to facilitate writing machine language code, which assigned each operation to a string readable by humans. High marine officer Grace Hopper pioneered the compiler paradigm by writing a program that converted a few text strings of computing logic into many spans of machine language code. In 1957, Backus, together with a small team from IBM, created Fortran while working for IBM, the world's first succesful high-level compiled programming language. Now, software written for a computer platform could be run on other platforms, as long as a Fortran compiler for it existed. Until late 1960's most computers were still operated by a human which fed each program at a time, in a process known as batch processing. Operating systems with time-sharing were programmed in Assembly language up until 1970 when Ken Thompson and Dennis Ritchie rewrote their Unix system in a new programming language called C. This meant that Unix could be compiled and run on other computer's architectures than the PDP-11

it was developed on.

Up until 1975, most computers were big and very expensive machines, which meant that only a few computers existed and they were created by a very large team with a strict chain of command. software development followed what is now referred to as waterfall development. This meant that the software's features and scope was determined by a team and each step of the development follows from design, programming and deployment from a top-down hierarchy. This strategy is still widely used, specially in the same enviroments that produce a strict chain of command and hierarchy: government and militar entities. Even though most of the cost of the development is spent by a single and centralized entity, most of the processes of actually developing the software is done by smaller groups which are kept in schedule by their peers and higher ups, by producing reports, documentation and applications (BROOKS, 1995). A general case would be a small team of administrators, each of which leads a group responsible for different aspects of software delopment, for example design, programming and testing teams.

Academic software has been developed much like BASIC has been developed in Darthmouth by John Kemeny and his collaborators: other faculty and undergraduate and graduate students. Some of them went on to work with computing and computer engineering, some didn't and most contributions were made within the period from a few months to a few years. Academic software is often published and its authors are given credit to their implementations, but little to no fee is required to obtain access to it, most are published on papers or textbooks. This style of development is also characterized for being led by a motivated individual or academic group for years on end. This individual or group resorts to working long term to develop with shorter term collaborators. Advisers often propose composing themes for their students to design more powerful designs. This style is different from most waterfall style methodologies, but still shares some similarities. The most blatant difference is that academic software is often free, while non-academic waterfall developed software is often proprietary. MIT's X windowing system is a good example of this, providing its software at a permissive free software license (known as MIT license), which is used extensively by operating systems, most of them proprietary. Some academic software is licensed under proprietary licenses but most are not published at all.

The waterfall style of software development is also used in very big commercial companies such as Boeing, Renault and other organizations which have strict ties with very big companies, as a mediator of the state funding such as the energy sector. Software developed this way is usually very well documented, application specific, and usually monolithic, or lacking compactness and orthogonality (RAYMOND, 2003). This style of development assumes that each part of the team has a limited access and scope of the software, hardware and application, and software teams also had a limiting scope to study, modify and share copies of the programs.

For example, a ballistic program developed for the army could only be developed by

a high rank officer, a team of programmers would be hired to work with a team of hardware technicians for a specific machine to be used, a design team of ballistic specialists, for example. Each team could have an administrator and smaller teams as necessary, kept on a tight schedule to their specific project, in this case a ballistic program. If the schedule is not met, as in the case that the ballistic program producing erroneous values, the administrator or superviser communicates this to its superior and the information is communicated until a decision about the project is taken by a smaller upper rank team.

Access to the software in this style of development is only permitted inside this teams during development and once the software was deployed, the same team could maintain it or not, but the software is owned by the entity.

6.2 Free software development

In 1985, the concept of free software was developed by Dr Richard Stallman at the MIT as he sets forth the GNU project and the Free Software Foundation [1]. Free software means software that respects users' freedom and community and roughly means that users have the freedom to run, copy, distribute, study, change and improve the software. This definition holds for the four essential freedoms, namely[2]:

- The freedom to run the program as you wish, for any purpose (freedom 0).
- The freedom to study how the program works, and change it so it does your computing as you wish (freedom 1). Access to the source code is a precondition for this.
- The freedom to redistribute copies so you can help others (freedom 2).
- The freedom to distribute copies of your modified versions to others (freedom 3). By doing this you can give the whole community a chance to benefit from your changes. Access to the source code is a precondition for this.

This is guaranteed by using a free software license, which is a legal document that specifies the terms and conditions the author imposes on the use and reproduction of the software. Most software are licensed with a proprietary license, which limits or even prohibits the access to source code, install and modification. There are many free software licenses, some more permissive than others, but the main license and the one that brought about the vast landscape of free software today is the GNU GPL (FREE SOFTWARE FOUNDATION, 2007).

[1] The GNU Manifesto - GNU Project - Free Software Foundation

[2] What is free software? - GNU Project - Free Software Foundation - https://www.gnu.org/philosophy/free-sw.en.html

With the advent of cheap personal computers connected to the internet in the beggining of the 1990's propiciated new development styles, the most succesful one being the distributed collaborative development of Linux. This program was developed in the internet, with communication using e-mail and with no hierarchy or strict chain of command. Linux is a free software, as it is licensed using the GNU GPL, which makes sure that anyone who uses the software can study, modify and distribute it. Because anyone can study and modify it, there were many modifications done to Linux and sent back to its author. This contributions improved the software in ways that traditional development styles couldn't achieve. This difference in development style and its impacts on software is described in the paper by Eric Raymond (RAYMOND, 1999), and this new techniques are adopted by software companies to develop more robust, well maintened and free software. The new style of software development is named Open Source, and it's made possible by the existence of free software (mostly using the GPL license) and the internet.

One of the new development styles is described as agile software development. This style is characterized by using its values to make decisions, as described in the agile manifesto:

> We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:
>
> - Individuals and interactions over processes and tools;
> - Working software over comprehensive documentation;
> - Customer collaboration over contract negotiation;
> - Responding to change over following a plan.
>
> That is, while there is value in the items on the right, we value the items on the left more.

The use of the agile development style results in small self-organizing groups, which collaborate frequently to solve problems and deliver working software. Two common well documented agile development methodologies are Kanban and Scrum. Both of them focus on smaller development cycles than the waterfall style with teams responsible for different parts of the project communicating frequently and design, programming and deployment being done as a iterative process. This can be achieved by prototyping and using proofs of concept to deliver working software early and constantly, delivering new features as requested by the end-users and not by following a strict long term project schedule.

6.3 UnBEM and existing BEM implementations

There are many BEM implementations on existence today, some are free software maintained by research departments, such as BEMpp (2015), and Deal.ii (2019). Some are

unmaintained research code often published on books and papers, one of which is the well designed program by Dominguez (1993) and the very succesful program from Liu (2009) or the program from Kirkup (2007). Some are proprietary such as Beasy [3]. The University of Brasília (UnB) has been involved in developing a BEM implementation for potential problems, isotropic and anisotropic elasticity problems and acoustics problems for more than 20 years and in this mean time produced a great deal of software for specific problems, specially in Fortran, Octave/MATLAB, Python [4], and Julia [5]. Software tends deteriorate when not used and many programs are in risk of being made obsolete for not being used, maintained and/or well-documented. To help resolve the short-comings of developing software in an academic environment, where many collaborators are undergraduate and graduated students who do not get involved in developing or maintaining software further than the research period, the use of software development methodologies are used in this work.

The use of version-control using Git[6] is also designed so that new developments are easily integrated into existing software. The collaborators for the project can be identified, new features can be added and removed more easily. Online repositories are used to decentralize the distribution and a free software license is used.

MATLAB[7] is a high level dynamically typed interpreted language developed in 1984 to supply an easy and efficient way to scientists and engineers to implement programs to solve numerical problems more easily using the LAPACK library, at the cost of speed [8]. The use of this language in research and development on universities is very high, partially because of the ease of prototyping and its excelent plotting library. It's well suited to a variety of numerical methods, but as the matrices grow, the speed of computation drops because of the nature of the language that is interpreted, not compiled.

Many Fortran programs for the BEM existed prior to the development of this thesis, and some of them had been translated to MATLAB to facilitate the development for newcomers to numerical methods and programming.

GNU Octave[9] is another high level programming language which resembles MATLAB. Octave is part of the GNU project and it's free software and each MATLAB program written in this work is tested in both languages to assure compatibility. This was done to provide a free option for those who may not possess a proprietary license of MATLAB.

Initially, this work intended to produce an Octave/MATLAB implementation only, but as the programming evolved to a more complex form, the limitations of this language

[3] BEASY - Engineering Software and Services - https://www.beasy.com
[4] BEM course in Python - https://github.com/eder-albuquerque/bem-course
[5] BEM_base - https://github.com/alvarocafe/BB
[6] Git Documentation - https://git-scm.com/
[7] MATLAB from MathWorks - https://mathworks.com
[8] LAPACK in MATLAB - MATLAB & Simulink - https://www.mathworks.com/help/matlab/math/lapack-in-matlab.html
[9] GNU Octave - https://www.gnu.org/software/octave/

started to show. MATLAB is a interpreted language and is not suited to large-scale problems. Even though, some scripts were produced and are presented below.

Julia [10] is a fairly new programming language, developed from 2012 onwards. It's a high level, high performance language and its main characteristic is that even simply written programs by scientists and engineers run as fast as dedicated compiled programs from low level programming languages such as C or FORTRAN.

6.3.1 BEM_base

The BEM implementation shown in this work has been carried out as BEM_base since July 2017. The implementation is in Julia and is mainly based on Octave scripts used in the author's master dissertation. Julia is a dynamically type high level programming language that takes advantage of LLVM (Low level virtual machine) to create just-in-time compilation (JIT) machine independent binaries [11]. This allows it to be fast while maintaining the generality atributed to high level languages. Julia is very user friendly and has a growing community and contributors, as most of Julia is written in Julia itself. Julia is developed and maintained by MIT and uses the MIT license.

Up to 2020, BEM_base is built out of 3 BEM programs linked together into useful modules. One of the goals of BEM_base's design was ease of extension, so that the implementation could soon reach and further the state-of-the-art of BEM implementations. Git was used as a source control application so that new features could be added easily by new contributors and online repositories could be easily built and manage. Adding source control also helps to create more stable programs and facilitates maintenance, specially for projects with hundreds of contributors. The inspiration for the design was the open source finite element library Deal.ii (ARNDT et al., 2019).

Most three-dimensional geometrical models created for this work are created using FreeCAD[12] and Gmsh[13] or just the latter. Gmsh is very powerful for creating geometrically simple models and meshing, but it lacks many of the CAD capabilities on FreeCAD. More complex models are then first drawn in FreeCAD and later meshed in Gmsh and most simple geometries are drawn directly in Gmsh.

6.3.2 Program organization and information flow

The program's source code is stored in the folder "src". There, each folder will contain the scripts for a different BEM implementation, which will constitute a module. Julia modules are usefull as specific functions are kept separate. All of the modules are

[10] Julia Programming Language - https://julialang.org
[11] The LLVM Compiler Infrastructure Project - https://llvm.org/
[12] FreeCAD: Your own parametric 3D modeler - https://www.freecadweb.org/
[13] A three-dimensional finite element mesh generator with built-in pre- and post-processing facilities - https://gmsh.info/

loaded by calling the script "BEM_base.jl". The 6 implementations solves the BIE for the constant 2D linear element, NURBS 2D element and a constant 3D bilinear triangular element. These implementations are all accelerated by the hierarchical methods described in chapter 5.

Inside folder "src", each implementation is on a separate subfolder, with names according to the implementation as explained below:

- waveconst2D - constant 2D linear element for Helmholtz problems
- waveconst3D - constant 3D bilinear triangular element for Helmholtz problems
- wavenurbs2D - NURBS element for Helmholtz problems

All of the implementations use the hierarchical matrices with both ACA+ and approximation by Lagrange polynomials and Chebyshev nodes.

Inside the "notebooks" folder, the specific implementations for different problems are presented in the form of Jupyter notebooks. These are interactive web-based Julia code and plots and LaTeXequations. The contents of each notebook is given in Appendix B.

The folder "tests" contains files for testing the Helmholtz and Laplace equations. These should be useful when modifying the program, as the tests are based on known analytical solutions given below.

The information flow used in this work is based on similar implementations in BEMpp and deal.II, consists in generating the CAD model in FreeCAD or Blender and then generate a mesh using Gmsh. The mesh may define Physical Surfaces which contains the boundary informations. This mesh ('.msh') file is then imported into our solver, the boundary conditions are set and the problem is solved. The results are saved into the mesh file so graphical analysis is possible both in Gmsh and ParaView [14]. This workflow was created using exclusively free software to be as transparent and repeatable as possible (BUZOGANY, 2017).

6.4 Classical acoustics problems

In this section, classical acoustics problems will be solved using the BEM_base and compared to experimental, numerical or analytical solutions. These problems will act as both validation tests of the implemented algorithm and as proofs of concept before solving more complex problems.

[14] ParaView - paraview.org

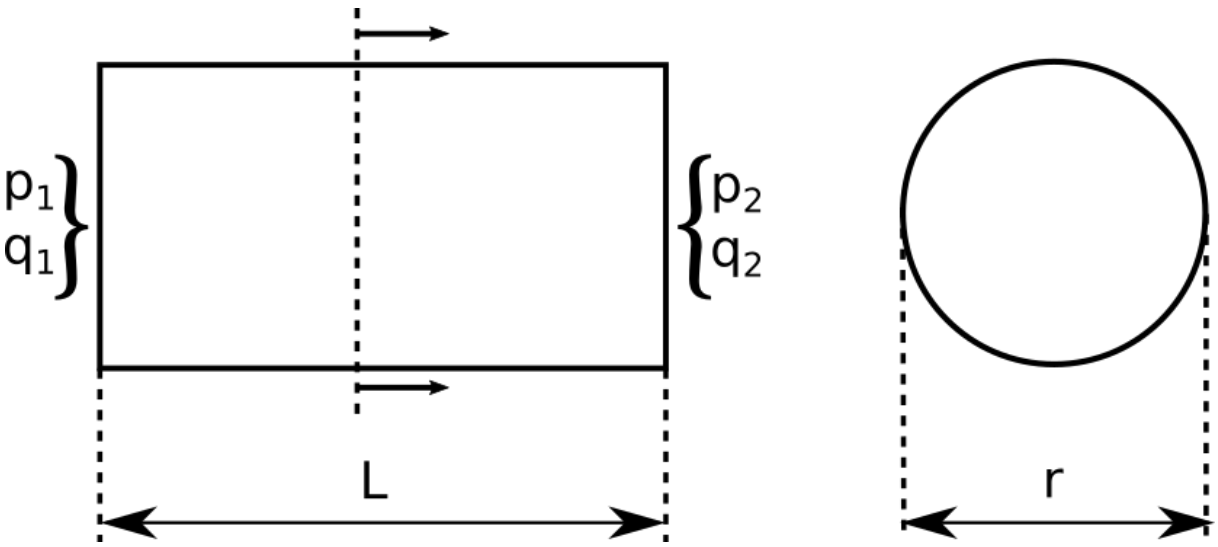

Figure 6.1 – Geometry and boundary conditions for acoustic ducts.

6.4.1 Ducts

Consider the geometry of a long duct, the diameter is considerably smaller than the length. In this case, the axial acoustic modes will have a much lower frequency than the radial modes. In this frequency range, the acoustic field will be approximately one-dimensional, and analytical solutions are easily obtained. A duct's acoustic behavior is defined by the boundary conditions on its endings. For this reason, the ducts presented here are separated by that. The solutions for this cavities can be found by applying a transfer matrix for the acoustic pressure and flux (GIBERT, 1988). A diagram showing the acoustic pressure p and flux q for points 1 and 2 which corresponds to the two endings of the duct. This duct is assumed to be cylindrical and its inner radius is r. The length of the duct is L.

Closed-closed duct

In Figure 6.1, this case corresponds to setting both values of q to zero, i.e., $q_1, q_2 = 0$. The solution of the transfer matrix system shows that the resonant wavenumbers are:

$$k_n = \frac{n\pi}{L}, n = 0, 1, 2, ... \tag{6.1}$$

where $n = 0, 1, 2, ...$ are all positive integers. Each n corresponds to a different resonant mode for the duct.

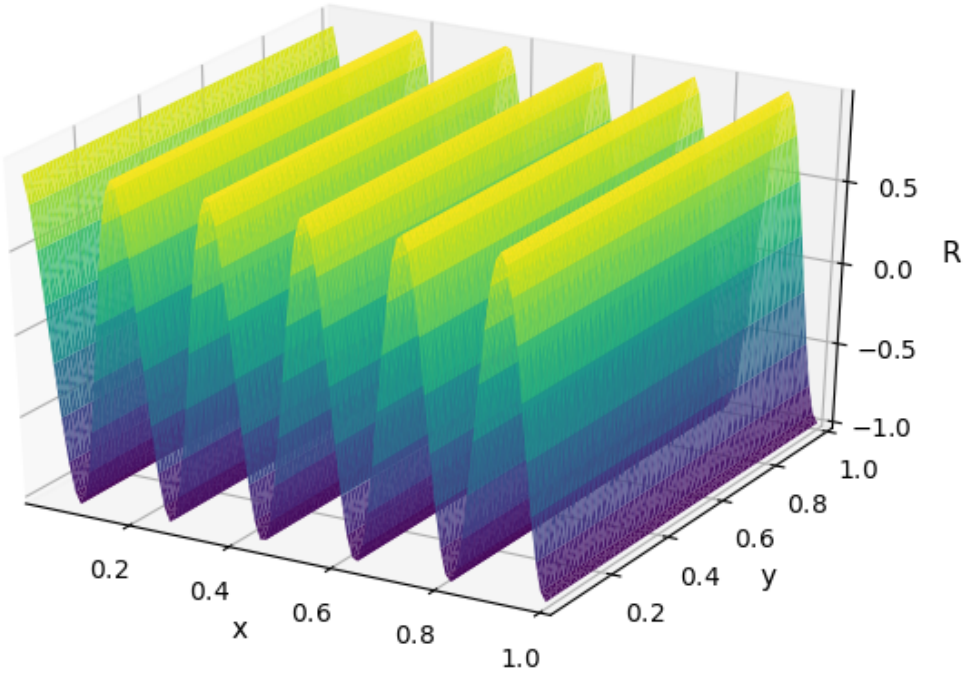

Figure 6.2 – Results for the closed-closed duct problem.

Open-closed duct

This case corresponds to setting the value p_1 and q_2 to zero. The solution of the transfer matrix system shows that the resonant wavenumbers are:

$$k_n = \frac{(2n-1)\pi}{2L}, n = 1, 2, ... \tag{6.2}$$

Each n corresponds to a different resonant mode for the duct.

Open-open duct

To obtain this configuration, set both values of p to zero, i.e., $p_1, p_2 = 0$. The solution of the transfer matrix system shows that the resonant wavenumbers are:

$$k_n = \frac{n\pi}{L}, n = 0, 1, 2, ... \tag{6.3}$$

This is, of course, the same frequencies for the closed-closed cavity.

6.4.2 Vibrating cylinder

This example is an infinite cylinder bidimensional model. This case was chosen because an analytical solution is readily available. The walls of the cylinder are subjected

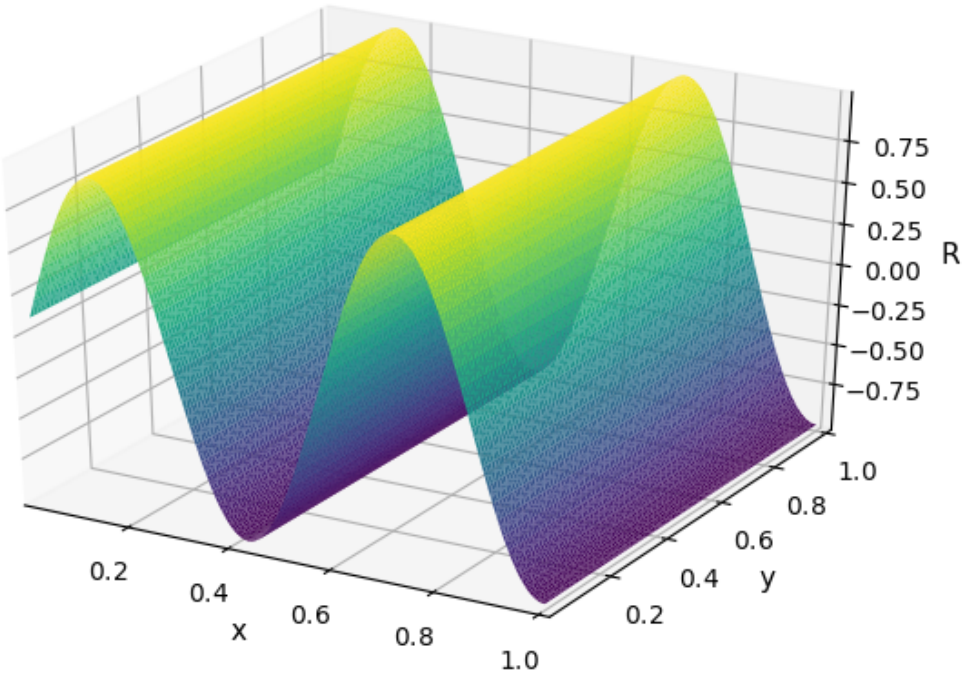

Figure 6.3 – Results for the open-closed duct problem.

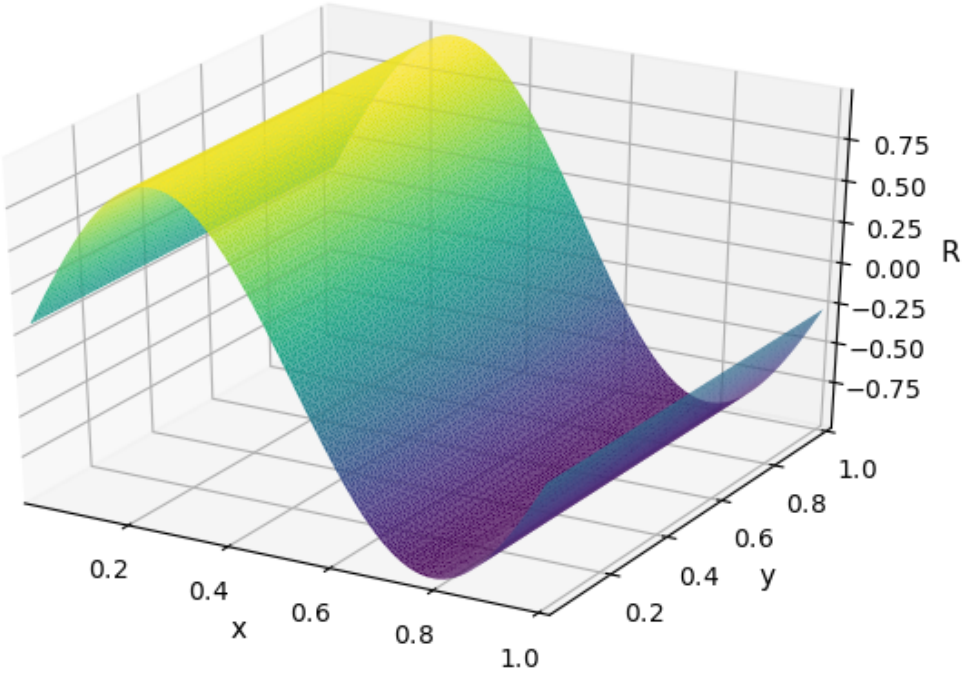

Figure 6.4 – Results for the open-open duct problem.

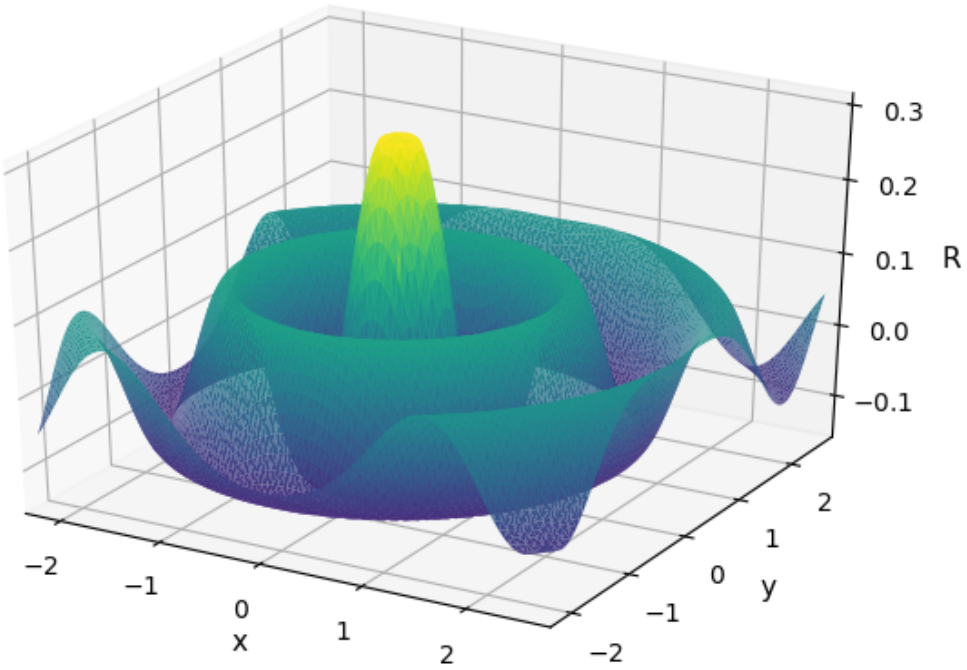

Figure 6.5 – Results from the exterior infinite cylinder problem.

to Dirichlet boundary conditions ($\phi = 0$). The analytical solution given by Costa, Borges e Afonso (2008) is given below

$$\phi_{cyl}(r, k, a) = -\frac{V}{k}\frac{H_0^{(2)}(kr)}{H_1^{(2)}(ka)} \tag{6.4}$$

where V is the amplitude of the cylinder vibration (set to unity), k is the wave number, a is the radius of the cylinder, r is the distance from the cylinder, $H_0^{(2)}$ and $H_1^{(2)}$ are the Hankel functions of second species and order 0 and 1, respectively.

6.4.3 Vibrating sphere

The vibrating sphere is also a classic problem, together with the vibrating cylinder. The solution for the vibrating sphere is the same of the cylinder, let an unit sphere be submitted to a constant acoustic flux of unity. The surrounding acoustic field will be described by the following velocity potential:

$$\phi_{sph}(r, k, a) = \frac{Va^2}{1 - ika}\frac{e^{ik(r-a)}}{r} \tag{6.5}$$

where V is the amplitude of the sphere vibration (set to unity), k is the wave number, a is the radius of the sphere, r is the distance from the sphere. The solution for the vibrating sphere is considerably simpler than the cylinder because of the symmetries allied to the use of a spherical coordinate system.

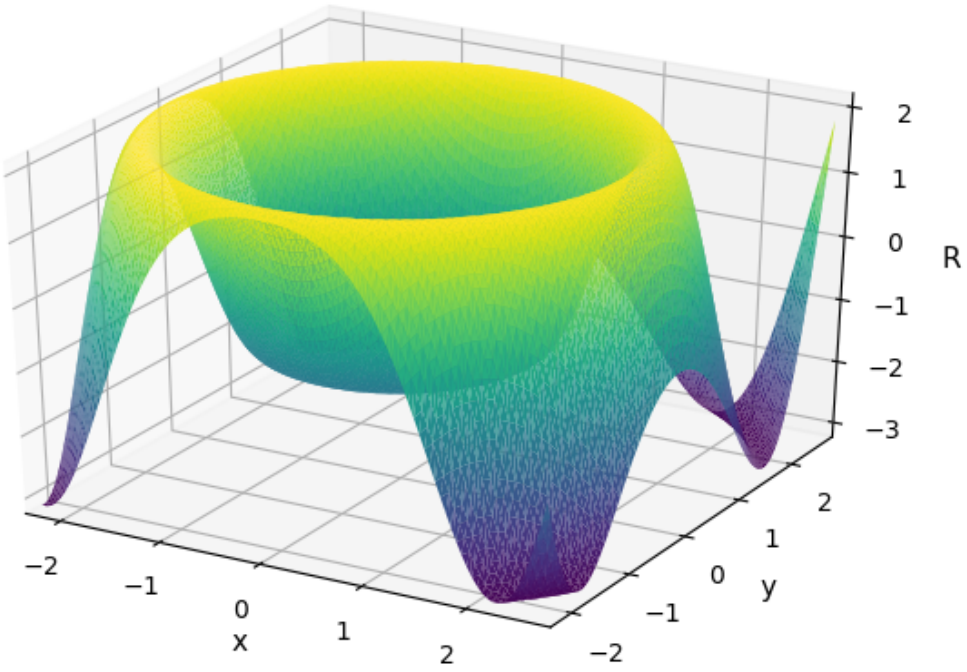

Figure 6.6 – Results for the vibrating sphere problem.

7 RESULTS

In this chapter, numerical results for internal and external problems using the BEM are given. The results are compared to traditional BEM models.

7.1 Vocal tract cavity

The study of the acoustics of the voice is very important to many areas, including phonoaudiology, music and noise control. Many professionals use the voice as their main media to convey information and ideas, such as teachers, singers, orators and theater professionals. It is desirable to produce a resonant voice to reduce the strain and fatigue while speaking or singing. This resonant quality is already well known by bel-canto singers and theater professionals, but its production is still based in body sensations of the speaker (TITZE, 2001).

Ekholm, Papagiannis e Chagnon (1998) tries to fill the gap in the communication between voice pedagogues and voice scientists. Subjective ratings were related to objective measurements taken from acoustic analysis of the voice signal. Some acoustic phenomena correlations to critical perceptual parameters were identified, helping to bridge the terminology gap between vocal artists and scientists.

The study of the acoustic properties of the resonant voice and Vocal Tract (VT) during phonation are an essential step for the advance of vocal technology and vocal coaching for professionals which use spoken speech. A schematic model of the speech system is shown in Figure 7.1. The VT is the region contained between the glotis and mouth.

The acoustic resonance of the VT defines the quality of the vowel produced. Different VT configurations produce different modes and resonance frequencies, perceived by listeners as different vowels. A study of the resonance strategies of 22 singers shows that trained singers may use the resonance of the vocal tract to obtain certain desirable characteristics in the voice (HENRICH; SMITH; WOLFE, 2011). The two first resonance frequencies of the VT were obtained experimentally. These results indicate that singers can repeatedly tune their resonances with precision, and that there can be considerable differences in the resonance strategies used by individual singers, particularly for voices in the lower ranges.

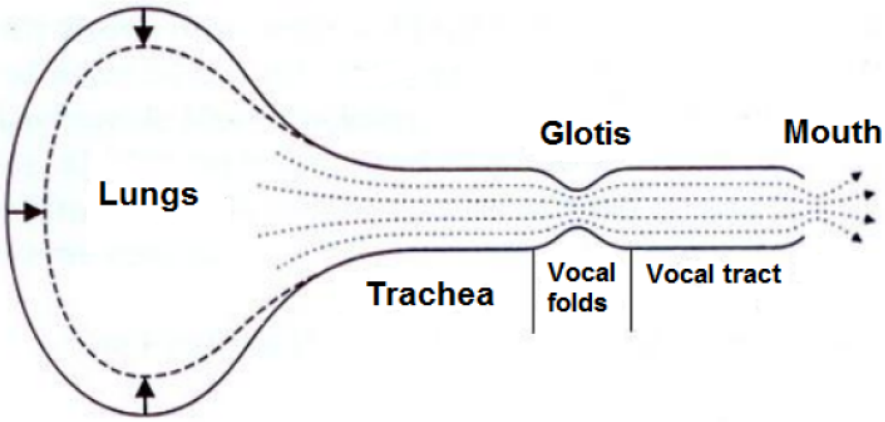

Figure 7.1 – Schematic model of the speech system (CATALDO; SAMPAIO; NICOLATO, 2004)

7.1.1 Case study description

Using Magnetic Resonance Imaging (MRI) visual procedures (CLéMENT et al., 2007) obtained geometric informations of the VT. Table 7.1 shows the cross-section areas of the vowel \a\.

Table 7.1 – Area A_i and standard deviation σ_i of the cross-section areas of the VT for the vowel \a\ in square centimeters (CLéMENT et al., 2007, modified)

Cross-section area															
i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
A_i	1,8	1,8	2,8	1,5	0,8	1,3	1,5	1,7	2,8	4,5	7,1	9,3	13,5	15,5	7,8
σ_i	0,1	0,1	0,3	0,2	0,1	0,1	0,1	0,2	0,2	0,3	0,2	0,4	0,3	0,4	0,2

The cross-sectional areas are essential dimensions to identify the acoustic behavior of the VT. These parameters will be investigated using a GA optimization. Ferreira (2014) created meshes of the VT and applied them into BEM, FEM and TM models. The FEM models were created using ANSYS. These models are compared to the GA optimization in the results section.

7.1.2 Vocal tract model

The VT configuration is a description of the geometry during phonation, so there are many biological restrictions to it. Zhou et al. (2013) divides the VT configuration into 6 different groups and 8-12 articulatory parameters, which are used to obtain the configuration from speech signals using an annealing algorithm. The VT model used in this work is composed of 15-17 equally spaced circular areas from the glotis to the mouth. The VT is divided into 6 groups and the cross-section area gradient is restricted so that there are no discrete jumps which are not organic. The VT is modelled using Gmsh by

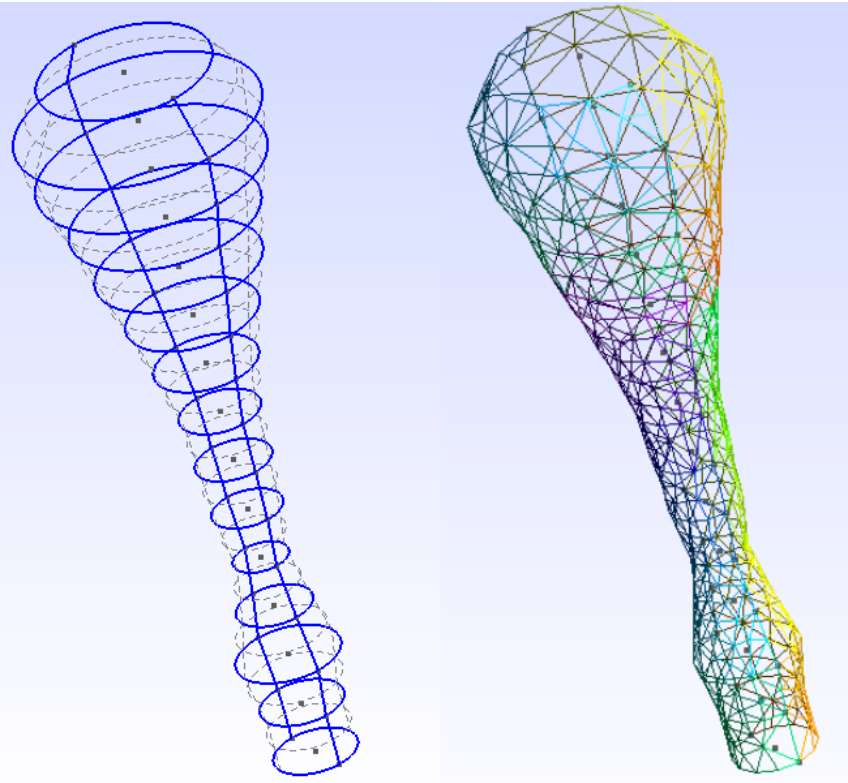

Figure 7.2 – Vowel /A/ geometry and mesh

manipulating its native geometrical file format (.geo) directly to generate a mesh with triangular elements.

The optimization is performed on the VT configuration described by Clément et al. (2007) for the vowel /a/, showed on Figure 7.2.

7.1.3 Genetic Algorithm (GA) optimization

A Genetic Algorithm (GA) is a stochastic algorithm that mimics natural phenomena as operators. GAs are search techniques based on the processes of natural selection for survival through population genetics (HOLLAND, 1975). For the GAs to start evolving, we can use the steps selection, recombination (crossover), mutation, and replacement, where the survival-of-the-fittest mechanism can be applied to the candidate solutions (HAUPT; HAUPT, 1998).

In the initialization, the algorithm replies evolutionary genetics and generates a random population by uniform distribution. The chromosomes represent the elements in evolutionary algorithm space, where their features, called genes (quantified by values called alleles), are the problem inputs to be rated by the fitness function.

The selection allocates more copies of those solutions with higher fitness values and the roulette-wheel select is the procedure chosen to accomplish this idea. After that, recombination combines parts of two parental solutions to create new, possibly better, solutions. The mutation, on the other hand, randomly modifies the solutions' allele. At

the end, a percentage of the best elements is simply copied to the next generation using an elitism probability. This is a mechanism to guarantee the best solutions are transmitted to the current generation (GOLDBERG, 1991).

The algorithm feeds the system back and evaluates each element from the population. Using the aforementioned evolutionary strategies, the fittest elements will have more probability to pass their features to the next generations. They are depicted by real base, where each gene matches to the hereditary characteristics to be combined and evaluated.

The real base is useful when the parameters to be optimized are continuous variables (RAHMAT-SAMII; MICHIELSSEN, 1999). Working with any decimal digits of precision the computer uses floating point numbers to represent the chromosome and its size is equal to a vector that represents the problem solution; thereby each gene represents one problem variable (RODRIGUES, 2007).

The aim of this study is to build a methodology to obtain the VT configuration given its acoustic behavior. This will be carried out using a Genetic Algorithm (GA) approach, in which the characteristics of the VT model will be selected based on the acoustic output produced by the model. The GA used is a floating point chromosome, which was used to maximize a fitness function defined by restricted parameters. Physical properties of the system such as resonance mode frequency and mode shapes were used to construct the fitness functions. The flowchart shown in Figure 7.3 outlines the process carried out in this work.

7.1.4 Fitness Function

Three different fitness functions are analyzed to reach the experimental data of Table 7.1.

The first fitness function devised uses the resonance frequency of the first n_{freq} acoustic modes to restrict the cross-section area and select the individuals. The value of the resonance frequency is obtained by selecting the peaks in the FRF of the system. The fitness function created controls the cross-section areas by selecting the individuals with the maximum value of the function showed in equation 7.1. This function is the square root of the inverse of the square difference between the goal value of the resonance frequency ω^i_{goal} and the value obtained for individual r, ω^i_r for each mode i.

$$fit_1(r) = \left[\sum_{i=1}^{N}(\omega^i_{goal} - \omega^i_r)^2\right]^{-\frac{1}{2}} \tag{7.1}$$

where N is the number of physical points being analyzed by the program.

One may use the value of the pressure along the tube to create a fitness function. The same operation of the Equation (7.1) is realized replacing the frequency into acoustic

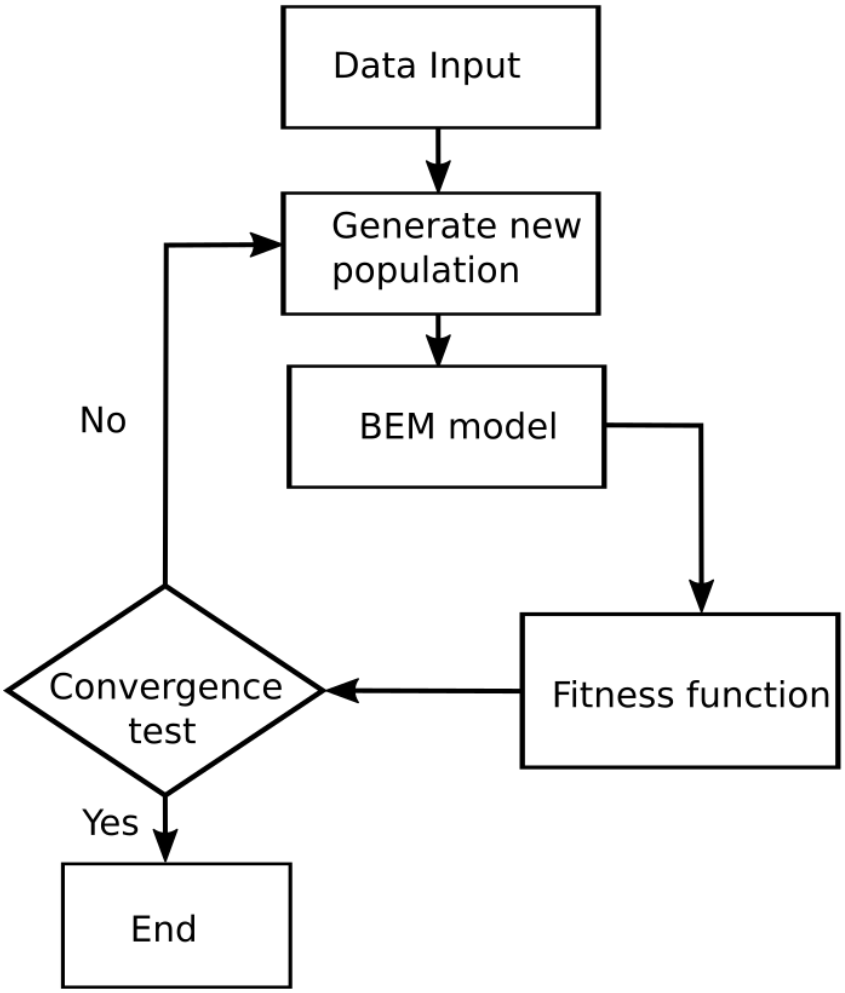

Figure 7.3 – Flowchart for the GA optimization problem

pressure values:

$$fit_2(r) = \left[\sum_{i=1}^{N}(p_{goal}^i - p_r^i)^2\right]^{-\frac{1}{2}} \tag{7.2}$$

Similarly, the value of the pressure variation at determined positions can be used to construct a fitness function defined by Equation (7.3)

$$fit_3(r) = \left[\sum_{i=1}^{N}(q_{goal}^i - q_r^i)^2\right]^{-\frac{1}{2}} \tag{7.3}$$

Preliminary results were conducted to test the convergence of the fitness in function of the cross-sectional areas. Two models with four areas of cross section $[1, 1, 1, 1]$ and $[1, 1, 2, 2]$ were performed. Table 7.2 and 7.3 shows the cross-sectional areas obtained using these three fitness functions.

A summary with convergence (OK) and non-convergence (X) is shown in Table 7.4

Therefore the fitness function 3 is the best choice to be optimized.

Table 7.2 – Comparison of the cross-section areas for different fitness functions. Goal: [1,1,1,1]

	S_1	S_2	S_3	S_4
Goal	1.000	1.000	1.000	1.000
Fitness function 1	2.126	1.849	1.867	2.149
Fitness function 2	2.228	2.228	2.228	2.228
Fitness function 3	0.999	0.998	1.005	0.999

Table 7.3 – Comparison of the cross-section areas for different fitness functions. Goal: [1,1,2,2]

	S_1	S_2	S_3	S_4
Goal	1.000	1.000	2.000	2.000
Fitness function 1	0.802	0.899	1.815	1.619
Fitness function 2	0.938	0.939	1.877	1.878
Fitness function 3	1.001	0.998	2.003	1.991

Table 7.4 – Summary of convergence of fitness functions

	Fitness 1	Fitness 2	Fitness 3
Modal frequency	OK	OK	OK
Pressure modal shape	X	OK	OK
Pressure variation modal shape	X	X	OK

7.1.5 Results

As fitness function fit_3 provides the best results for the desired optimization, this analysis establishes for the genetic toolbox the following parameters:

- $N_{ger} = 50000$, the number of generations;
- $N_{ind} = 150$, the number of individuals in the population;
- $p_c = 60\%$, crossover probability;
- $p_m = 4\%$, mutation probability;
- $p_{elit} = 2\%$, elitism probability;
- $p_{diz} = 20\%$, decimation probability;
- $N_{diz} = 100$, the step of generations for the occurrence of decimation.

The number of modes used in the optimization was $n_{freq} = 10$.

Figure 7.4 shows the graphical development of the best individuals and their means over the generation.

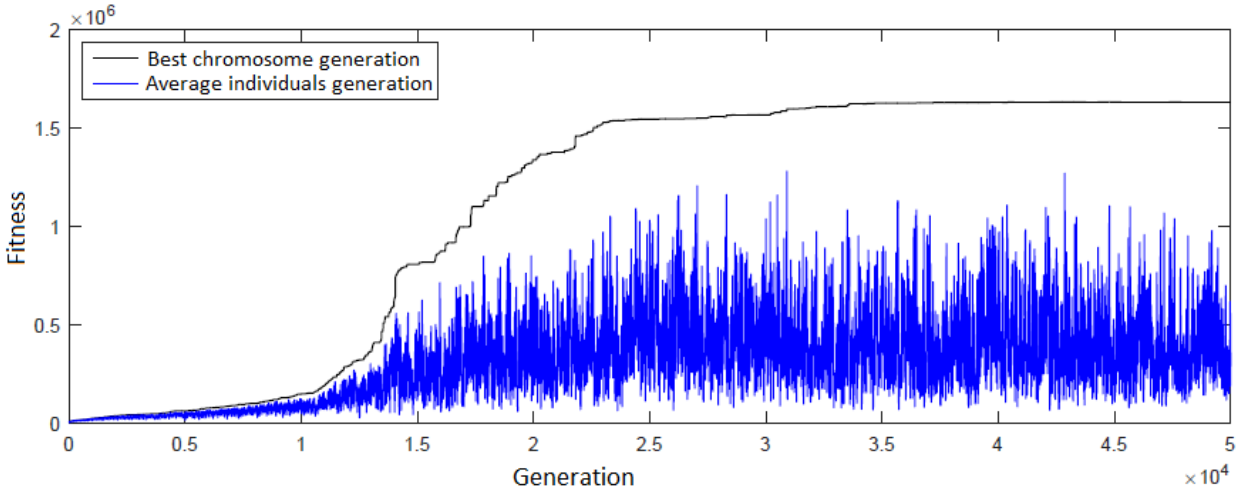

Figure 7.4 – Fitness function of the best chromosomes and their means over the generation

Table 7.5 presents the results obtained by the optimization and compares them to the fitness values, with a relative error. The largest relative error is $3,51\%$.

Table 7.5 – Cross-sectional areas A_i of the VT for the vowel \a\ (CLéMENT et al., 2007) and the best chromosome founded by the GA optimization toolbox

Cross-sectional areas							
Section	1	2	3	4	5	6	7
A_{fit}	1,80	1,80	2,80	1,50	0,80	1,30	1,50
A_{AG}	1,77	1,78	2,79	1,53	0,82	1,33	1,54
error (%)	1,47	0,63	0,06	2,05	2,77	2,21	2,65

8	9	10	11	12	13	14	15
1,70	2,80	4,50	7,10	9,30	13,50	15,50	7,80
1,75	2,89	4,66	7,34	9,59	13,91	16,00	8,07
3,04	3,11	3,51	3,41	3,16	3,02	3,24	3,40

Figure 7.5 shows the FRF result obtained by GA optimization, compared to the goal produced by the TM. The frequency response of the acoustic behavior obtained by the GA is practically the same as the fitness function.

Figure 7.6 shows the comparison between the objectives achieved by the TM and the optimization using pressure mode shape values.

Figure 7.7 shows the corresponding MAC values for these first 10 modal forms. You may notice that there is a strong correlation between the vibration modes of the diagonal elements with MAC values nearly equal to one. Other values outside the main diagonal have practically zero values. This indicates that there is a strong correlation between the pressure mode shapes optimization obtained by the GA and the goal. Between the second and third modes exists a poor orthogonality with MAC values equals to 0.2.

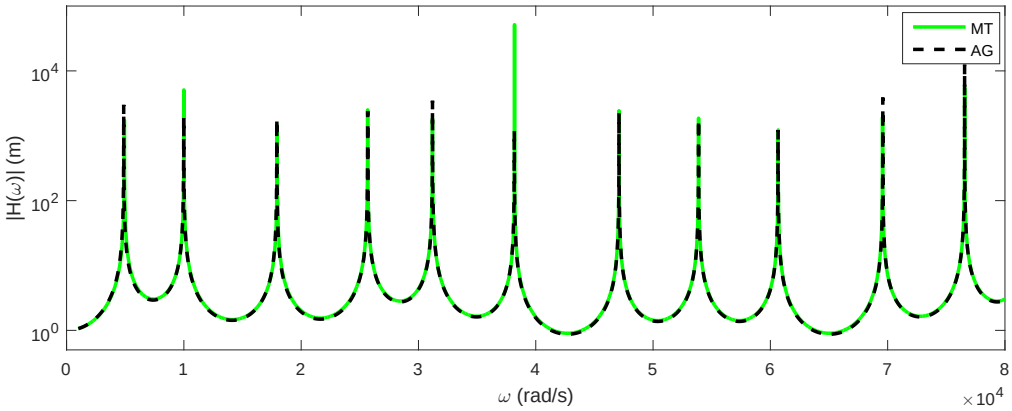

Figure 7.5 – FRF comparison between TM and GA optimization

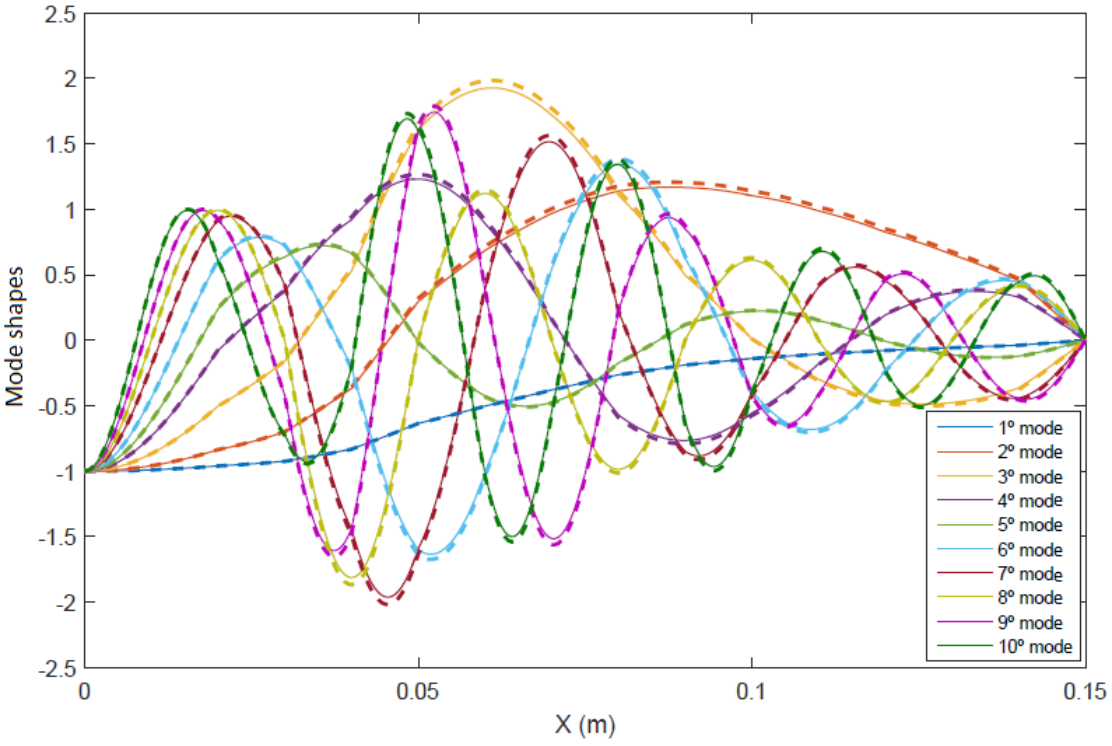

Figure 7.6 – Pressure mode shapes comparison between the TM and the GA optimization (dashed) for the firsts 10 mode shapes

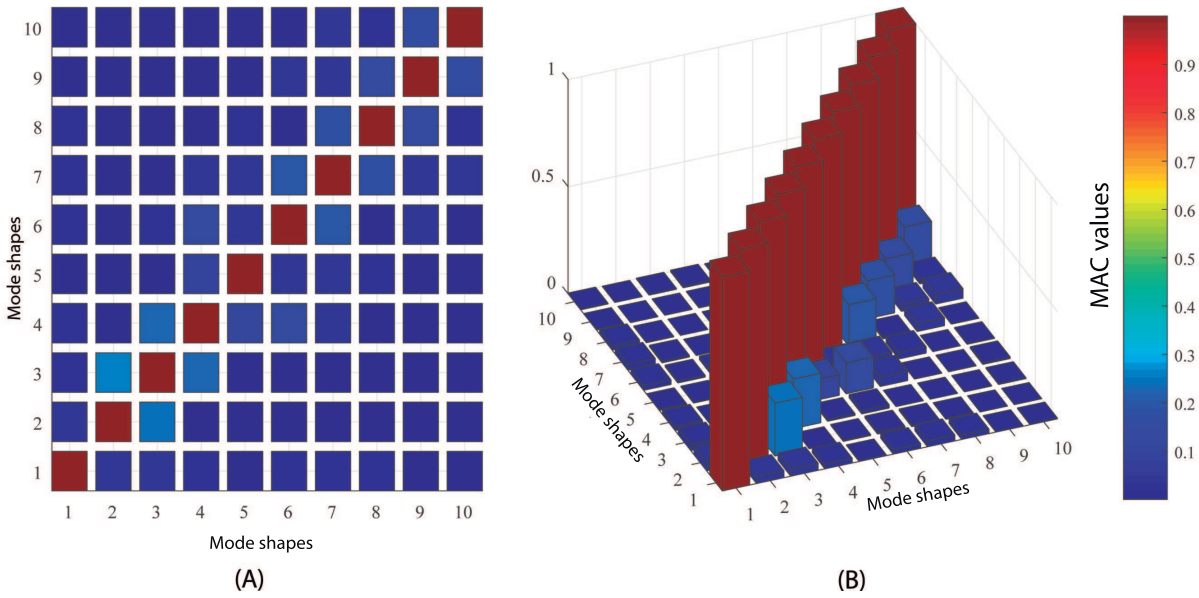

Figure 7.7 – MAC values for the firsts 10 pressure mode shapes

Figure 7.8 shows a comparison between the objectives achieved by the TM and using the GA toolbox for the pressure variation mode shapes.

Figure 7.9 presents the corresponding MAC values for the firsts 10 mode shapes.

There is a strong correlation between the vibration modes of the diagonal elements,

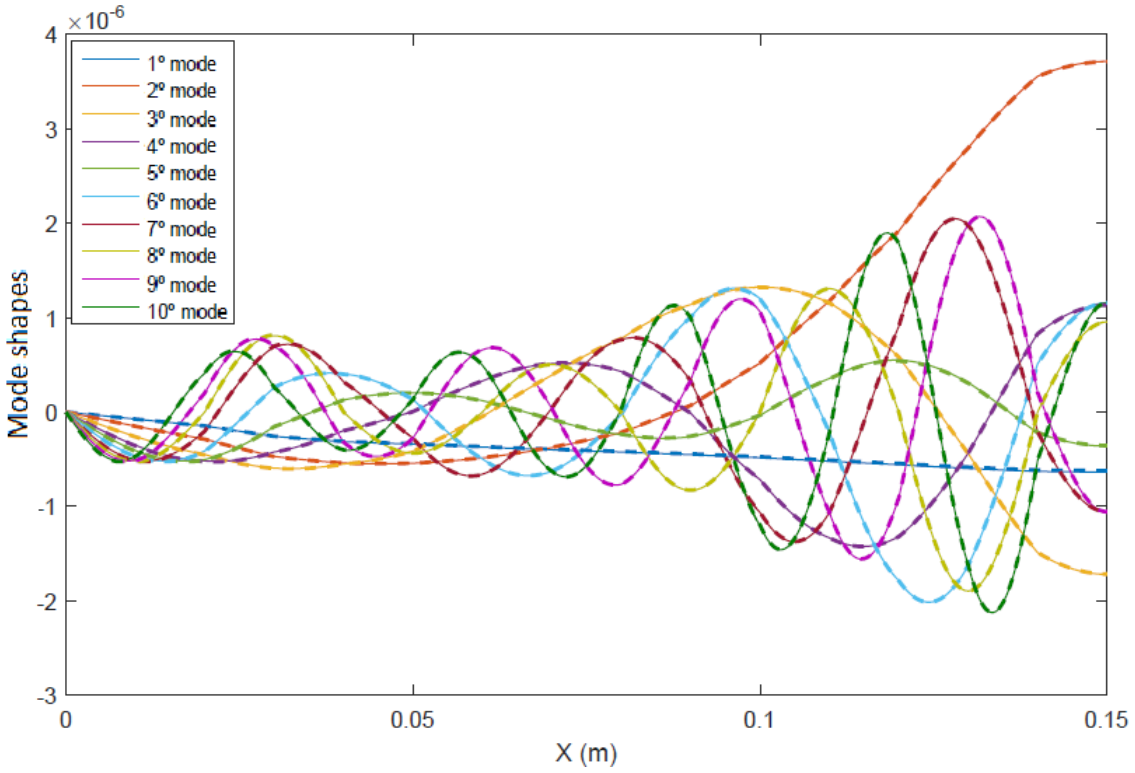

Figure 7.8 – Pressure variation mode shapes comparison between the TM and the GA optimization (dashed) for the firsts 10 mode shapes

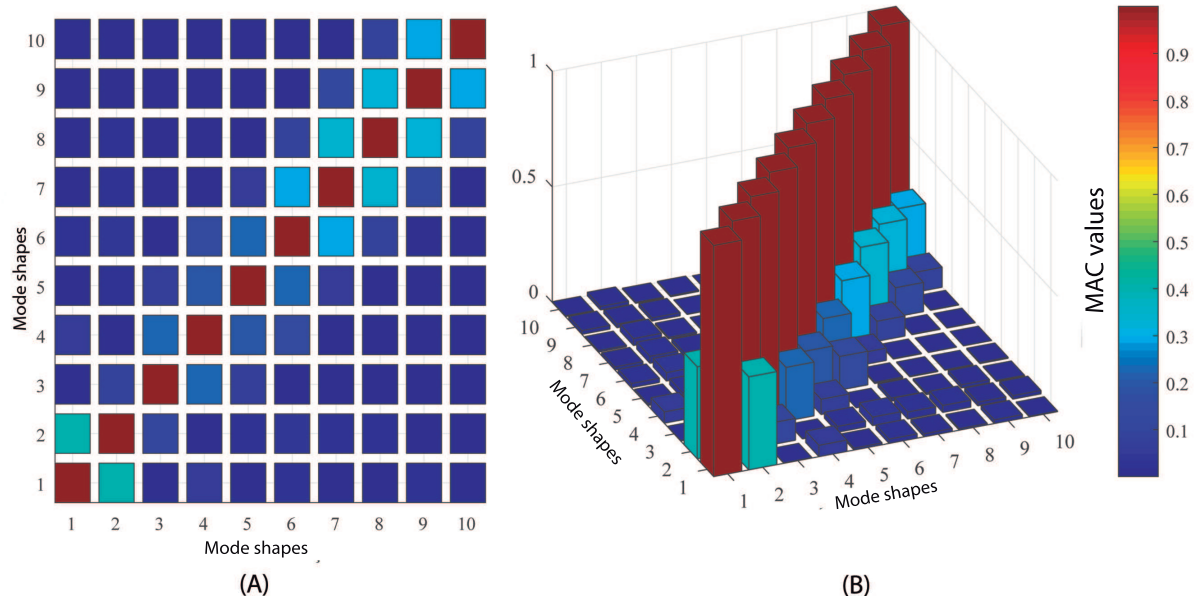

Figure 7.9 – MAC values for the firsts 10 pressure variation mode shapes

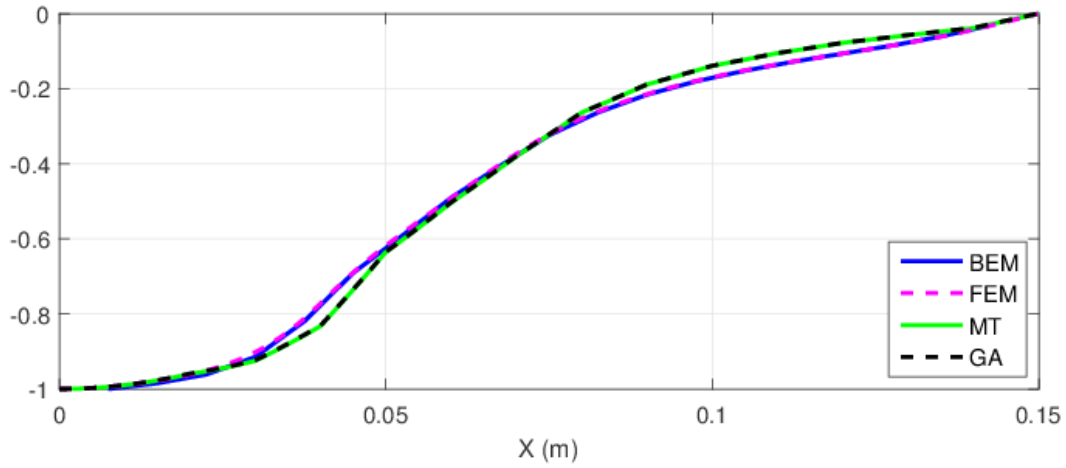

Figure 7.10 – First mode shape: comparison between BEM, FEM, TM and GA

as they have MAC values almost equal to one. Other values have virtually null, except for nearby vibration modes (modes 1 and 2, 2 and 3, ...). There average correlation between the first and second order of about 40%.

In this section we compared the Boundary Element Method (BEM) and Finite Element Method (FEM) with the GA optimization results for the firsts three normalized mode shapes in figures 7.10, 7.11 and 7.12.

The TM results are similar to the BEM and FEM (FERREIRA, 2015) results. The TM analytical formulation is a powerful tool that allows the GA implementation for achieving a specific goal.

The difference between the TM model with respect to the BEM and FEM models can be explained by the absence of three-dimensional effects, not implemented in the

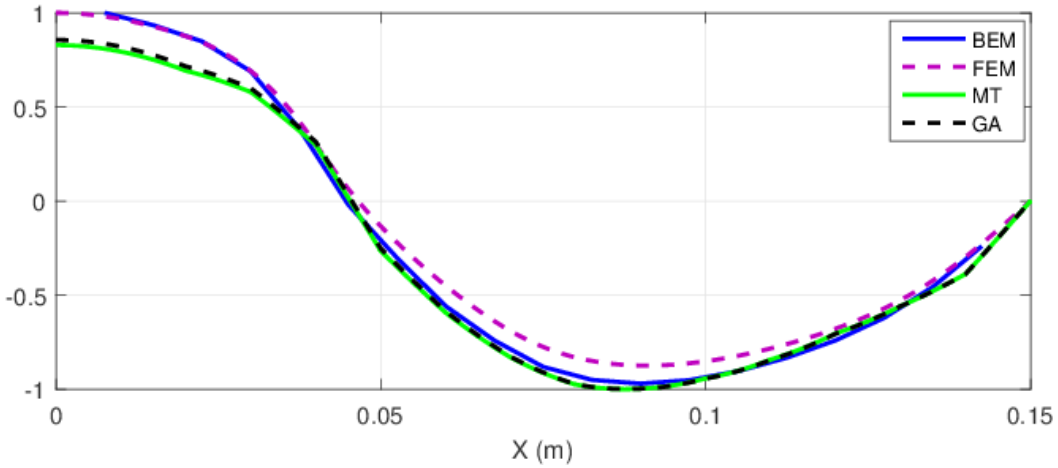

Figure 7.11 – Second mode shape: comparison between BEM, FEM, TM and GA

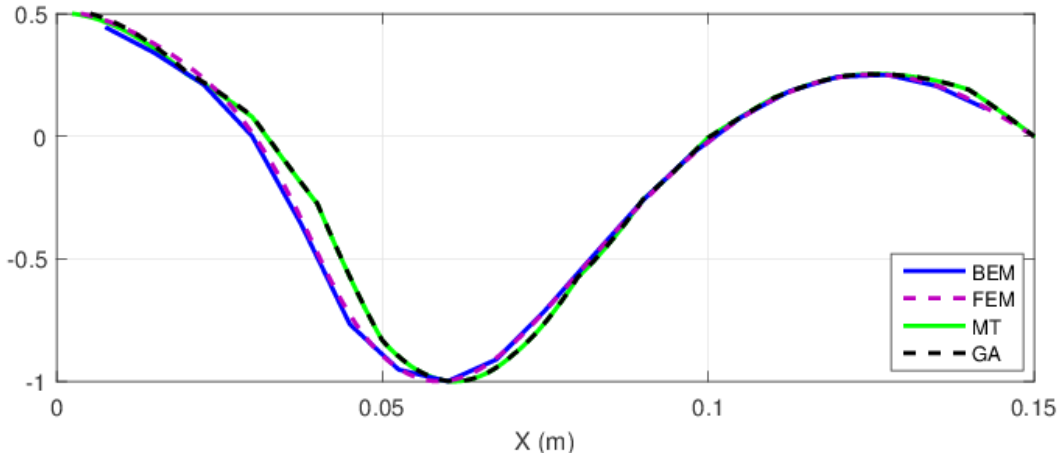

Figure 7.12 – Third mode shape: comparison between BEM, FEM, TM and GA

TM analytical model. There is almost no difference between the GA and TM graphical models, indicating the GA efficiency.

An analysis of three fitness function using different parameters of the model was carried out. A fitness function using the pressure variation along the tube of both the goal and the individual tubes was used in the final genetic algorithm. The algorithm was able to reflect the cross section areas of an arbitrary tube with constant length. The use of pressure variation mode to construct the fitness function is adequate, as the pressure variation is a dual variable and has a direct influence from both the position in length and the cross section area of the tube. The use of the pressure mode to construct the fitness function does not yield the expected values as the pressure is a primal variable and describes a state of the section and not its dynamic. The use of the resonance frequency to construct the fitness function does not guarantees that the solution is unique, as many different acoustic configurations can generate the same FRF and yet show completely different acoustic behavior.

Two validation cases were carried out using three different fitness functions. The

fitness function which obtained the best results used the pressure variation difference between the goal case and the individual. For these cases, only four variables were controlled by the genetic algorithm and the results were consistent. Afterwards, a vocal tract model based on anatomical data was created and the genetic algorithm was used to attempt to reproduce the cross section area of the vocal tract. This simulation used 15 variables of cross section area and a constant length. The cross section areas converged to the goal case with 3.4% relative error. MAC values guarantees the orthogonality between the optimization and TM analytical formulation for the pressure and pressure variation mode shapes. The difference between the TM model in relation of the BEM and FEM models can be explained by the absence of three-dimensional effects, not implemented in the TM analytical model. There is almost no difference between the GA and TM graphical models, indicating the GA efficiency.

7.2 Acoustic Levitator - TinyLev

Acoustic levitation is the phenomenon of trapping particles in mid-air using sound waves. This is possible because standing waves contains nodes which act like tweezers, trapping the particle between high acoustic pressure regions. In August 2017, the mechanical engineering department of the University of Bristol launched a new project called TinyLev with the goal of popularizing the study of acoustic levitation worldwide. The project consisted of the design, building and evaluation of a novel single-axis levitator based on multiple low-voltage ultrasonic transducer. The levitator was named TinyLev for its compact size and operates at 40 kHz in air and can trap objects above 2.2 [g/cm^3] density and 4 mm in diameter. The levitator can be built using off the shelf components, so its easy to reproduce anywhere in the world. Figure 7.13 shows the final configuration of the levitator while trapping a small styrofoam ball.

7.2.1 Analytical and experimental modelling of the TinyLev

A circular piston oscilating harmonically at a single frequency produces the a complex acoustic pressure p at a point located at point $\mathbf{r}$:

$$p(\mathbf{r}) = p_0 V \frac{D_f(\theta)}{d} e^{i(\phi + kd)} \tag{7.4}$$

were $k = \omega/c$ is the wavenumber, c is the wave propagation speed, p_0 is the amplitude constant that defines the piston's output power, V is the oscillation peak-to-peak amplitude, θ is the angle between the piston normal and $\mathbf{r}$, ϕ is the emitting phase of the source, d is the propagation distance in free space. The function $D_f = 2J_1(ka\sin(\theta))/ka\sin(\theta)$ is the directivity function of a piston source, J_1 is a first order Bessel function os the first kind and a is the radius of the piston.

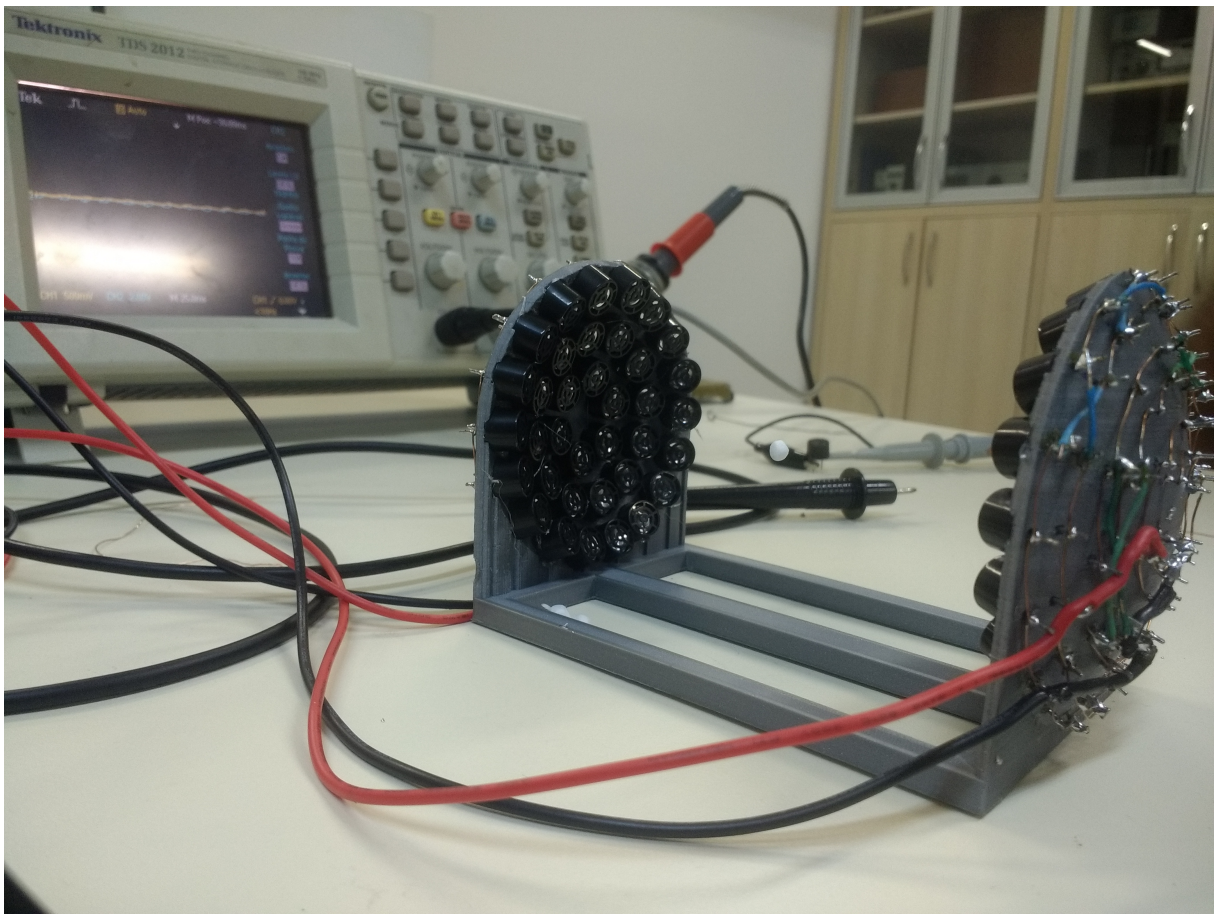

Figure 7.13 – Acoustic levitator TinyLev developed by researchers from the university of Bristol. There's a small styrofoam ball levitating between the transducers.

On the original paper by Marzo, Barnes e Drinkwater, the total acoustic field generated by N transducers is the addition of the acoustic fields created by each of them. The force exerted on a sphere due to this complex pressure field is calculated using the Gor'kov potential $\mathbf{F} = -\nabla \mathbf{U}$ (1962):

$$U = 2K_1(|p|^2) - 2K_2(|p_x|^2 + |p_y|^2 + |p_z|^2), \tag{7.5}$$

where

$$K_1 = \frac{1}{4}V\left(\frac{1}{c_0^2\rho_0} - \frac{1}{c_p^2\rho_p}\right), \tag{7.6}$$

and

$$K_2 = \frac{3}{4}V_p\left(\frac{\rho_0 - \rho_p}{\omega^2\rho_0(\rho_0 + 2\rho_p)}\right). \tag{7.7}$$

where V_p is the volume of the spherical partical, c_p and ρ_p are the speed of propagation and specific mass of the particle, p is the complex pressure obtained for each transducer, and p_x is the derivative of p with respect to x.

With this analytical model, the position, direction and phase of the acoustic signal was determined to produce standing waves which produces acoustic levitation. Once this configuration was first developed, other configurations were obtained by altering the original TinyLev. In this work, the position of the two sides of the levitator were approximated

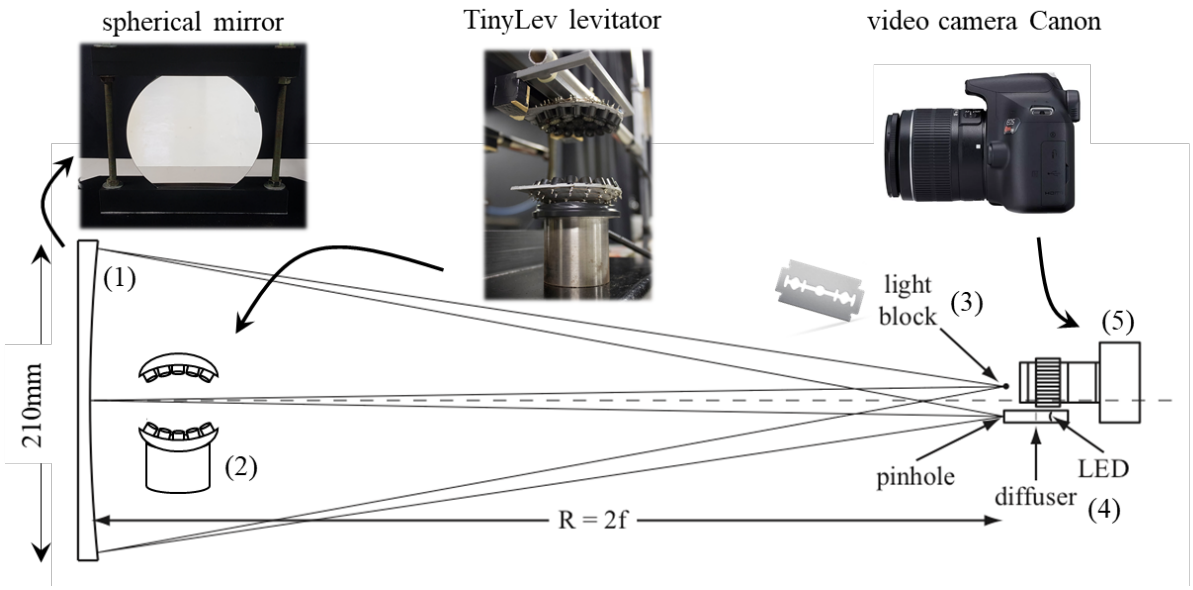

Figure 7.14 – Z-Schlieren experimental arrangement. (adapted from (SCHLIEREN..., 2020)). (1) spherical mirror; (2) TinyLev acoustic levitator (3) Light block (razor blade); (4) Point light source; (5) video camera.

so that the standing waves could be detected in the experimentally obtained images from a Schlieren apparatus.

7.2.2 Schlieren apparatus

As a dynamic and straightforward visualization tool, schlieren systems are primarily applied to conduct qualitative visual studies. Schlieren optics provide an informative, non-intrusive method for studying transparent and optical media (GWYER, 1934). The setup including the spherical mirror, light source, light block, and video camera requires careful alignment in a space that is at least 4 meters long and 2 meters wide. The visualization system was composed of a spherical mirror with a diameter of 21 cm and 139 cm focal length, protected aluminum mirror. A CANON camera model EOS Rebel T6 was used for image capture, adjusted under a shutter speed of 1/4000 s; the aperture of f5.3 and ISO 800 (SAMSUDIN et al., 2015). The point light source is an 8800 W battery-powered white LED flashlight coupled to a closed chamber with a single hole (approximately 400 micrometers). A standard razor blade mounted on an $x - y$ adjustable optics post and positioned next to the light source was used as the light block. The acoustic levitator understudy is positioned vertically on a horizontal mount with a 60 mm distance between acoustic transducers. The stream-wise direction of the fluid flow is the y-direction, and the knife-edge is positioned parallel to this direction. Figure 7.14 shows a schematic representation of the experimental setup.

The TinyLev was modelled in 2 and 3 dimensions. The physical characteristics for both problems were the same as described for the analytical model, section 7.2.1. Briefly, room temperature $T \approx 300$ K, atmospheric pressure $p = 0.87$ atm, based on the altitude

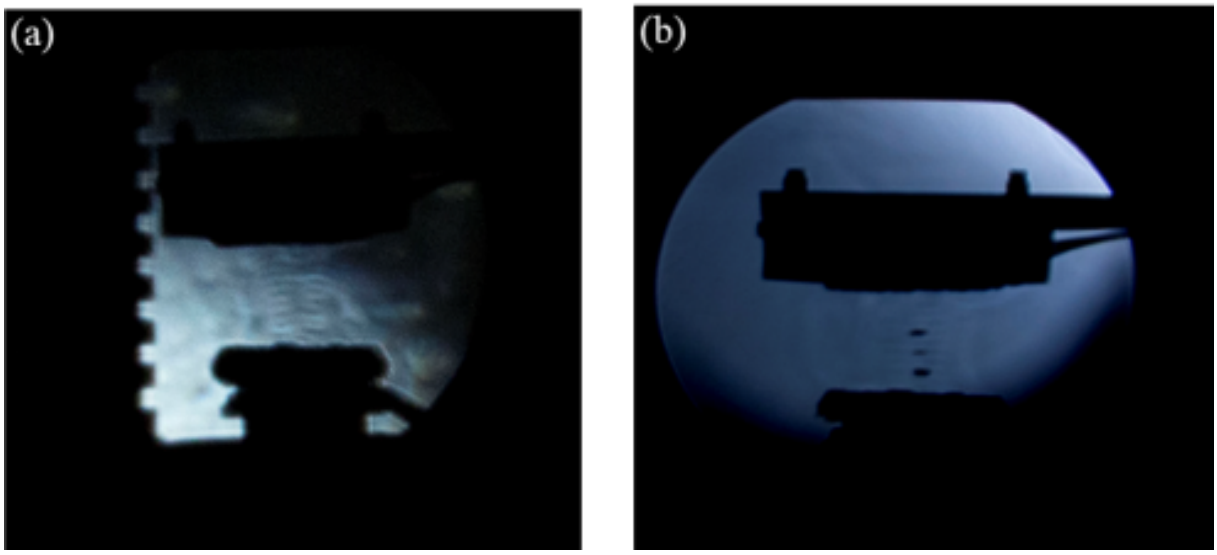

Figure 7.15 – An instantaneous schlieren image for 40kHz. (a) standing pressure waves image. (b) small objects levitation image.

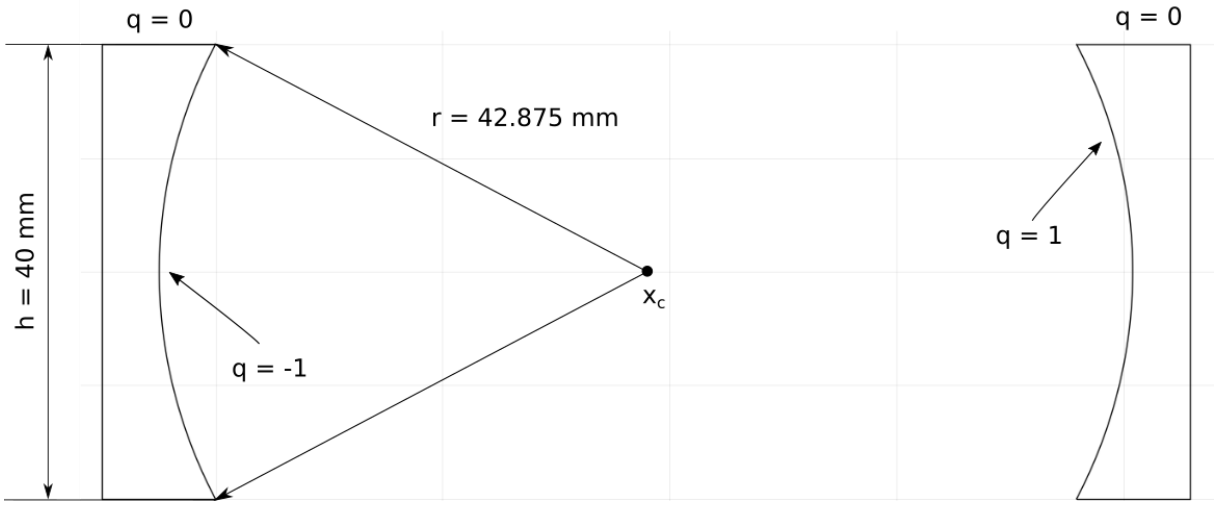

Figure 7.16 – Geometrical information for 2D TinyLev model

from sea level and temperature. The speed of sound in this conditions is calculated as $c \approx 344$ m/s.

The 2D model was built using 8 NURBS curves, 4 for each side of the levitator. The model was based on ANDRADE et al. (2014,2010) and consists of two scattering surfaces shaped like circle arcs. These arcs belong to a circle with a radius which is a multiple of the wavelength of the excitation. The distance between the two sides of the levitator is determined by this radius and the height of the levitator. The radius of the curves is set to five times the wavelength of the acoustic excitation, e.g., $r = 42.875$ mm.

The resulting acoustic response for such configuration is shown in Figure 7.17. It's possible to count 16 fringes of 8.575 mm each between the two sides of the levitator. The configuration obtained is consistent with previous work in 2D acoustic levitation modelling (ANDRADE et al., 2014).

In the 3D model, the BEM with constant triangular elements accelerated by hi-

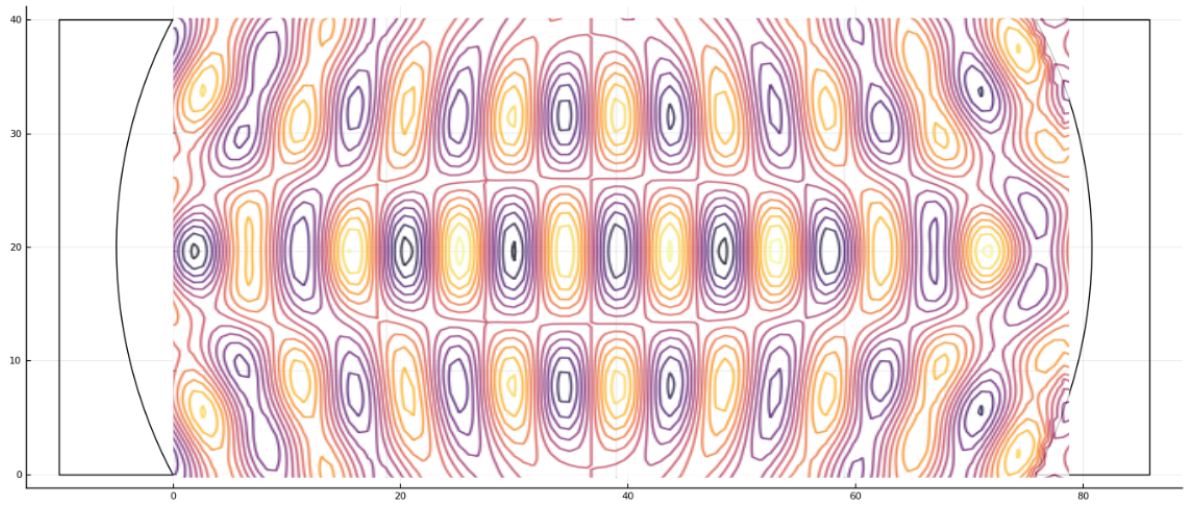

Figure 7.17 – Acoustic response for 2D TinyLev model

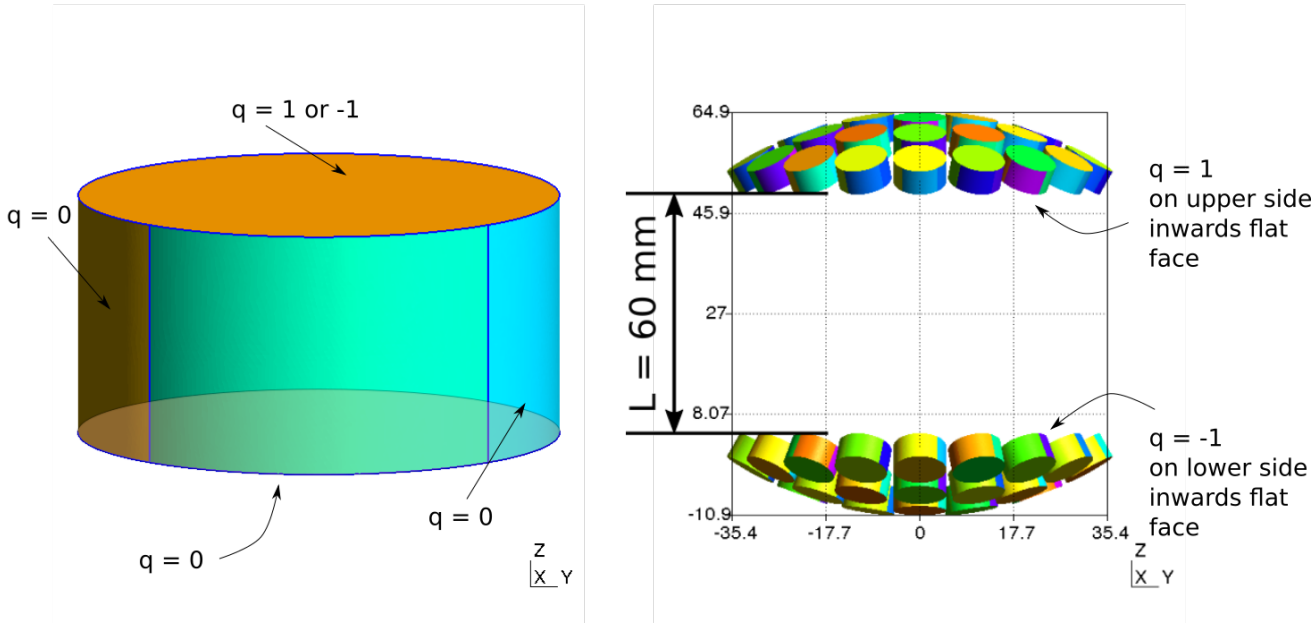

Figure 7.18 – Boudnary conditions for each cylinder

erarchical matrices was used. The maximum number of elements for each cylinder was 64 and the total number of elements was 4438, for the 72 cylinders. The mesh was generated by Gmsh from reading a .geo file created using a Julia script and the geometric information. The boundary conditions for each cylinder is of rigid wall, except for the plane surface pointed towards the center of the levitator. The boundary conditions for these surfaces depends on which side of the levitator the cylinder is located. In one side, the boundary condition is $q = 1$ and, on the other, $q = -1$. The left side of figure 7.18 shows one cylinder with the boundary conditions applied to it.

Due to experimental constraints, the two sides of the TinyLev apparatus were approximated to produce an adequate Schrilien imaging. The result was an approximation

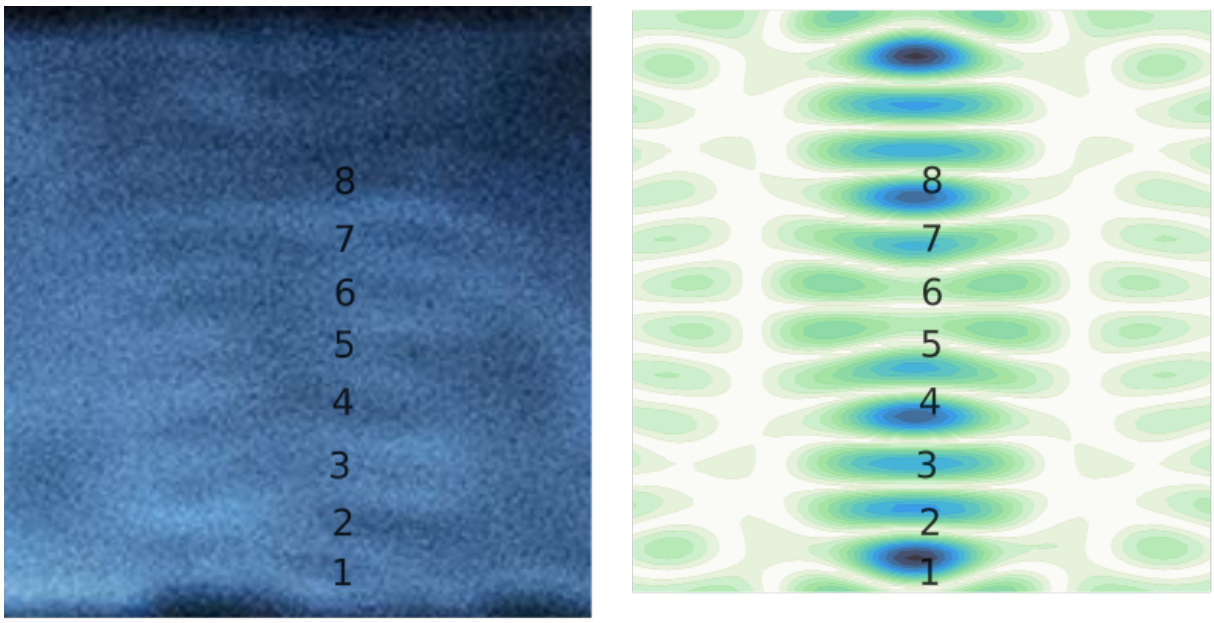

Figure 7.19 – 3D TinyLev acoustic response and experimental results

of the two sides until the distance between them was $L = 60$ mm, as can be seen on the right side of Figure 7.18.

Figure 7.19 shows the acoustic pressure obtained by the numerical model and the visualization produced by the Schrilien apparatus. It also numbers the fringes on both results and the correlation between them numbered from 1 to 8. It's possible to notice a good correlation between the results, for the given region between the two sides of the levitator. The position where levitation is observed corresponds to the fringes observed in the Schrilien imaging.

A three-dimensional $\mathscr{H}$-Matrices BEM implementation was built and used to model the novel non-reverbetant acoustic levitator TinyLev. Up to this point, the reverberant properties of the TinyLev were known to be very small compared to the trapping force, which was observed in the simulations carried out using our algorithm. The implementation used NURBS curves to model the geometry in both 2D and 3D, and the effects of the scattering of waves by the 3D printed base and the transducers were taken into account.

The $\mathscr{H}$-Matrices approximation by polynomial interpolation was an important addition to be able to carry out the three-dimensional simulations as many curves were used to create the model. Even though the speed up may seem small, the methodology allowed us to run ever bigger models, as memory requirements for the $\mathscr{H}$-Matrices scales up in a logarithmic trend, as opposed to the exponential encountered in the traditional implementation.

8 CONCLUSIONS

This work presents a formulation for a hierarchical matrices accelerated BEM for the Helmholtz equation. An implementation of this formulation is presented and results were obtained for internal and external acoustics. The software developed is available at github.com/alvarocafe/BEM_base and is licensed under the GPL V.3.0. The methodology for further the development and maintenance of the software is presented, as well as tests which corresponds to analytical cases the software is able to reproduce. To take full advantage of modern hardware, parallelization was applied whenever possible, from the matrix building to solving the linear system. The two-dimensional isogeometric boundary accelerated by hierarchical matrices is the single most important contribution of this work, but solutions using constant elements are also provided in both two and three dimensions. The implementation is then used in two models: a vocal tract and an acoustic levitator.

Hierarchical matrices are used to divide the domain of the problem into clusters labeled by a binary tree. These clusters are then tested using an admission criteria which decides if the interaction between those cluster are adequate for low rank approximation. Two methodologies to approximate the influence matrices were implemented.

An optimization using genetic algorithms is performed to obtain the geometric configuration of a vocal tract from acoustic information only. This inverse problem is solved by defining a fitness function which minimizes the difference between the acoustic flux in a finite number of points located inside the vocal tract. This optimization proved to be able to infer the three-dimensional axissymetric geometry of the vocal tract from the acoustic flux measured in 20 points, arranged in a straight line.

The TinyLev acoustic levitator is a novel experimental apparatus which is intended to be low-cost and readily available to therefore democratize acoustic levitation throughout the world. The design is open source and all the parts needed can be purchased in electronics stores. The levitator operates by producing ultrasound from 72 transducers arranged in a specific geometric configuration. The transducers are divided equally into two groups, each positioned opposite to each other. The transducers from each group will generate an ultrasound signal of 40 kHz in phase, but the transducers of the first group are in phase opposition with the second group. Each group is displayed in three concentric circles on an hexagonal pattern. The circles themselves are positioned on the surface of a sphere which radius consists of a multiple of the wavenumber of the sig-

nals produced by the transducers. This configuration allied with the opposition of phase from the two sides of the levitator produces a stationary wave. In the peaks and valleys this three-dimensional wave pattern, non-resonant acoustic levitation is obtained. A novel two-dimensional model for this type of non-resonant levitator was developed in this work, based on similar solutions from the bibliography. The TinyLev levitator was manufactured by the author and a Schlirien apparatus was set up to observe this stationary wave. The experiment was succesfull and the image is compared to the numerical results obtained using the BEM implementation.

Throughout the project, many technical details of the implementation were continually changed to find the best tools for the project. The first implementation of the ACA partially and fully pivoted algorithms were done in Octave/MATLAB, but the overhead time of these compiled languages was so large that no assessments of the improvement in time were possible. An implementation in C++ using the Armadillo and an external NURBS library was succesfull, but proved to be very difficult to compile and use, specially on machines running Windows. A new operational system was chosen for the development of the project, Ubuntu 16.04 or newer LTS, and a new language was also chosen, Julia. This very new language showed to be very promising as earlier Octave/MATLAB codes could be easily adapted and ran much faster. It was on this implementation that the shortcomings of the ACA+ procedure were identified and the Lagrangian polynomial interpolation method was implemented. The rapid development of the language from 2016 to 2020 was somewhat challenging as code written in Julia 0.4 and Julia 0.6 would not run on 0.7 and 1.0. But once the code was written in Julia 1.0, there has been little to no problem maintaining it. Before Julia 1.0, the language was not mature enough to provide stability.

Although the program is written in Julia, in further development new functionality is likely to be implemented in Python, as this language provides many more libraries and support than Julia. Much of the speed critical code is expected to remain in Julia, as the interface between them is straightforward.

Although the Lagrangian polynomial interpolation using Chebyshev nodes is used in the final implementation of the formulation, other approximation techniques can be considered. One of them is the matrix completion by penalizing the nuclear norm of the approximation. This is shown to produce low-rank estimations of a data matrix with many incomplete entries. This idea may be more time efficient than the Lagrangian polynomial interpolation, but may also be memory intensive. Therefore, an implementation of this approximation is encouraged.

Bibliography

ANDRADE, M.; BUIOCHI, F.; ADAMOWSKI, J. Finite element analysis and optimization of a single-axis acoustic levitator. *IEEE Transactions on Ultrasonics, Ferroelectrics and Frequency Control*, v. 57, n. 2, p. 469–479, fev. 2010. ISSN 0885-3010. Disponível em: <http://ieeexplore.ieee.org/document/5417206/>. Quoted on page 89.

ANDRADE, M. A. B. et al. Nonlinear characterization of a single-axis acoustic levitator. *Review of Scientific Instruments*, v. 85, n. 4, p. 045125, abr. 2014. ISSN 0034-6748, 1089-7623. Disponível em: <http://aip.scitation.org/doi/10.1063/1.4872356>. Quoted on page 89.

ARNDT, D. et al. The `deal.II` library, version 9.1. *Journal of Numerical Mathematics*, v. 27, n. 4, p. 203–213, 2019. Disponível em: <https://dealii.org/deal91-preprint.pdf>. Quoted 2 times in pages 66 and 68.

BACKUS, J. The history of fortran i, ii, and iii. *IEEE Annals of the History of Computing*, v. 20, n. 4, p. 68–78, Oct 1998. ISSN 1058-6180. Quoted 2 times in pages 4 and 63.

BEBENDORF, M. Approximation of boundary element matrices. *Nummerical Mathematics*, v. 86, p. 565–589, 2000. Quoted on page 55.

BEBENDORF, M. *Hierarchical Matrices - A Means to Efficiently Solve Elliptic Boundary Value Problems*. [S.l.]: Springer, 2008. Quoted on page 53.

BEBENDORF, M.; RJASANOW, S. Adaptive low-rank approximation of collocation matrices. *Computing*, v. 70, p. 1–24, 2003. Quoted on page 55.

BORDEN, M. J. et al. Isogeometric finite element data structures based on Bézier extraction of NURBS. *International Journal for Numerical Methods in Engineering*, v. 87, n. 1-5, p. 15–47, jul. 2011. ISSN 00295981. Disponível em: <http://doi.wiley.com/10.1002/nme.2968>. Quoted 2 times in pages 41 and 42.

BORM, S.; GRASEDYCK, L.; HACKBUSH, W. Introduction to hierarchical matrices with applications. *Engineering Analysis with Boundary Elements*, v. 27, p. 405–422, 2003. Quoted on page 55.

BRANCATI, A. *Boundary Element Method for Fast Solution of Acoustic Problems: Active and Passive Noise Control*. Tese (Doutorado) — Imperial College London, 2010. Quoted 2 times in pages 17 and 48.

BRANCATI, A.; ALIABADI, M. H. Boundary element simulations for local active noise control using an extended volume. *Engineering analysis with boundary elements*, v. 36, p. 190–202, 2012. Quoted on page 8.

BROOKS, F. P. *The Mythical Man-Month (Anniversary Ed.)*. USA: Addison-Wesley Longman Publishing Co., Inc., 1995. ISBN 0201835959. Quoted on page 64.

BUZOGANY, A. F. Análise de problemas tridimensionais usando o método dos elementos de contorno com expansão em multipolos. nov. 2017. Accepted: 2018-04-24T13:21:58Z. Disponível em: <https://bdm.unb.br/handle/10483/19999>. Quoted on page 69.

CABRAL, J. J. S. P.; WROBEL, L. C.; BREBBIA, C. A. A bem formulation using b-splines: I - uniform blending functions. *Engineering Analysis with Boundary Elements*, v. 7(3), 1990. Quoted on page 7.

CABRAL, J. J. S. P.; WROBEL, L. C.; BREBBIA, C. A. A bem formulation using b-splines: Ii - multiple knots and non-uniform blending functions. *Engineering Analysis with Boundary Elements*, v. 8(1), 1991. Quoted on page 7.

CAMPOS, L. S. MÉTODO DOS ELEMENTOS DE CONTORNO ISOGEOMÉTRICOS ACELERADO PELA APROXIMAÇÃO CRUZADA ADAPTATIVA. p. 124. Quoted on page 57.

CAMPOS, L. S.; ALBUQUERQUE Éder Lima de; WROBEL, L. C. An ACA accelerated isogeometric boundary element analysis of potential problems with non-uniform boundary conditions. *Engineering Analysis with Boundary Elements*, v. 80, p. 108–115, 2017. ISSN 09557997. Quoted on page 9.

CATALDO, E.; SAMPAIO, R.; NICOLATO, L. Uma discussão sobre modelos mecânicos de laringe para síntese de vogais. ENGEVISTA, 2004. Quoted 2 times in pages xii and 76.

CLéMENT, P. et al. Acoustic analysis of the vocal tract during vowel production by finite-difference time-domain method. Journal of the Voice, 2007. Quoted 4 times in pages xv, 76, 77, and 81.

COSTA, E. S.; BORGES, E. N. M.; AFONSO, M. M. Metodo de elementos de contorno aplicado a radiacao acustica. *Congresso Iberoamericano de Acustica - FIA*, 2008. Quoted on page 73.

DHANDOLE, S. D.; MODAK, S. V. Review of vibro-acoustics analysis procedures for prediction of low frequency noise inside a cavity. *IMAC-XXV: Conference and Exposition on Structural Dynamics*, 2007. Quoted on page 2.

DOMINGUEZ, J. *Boundary Elements in Dynamics*. Av. Reina Mercedes, s/n, 41012 Seville, Spain: Escuela Superior de Ingenieros Industriales Universidad de Sevilla, 1993. Quoted 2 times in pages 20 and 67.

EKHOLM, E.; PAPAGIANNIS, G. C.; CHAGNON, F. P. Relating objective measurements to expert evaluation voice quality in western classical singing" critical perceptual parameters. *Journal of Voice*, v. 12(2), p. –, 1998. Quoted on page 75.

ESNORFF, O. V. Challenges in technical acoustics: What can be computed today. *LS-DYNA*, 2008. Quoted on page 2.

FERREIRA, A. C. Dissertação de mestrado em ciências mecânicas: Análise harmônica de cavidades acústicas pelo método dos elementos de contorno direto. ENM.DM - 119 A/07. Universidade de Brasília, Faculdade de Tecnologia, Departamento de engenharia mecânica, 2014. Quoted on page 76.

FERREIRA, A. C. Analytical and numerical modeling of vocal tract in vowel phonation. CILAMCE: XXXVI Ibero-Latin American Congress on Computational Methods in Engineering, 2015. Quoted on page 84.

FREE SOFTWARE FOUNDATION. *GNU General Public License*. 2007. Disponível em: <http://www.gnu.org/licenses/gpl.html>. Quoted on page 65.

GIBERT, R. J. *Vibrations des structures - Interactions avec les fluides - Sources d'excitation aléatoires*. [S.l.]: Eyrolles, 1988. Quoted on page 70.

GOLDBERG, D. E. Real-coded genetic algorithms, virtual alphabets, and blocking. Complex Systems, v. 5, n. 2, p. 139–168, 1991. Quoted on page 78.

GONG, Y.; DONG, C.; BAI, Y. Evaluation of nearly singular integrals in isogeometric boundary element method. *Engineering Analysis with Boundary Elements*, v. 75, p. 21–35, fev. 2017. ISSN 09557997. Disponível em: <http://linkinghub.elsevier.com/retrieve/pii/S0955799716302909>. Quoted on page 23.

GOR'KOV, L. P. On the Forces Acting on a Small Particle in an Acoustical Field in an Ideal Fluid. *Soviet Physics Doklady*, v. 6, p. 773, mar. 1962. Disponível em: <http://adsabs.harvard.edu/abs/1962SPhD....6..773G>. Quoted on page 87.

GWYER, A. G. C. Improvements in the Schlieren method. *Journal of Scientific Instruments*, v. 11, n. 1, p. 31–31, jan. 1934. ISSN 0950-7671. Publisher: IOP Publishing. Disponível em: <https://doi.org/10.1088%2F0950-7671%2F11%2F1%2F116>. Quoted on page 88.

HACKBUSH, W. A sparse matrix arithmetic based on h-matrices. *Computing*, v. 62, p. 89–108, 1999. Quoted on page 55.

HAUPT, R. L.; HAUPT, S. E. *Practical Genetic Algorithm*. [S.l.]: John Wiley G. Sons Inc, 1998. Quoted on page 77.

HENRICH, N.; SMITH, J.; WOLFE, J. Vocal tract resonances in singing: Strategies used by sopranos, altos, tenors, and baritones. *journal of the acoustical society of america*, v. 129(2), p. –, 2011. Quoted on page 75.

HOLLAND, J. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. University of Michigan Press, 1975. ISBN 9780472084609. Disponível em: <https://books.google.com.br/books?id=JE5RAAAAMAAJ>. Quoted on page 77.

KING, A. A. Notes of the translator. sketch of the analytical engine invented by charles babbage. *Scientific Memoirs Selected from the Transactions of Foreign Academies of Science and Learned Societies*, III, 1843. ISSN 0034-6748, 1089-7623. Quoted on page 3.

KIRKUP, S. M. *The Boundary Element Method in Acoustics*. [S.l.]: Integrated Sound Software, 2007. Quoted 2 times in pages 20 and 67.

KOPUZ, S.; LALOR, N. Analysis of interior acoustic fields using the finite element method and the boundary element method. *Applied Acoustics*, v. 45, p. 193–210, 1995. Quoted on page 2.

KURZ, S.; RAIN, O.; RJASANOW, S. The adaptive cross-approximation technique for the 3-d boundary-element method. *IEEE TRANSACTIONS ON MAGNETICS*, v. 38(2), p. 421–424, 2002. Quoted on page 8.

KURZ, S.; RAIN, O.; RJASANOW, S. *Fast boundary element methods in computational eletromagnetism*. [S.l.]: Springer, 2007. Quoted on page 8.

LIGHTHILL, M. *Waves in Fluids*. Cambridge University Press, 1978. ISBN 9780521216890. Disponível em: <https://books.google.com.br/books?id=oVXTngEACAAJ>. Quoted on page 12.

LIU, Y. *Fast Multipole Boundary Element Method Theory and Applications in Engineering*. [S.l.]: Campridge University Press, 2009. Quoted on page 67.

LIU, Y. J.; NISHIMURA, N. The fast multipole boundary element method for potential problems: A tutorial. *Engineering Analysis with Boundary Elements*, v. 30, p. 371–381, 2006. Quoted on page 8.

MALLARDO, V.; ALIABADI, M. H.; BRANCATI, A. An accelerated bem for simulation of noise control in the aircraft cabin. *Aerospace Science and Technology*, v. 23, p. 418–428, 2012. Quoted on page 9.

MARUSSIG, B. et al. Fast isogeometric boundary element method based on independent field approximation. *Computer Methods Applied to Mechanical Engineering*, 2015. Quoted on page 9.

MARZO, A.; BARNES, A.; DRINKWATER, B. W. TinyLev: A multi-emitter single-axis acoustic levitator. *Review of Scientific Instruments*, v. 88, n. 8, p. 085105, ago. 2017. ISSN 0034-6748, 1089-7623. Disponível em: <http://aip.scitation.org/doi/10.1063/1.4989995>. Quoted on page 87.

MORGADO, P. L. d. A. Matrizes hierárquicas e a aproximação cruzada adaptativa. nov. 2017. Accepted: 2018-04-17T13:17:40Z. Disponível em: <http://bdm.unb.br/handle/10483/19946>. Quoted on page 57.

NEUMANN, M. *Hierarchical Matrices An Implementation in Python*. Dissertação (Mestrado) — University of Zurich, 2017. Quoted 2 times in pages 60 and 61.

NICOLAIDIS, T. R. *Análise de escoamentos de fluidos viscosos utilizando o método dos elementos de contorno*. [S.l.]: Departamento de Engenharia Mecânica, UnB, 2018. Quoted on page 62.

OSSENDRIJVER, M. Ancient Babylonian astronomers calculated Jupiter's position from the area under a time-velocity graph. *Science*, v. 351, n. 6272, p. 482–484, jan. 2016. ISSN 0036-8075, 1095-9203. Publisher: American Association for the Advancement of Science Section: Report. Disponível em: <https://science.sciencemag.org/content/351/6272/482>. Quoted on page 3.

PEAKE, M.; TREVELYAN, J.; COATES, G. Extended isogeometric boundary element method (XIBEM) for two-dimensional Helmholtz problems. *Computer Methods in Applied Mechanics and Engineering*, v. 259, p. 93–102, jun. 2013. ISSN 00457825. Disponível em: <http://linkinghub.elsevier.com/retrieve/pii/S0045782513000674>. Quoted on page 42.

PEAKE, M.; TREVELYAN, J.; COATES, G. Extended isogeometric boundary element method (XIBEM) for three-dimensional medium-wave acoustic scattering problems. *Computer Methods in Applied Mechanics and Engineering*, v. 284, p. 762–780, fev. 2015. ISSN 00457825. Disponível em: <http://linkinghub.elsevier.com/retrieve/pii/S0045782514004095>. Quoted on page 42.

PEAKE, M. J.; TREVELYAN, J.; COATES, G. Extended isogeometric boundary element method (xibem) for two-dimensional helmholtz problems. *Computer Methods Applied to Mechanical Engineering*, v. 259, p. 93–102, 2013. Quoted 2 times in pages 7 and 25.

PEAKE, M. J.; TREVELYAN, J.; COATES, G. Extended isogeometric boundary element method (xibem) for three-dimensional medium-wave acoustic scattering problems. *Computer Methods Applied to Mechanical Engineering*, v. 284(1), 2015. Quoted on page 7.

PIEGL, L.; TILLER, W. *The NURBS Book*. [S.l.]: Springer, 1997. Quoted on page 33.

RAHMAT-SAMII, Y.; MICHIELSSEN, E. *Electromagnetic Optimization by Genetic Algorithms*. Wiley, 1999. (Wiley Series in Microwave and Optical Engineering). ISBN 9780471295457. Disponível em: <https://books.google.com.br/books?id=0x5TAAAAMAAJ>. Quoted on page 78.

RAYMOND, E. S. *The Cathedral and the Bazaar*. 1st. ed. Sebastopol, CA, USA: O'Reilly & Associates, Inc., 1999. ISBN 1565927249. Quoted on page 66.

RAYMOND, E. S. *The Art of UNIX Programming*. [S.l.]: Pearson Education, 2003. ISBN 0131429019. Quoted on page 64.

RODRIGUES, A. P. S. P. Dissertação de mestrado em ciências mecânicas: Parametrização e simulação numérica da turbina hidrocinética - otimização via algoritmos genéticos. ENM.DM - 119 A/07. Universidade de Brasília, Faculdade de Tecnologia, Departamento de engenharia mecânica, 2007. Quoted on page 78.

ROGUS, B. *The adaptive cross approximation algorithm applied to electromagnetics scattering by bodies of revolution*. Dissertação (Mestrado) — Duquesne University, 2008. Quoted on page 8.

SAMSUDIN, D. et al. *Study of Schlieren Optical Visualization Basics Technique and the Principle*. 2015. Disponível em: <https://www.scientific.net/amm.773-774.506>. Quoted on page 88.

SCHLIEREN Optics. 2020. Library Catalog: sciencedemonstrations.fas.harvard.edu. Disponível em: <https://sciencedemonstrations.fas.harvard.edu/presentations/schlieren-optics>. Quoted 2 times in pages xii and 88.

SIMPSON, R. N. et al. Acoustic isogeometric boundary element analysis. *Computer Methods Applied to Mechanical Engineering*, v. 269, p. 265–290, 2014. Quoted on page 7.

ŚMIGAJ, W. et al. Solving boundary integral problems with BEM++. *ACM Transactions on mathematical software*, 41, n. 2, p. 6:1–6:40, 2015. Quoted on page 66.

TELLES, J. C. F. A self-adaptive co-ordinate transformation for efficient numerical evaluation of general boundary element integrals. *International Journal for Numerical Methods in Engineering*, v. 24, n. 5, p. 959–973, 1987. Disponível em: <http://onlinelibrary.wiley.com/doi/10.1002/nme.1620240509/full>. Quoted 2 times in pages 23 and 104.

TITZE, I. Acoustic interpretation of resonant voice. *Journal of Voice*, v. 15(4), p. 519–528, 2001. Quoted on page 75.

WROBEL, L. C. *The Boundary Element Method - Applications in Thermo-Fluids and Acoustics*. [S.l.]: John Wiley & Sons, 2001. Quoted 2 times in pages 3 and 20.

ZHANG, Y.; QU, W.; GU, Y. Evaluation of Singular and Nearly Singular Integrals in the BEM with Exact Geometrical Representation. *Journal of Computational Mathematics*, v. 31, n. 4, p. 355–369, jul. 2013. ISSN 02549409. Disponível em: <http://www.global-sci.org/jcm/volumes/v31n4/pdf/314-355.pdf>. Quoted on page 23.

ZHOU, X. et al. Improved vocal tract reconstruction and modeling using an image super-resolution technique. *The Journal of the Acoustical Society of America*, v. 133, n. 6, p. EL439–EL445, jun. 2013. ISSN 0001-4966. Quoted on page 76.

Appendix

A Source code

Given the scope of the BEM_base, it was deemed better to make it available in an online repository with a GPL. To access the source code of BEM_base, please refer to the Git repository: https://github.com/alvarocafe/BB.

B Boundary discretization using Lagrangian elements

B.1 Introduction

For the purposes of understanding the fundamental solutions and the effects of the discretization technique chosen, the integral equations which will be studied are the ones present in Eq. (3.10):

$$I_1 = \int_\Gamma \phi^*(x', x, k) \frac{\partial \phi(x)}{\partial \mathbf{n}} d\Gamma \tag{B.1}$$

and

$$I_2 = \int_\Gamma \phi(x) \frac{\partial \phi^*}{\partial \mathbf{n}}(x', x, k) d\Gamma, \tag{B.2}$$

where the collocation point is x', the field point is x and ϕ^* is the fundamental solution of the Helmholtz equation, or other partial differential equation owning a free-space solution. The domain of this partial differential equation is given by Ω with boundary S, as shown in Figure A.1. The boundary will be divided into elements and the approximated boundary described by this elements is henceforth called $\Gamma = \sum_{i=1}^{NE} \Gamma_i$, where NE is the number of elements. If the boundary is described by a single element, or if the only section of the boundary of interest is given by this element, one may simply refer to the boundary as Γ.

The integration will be carried out in an element denoted by Γ, which will be described by different curves: linear, quadratic, and quadratic Bézier. The integration will be carried out numerically using Gaussian quadrature. The analysis will be performed for a circular boundary in the 2D cases and in a half cylinder for the 3D cases. Where possible, only one element will be used to describe this boundary, but this is often not possible.

B.2 Two-dimensional linear elements

Linear elements describe the boundary as a collection of straight lines. Linear and constant elements both uses a linear approximation to describe the boundary, but

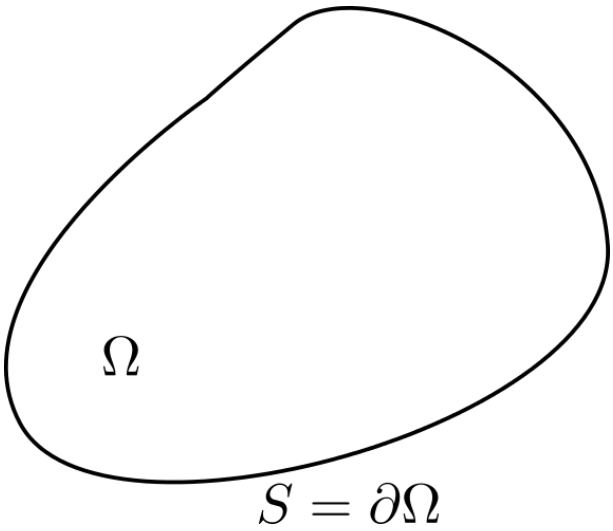

Figure B.1 – Domain of the problem and its boundary

constant elements don't use this interpolation to describe the field variable, instead this variable is considered to be constant throughout the element. The linear element uses the interpolation to describe the integration variable and the differences between these two elements is shown below.

B.2.1 Constant elements

The bidimensional constant element is a straight line and the integration variable is constant along the element and applied to a node in its center. The geometry of the element is a straight line that cross two points (x_1, y_1) and (x_2, y_2), which is described as

$$x = N_1(\xi)x_1 + N_2(\xi)x_2, \tag{B.3}$$

$$y = N_1(\xi)y_1 + N_2(\xi)y_2, \tag{B.4}$$

where N_1 and N_2 are the linear shape functions which will describe the geometry of the boundary in terms of the parametric variable $\xi \in [-1, 1]$. Suitable shape functions are:

$$N_1 = \frac{1}{2}(1 - \xi) \tag{B.5}$$

and

$$N_2 = \frac{1}{2}(1 + \xi). \tag{B.6}$$

The shape functions can be viewed in Figure A.2.

A set of Gaussian points and weights are created in the interval $[-1, 1]$ to evaluate the shape functions using the Gaussian quadrature. Figure A.3 shows the element with the geometric points (x, y), Gaussian points and the node, in the case of a constant element with geometrical points $p1 = (0, 0), p2 = (1, 1)$.

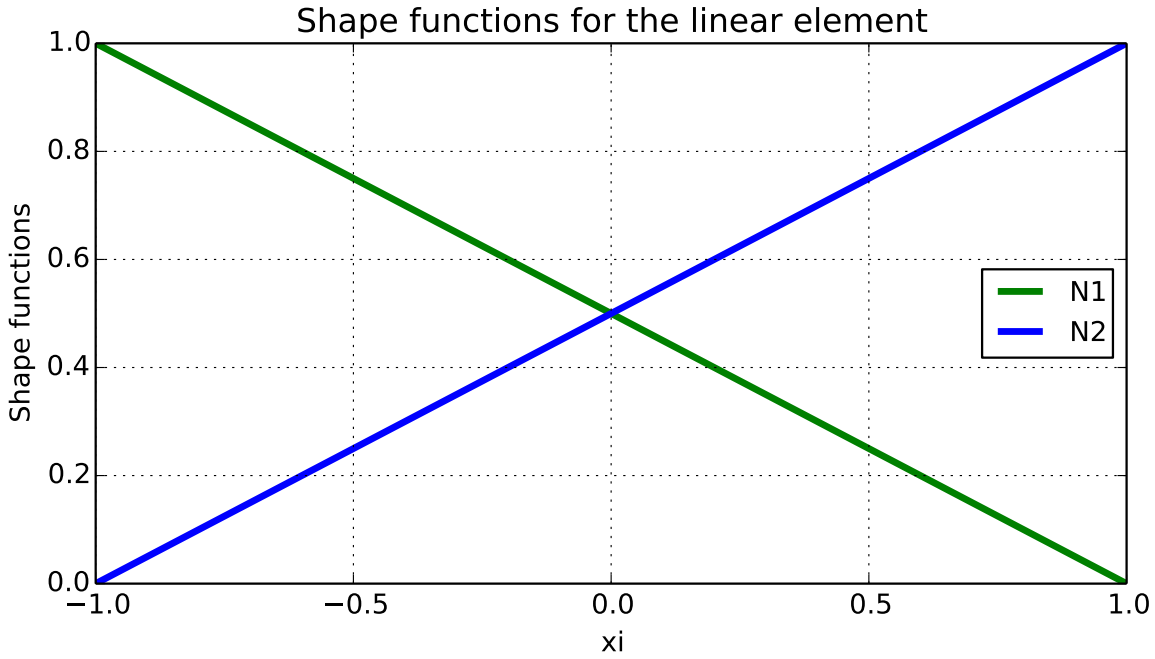

Figure B.2 – Linear Langrange polynomials used in linear and constant elements. In constant element, linear shape functions are used to interpolate only the geometric variables.

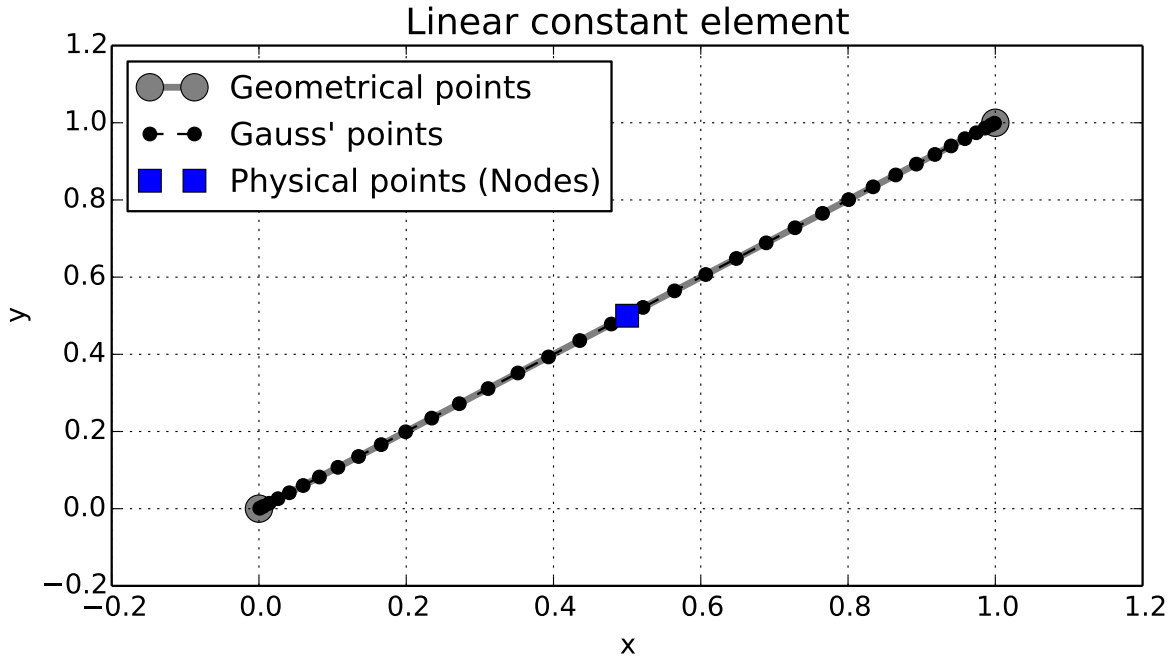

Figure B.3 – 2D constant element for 36 Gauss' points

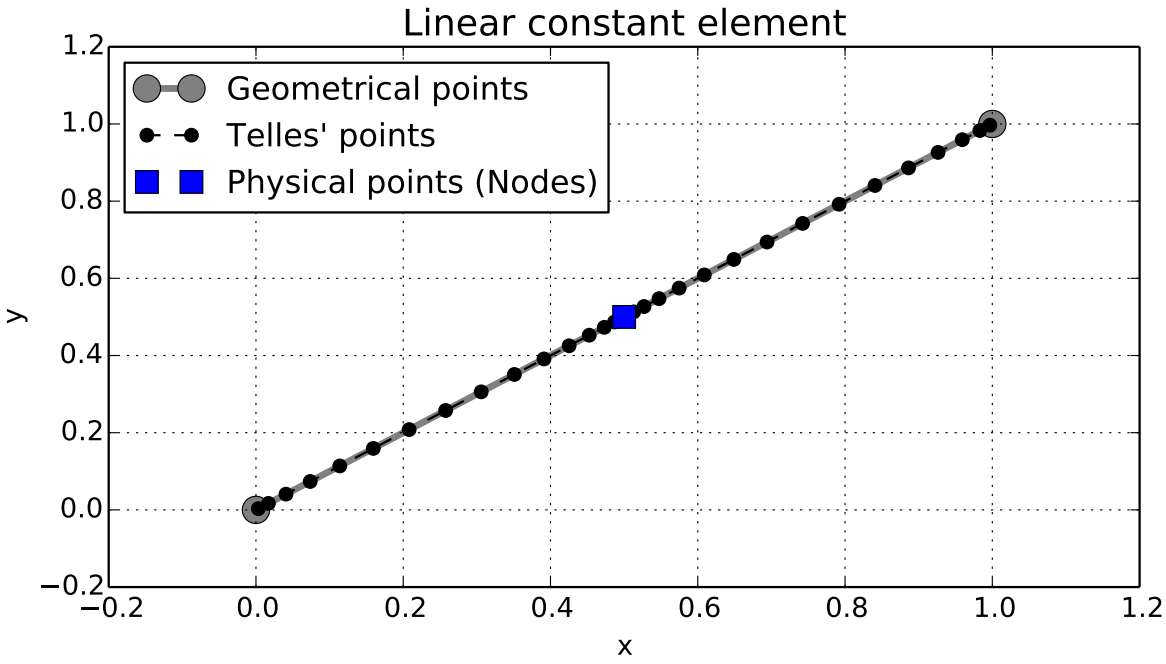

Figure B.4 – 2D linear constant element for 36 Gauss' points after the Telles transformation

One may notice how the Gauss' points are mostly concentrated in the end of the element. This is undesirable, as the physical node is located at the center of the line and most of the fundamental solution contribution will be concentrated at that point. There is an effective way to redistribute the Gauss' points around the singularity by using the Telles transformation (TELLES, 1987). This transformation changes the position of the quadrature points around the singularity. Figure A.4 shows the distribution of Gauss' points after the Telles transformation and it's possible to see that most of them are now concentrated around the physical node.

The impact of this transformation can be seen in Figure A.5, where the fundamental solution for the Helmholtz equation along the constant element is shown. It is clear that the behaviour of the Green's function is better described after using the Telles transformation.

Now that the element has been described in the parametric domain, the integral equations of interest are:

$$I_1 = \int_\Gamma \phi^*(x', x, k) \frac{\partial \phi}{\partial \mathbf{n}}(x) d\Gamma \tag{B.7}$$

and

$$I_2 = \int_\Gamma \phi(x) \frac{\partial \phi^*}{\partial \mathbf{n}}(x', x, k) d\Gamma. \tag{B.8}$$

Even though the geometry is being approximated by this straight element, the velocity potential and flux are constant throughout the element, and, as such, can be

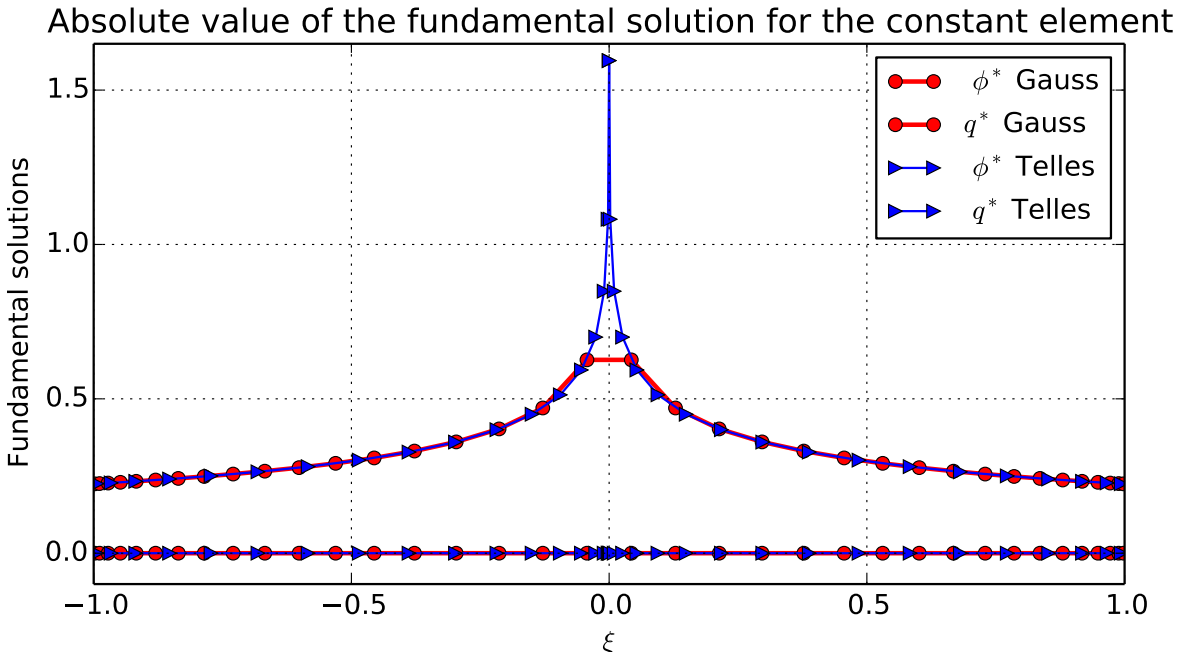

Figure B.5 – Fundamental solution of the Helmholtz equation for the constant element for 36 Gauss' points before and after the Telles transformation

written out of the integrals in equations (A.11) and (A.12),

$$I_1 = \frac{\partial \phi}{\partial \mathbf{n}} \int_\Gamma \phi^*(x', x, k) d\Gamma \tag{B.9}$$

and

$$I_2 = \phi \int_\Gamma \frac{\partial \phi^*}{\partial \mathbf{n}}(x', x, k) d\Gamma, \tag{B.10}$$

where now $\bar{\phi}$ and $\frac{\partial \bar{\phi}}{\partial \mathbf{n}}$ are the nodal values of the potential velocity and its flux and are considered constant along the element, applied to its center.

The integration domain is transformed from Γ to ξ so that the integration can take place in the parametric domain. The transformation is carried out by applying a Jacobian in equations (A.11) and (A.12) to obtain

$$I_1 = \frac{\partial \phi}{\partial \mathbf{n}} \int_{-1}^{1} \phi^*(x', \xi, k) \frac{\partial \Gamma}{\partial \xi} d\xi \tag{B.11}$$

and

$$I_2 = \phi \int_{-1}^{1} \frac{\partial \phi^*}{\partial \mathbf{n}}(x', \xi, k) \frac{\partial \Gamma}{\partial \xi} d\xi, \tag{B.12}$$

where the Jacobian $\frac{\partial \Gamma}{\partial \xi} = L/2$ and L is the length of the element.

The integrals are then approximated using Gaussian quadrature

$$I_1 = \frac{\partial \phi}{\partial \mathbf{n}} \sum_{i=1}^{NPG} w_i \phi^*(x', \xi_i, k) \frac{\partial \Gamma}{\partial \xi_i}, \tag{B.13}$$

$$I_2 = \phi \sum_{i=1}^{NPG} w_i \frac{\partial \phi^*}{\partial \mathbf{n}}(x', \xi_i, k) \frac{\partial \Gamma}{\partial \xi_i}, \tag{B.14}$$

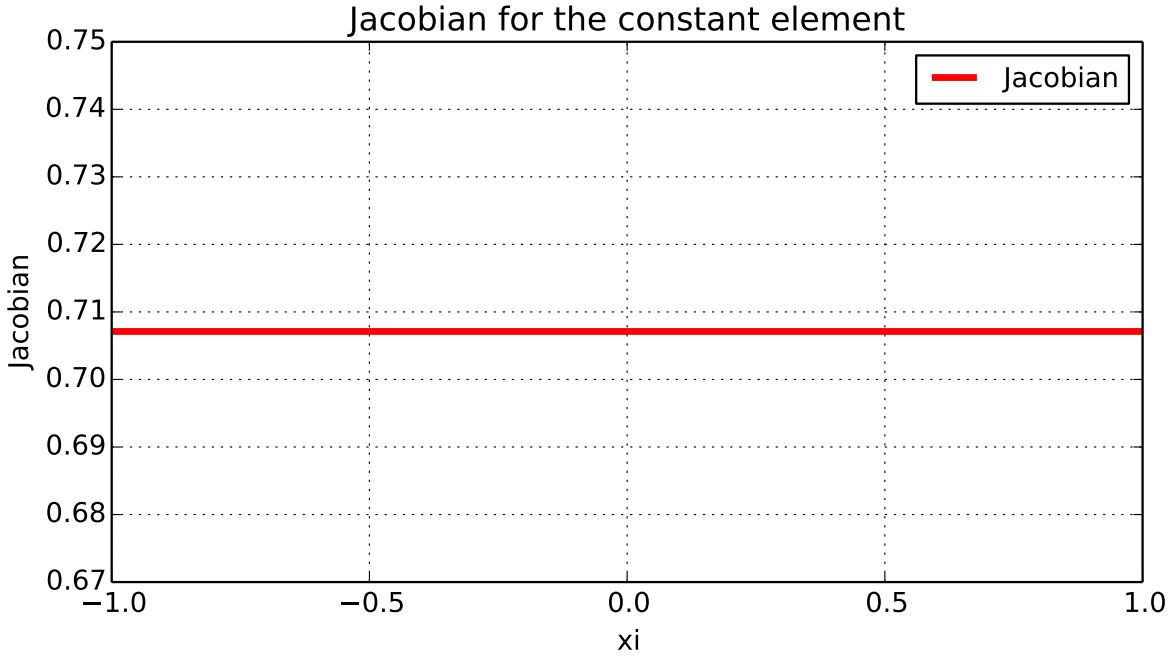

Figure B.6 – Jacobian for the linear element

where $i = 1, 2, ...NPG$ are the Gaussian points and w_i are the weights.

The boundary will be divided into elements $\Gamma = \sum_{i=1}^{NE} \Gamma_i$, where NE is the number of elements. Each element i has one and only one corresponding node, named node i, and the value of the acoustic potential is defined at this point. So, for the element Γ_i, there's only one node i where the acoustic potential ϕ_i and acoustic flux $\frac{\partial \phi_i}{\partial \mathbf{n}}$ is defined. The new equation of interest is now:

$$\frac{1}{2}\phi_j + \sum_{i=1}^{NE} \int_{\Gamma_i} \phi_i \frac{\partial {\phi^*}_{i,j}}{\partial \mathbf{n}} \frac{\partial \Gamma}{\partial \xi} d\xi = \sum_{i=1}^{NE} \int_{\Gamma_i} \frac{\partial \phi_i}{\partial \mathbf{n}} {\phi^*}_{i,j} \frac{\partial \Gamma}{\partial \xi} d\xi. \tag{B.15}$$

Each integral is evaluated as in Eqs. (A.13) and (A.14). The nodal values of ϕ_i and $\frac{\partial \phi_i}{\partial n}$ are constant throughout the element, and can be written out of the integral equation, thus:

$$\frac{1}{2}\phi_j + \sum_{i=1}^{NE} \phi_i \int_{\Gamma_i} \frac{\partial {\phi^*}_{i,j}}{\partial \mathbf{n}} \frac{\partial \Gamma}{\partial \xi} d\xi = \sum_{i=1}^{NE} \frac{\partial \phi_i}{\partial \mathbf{n}} \int_{\Gamma_i} {\phi^*}_{i,j} \frac{\partial \Gamma}{\partial \xi} d\xi. \tag{B.16}$$

It's possible now to define a matricial form of the integral equation by aggregating the terms of the left and right hand side as:

$$H_{ij} = \begin{cases} \frac{1}{2} + \int_{\Gamma_i} \frac{\partial {\phi^*}_{i,j}}{\partial \mathbf{n}} \frac{\partial \Gamma}{\partial \xi} d\xi & \text{, if } i = j \\ \int_{\Gamma_i} \frac{\partial {\phi^*}_{i,j}}{\partial \mathbf{n}} \frac{\partial \Gamma}{\partial \xi} d\xi & \text{, if } i \neq j \end{cases} \tag{B.17}$$

$$G_{ij} = \int_{\Gamma_i} {\phi^*}_{i,j} \frac{\partial \Gamma}{\partial \xi} d\xi \tag{B.18}$$

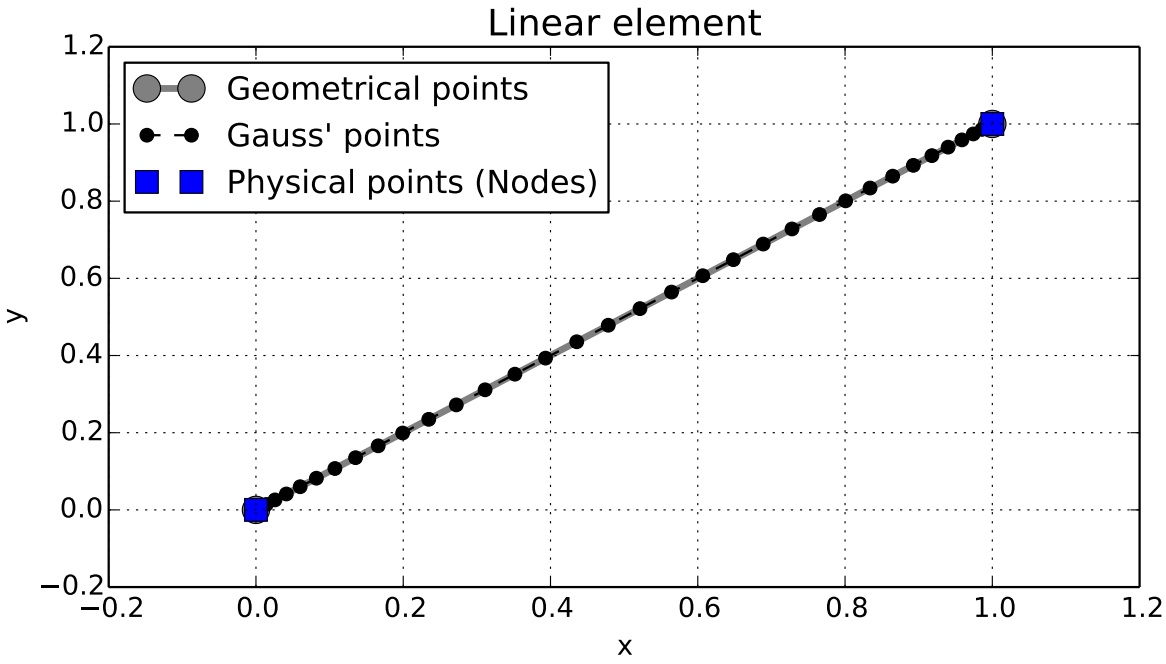

Figure B.7 – 2D linear element for 36 Gauss' points

And the matricial form of the equation is finally

$$\sum_{i=1}^{NE} H_{ij}\phi_i = \sum_{i=1}^{NE} G_{ij}\frac{\partial \phi_i}{\partial \mathbf{n}}. \tag{B.19}$$

B.2.2 Linear elements

If the node is located at each geometrical point of the line, the element is commonly called linear element and is shown in Figure A.7.

Now, not only is the geometry approximated by this linear interpolation, but the field variables will also be described by it. This means that

$$\phi = N_1(\xi)\phi_1 + N_2(\xi)\phi_2, \tag{B.20}$$

$$\frac{\partial \phi}{\partial \mathbf{n}} = N_1(\xi)\frac{\partial \phi_1}{\partial \mathbf{n}} + N_2(\xi)\frac{\partial \phi_2}{\partial \mathbf{n}}, \tag{B.21}$$

where N_1 and N_2 are the linear shape functions, ϕ_1 and ϕ_2 are the velocity potential at nodes 1 and 2 and q is shorthand for $\frac{\partial \phi}{\partial \mathbf{n}}$. From now on, it's better to express the coordinates of the nodes and the unknowns in their vector form, as now there's more than one node at each element. This will be valid for every other shape function except for the constant element. For the linear element, the variables are interpolated using the linear shape function:

$$[\phi] = \mathbf{N}\phi, \tag{B.22}$$

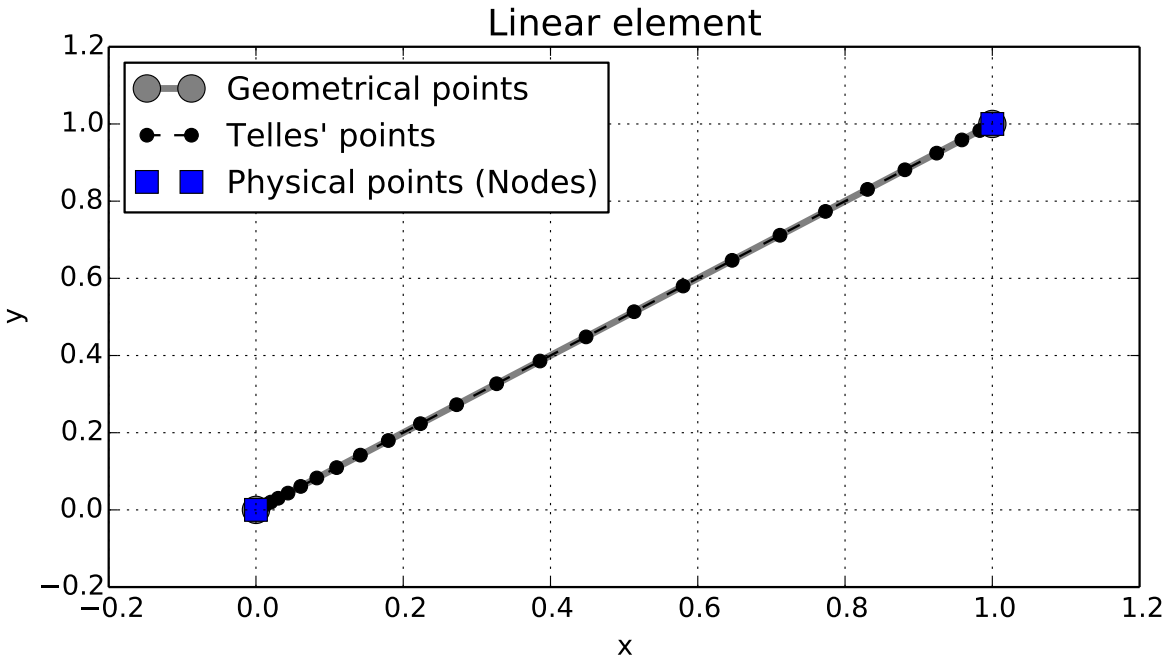

Figure B.8 – 2D linear element for 36 Gauss' points after the Telles transformation for the first physical node

$$\left[\frac{\partial \phi}{\partial \mathbf{n}}\right] = \mathbf{N}\frac{\partial \phi}{\partial \mathbf{n}}, \tag{B.23}$$

$$\mathbf{N} = [N_1(\xi) \quad N_2(\xi)], \tag{B.24}$$

$$\phi = \begin{bmatrix} \phi_1 \\ \phi_2 \end{bmatrix} \tag{B.25}$$

and

$$\frac{\partial \phi}{\partial \mathbf{n}} = \begin{bmatrix} \frac{\partial \phi_1}{\partial \mathbf{n}} \\ \frac{\partial \phi_2}{\partial \mathbf{n}} \end{bmatrix}. \tag{B.26}$$

Now there's two values for the acoustic pressure for the element. Substituting Eqs. (A.22) and A.23 into Eq. (3.10), one obtains:

$$c(x')\phi(x') + \int_\Gamma [N_1(\xi) \quad N_2(\xi)] \begin{bmatrix} \phi_1 \\ \phi_2 \end{bmatrix} \frac{\partial \phi^*(x', x, k)}{\partial n} d\Gamma = \int_\Gamma \frac{\partial \phi(x)}{\partial n} \phi^*(x', x, k) d\Gamma \tag{B.27}$$

The Telles transformation is applied to this element, if the node is on the first geometrical point, the integration points are moved closer to it, as shown in Figure A.8. The Telles transformation for the second node, located at the second geometrical point, the transformation is shown in Figure A.9. The fundamental solution in the parametric domain is shown in Figure A.10.

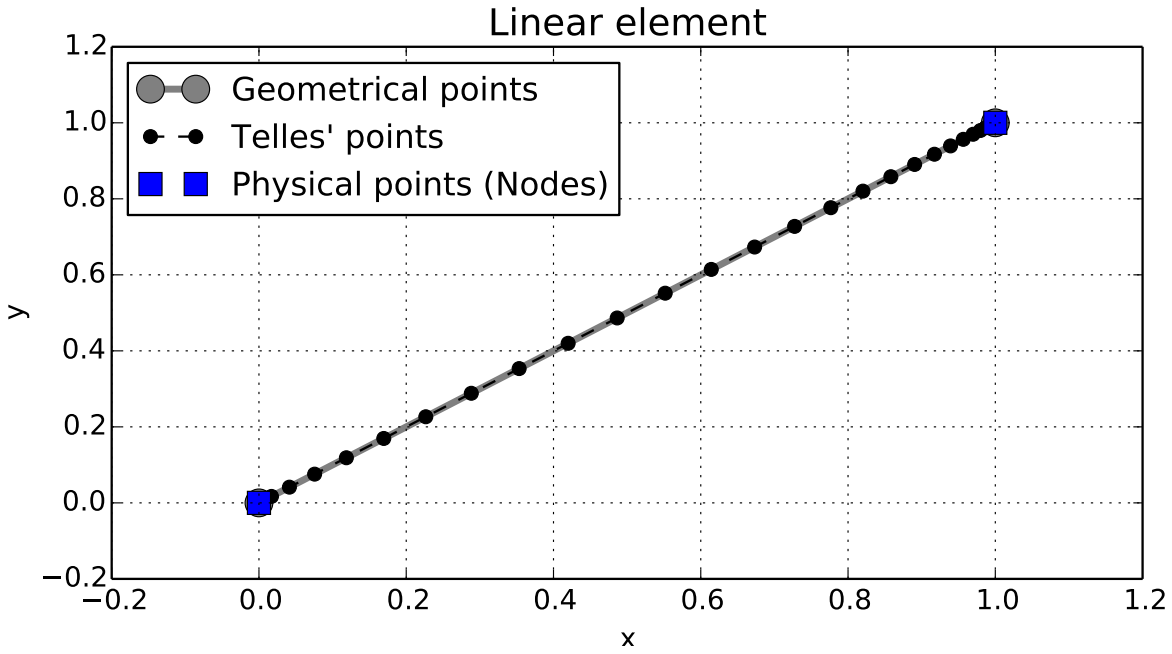

Figure B.9 – 2D linear element for 36 Gauss' points after the Telles transformation for the second physical node

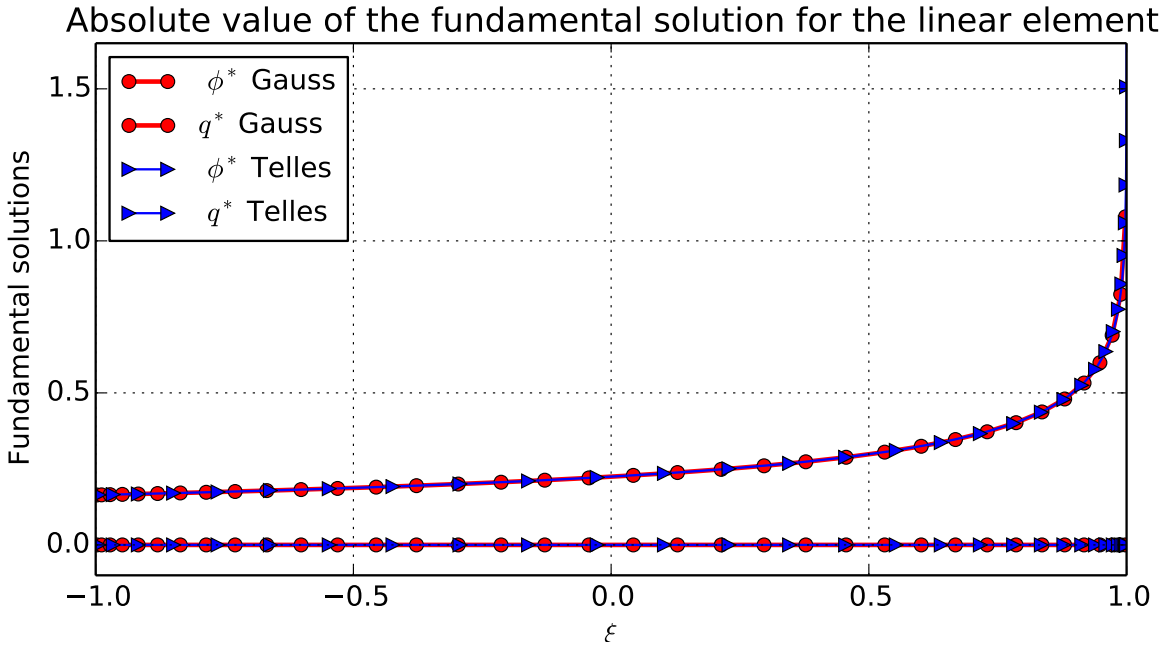

Figure B.10 – Fundamental solution of the Helmholtz equation for the linear element for 36 Gauss' points before and after the Telles transformation for the first node

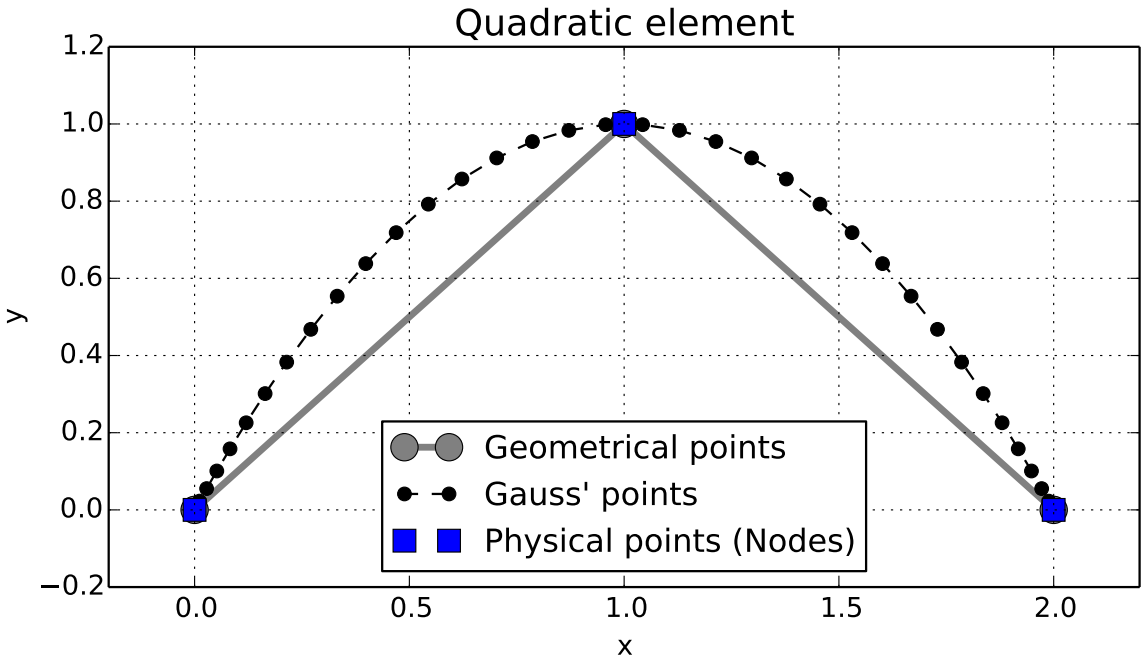

Figure B.11 – 2D quadratic element for 36 Gauss' points

B.3 Two-dimensional quadratic elements

The geometry is approximated by a parabolic or quadratic polynomial. This kind of approximation is very commom and the coordinates (x_1, y_1), (x_2, y_2), and (x_3, y_3) will be described by:

$$x = N_1(\xi)x_1 + N_2(\xi)x_2 + N_3(\xi)x_3 \tag{B.28}$$

and

$$y = N_1(\xi)y_1 + N_2(\xi)y_2 + N_3(\xi)y_3, \tag{B.29}$$

where N_1 and N_2 and N_3 are the quadratic shape functions which will describe the geometry of the boundary. Suitable shape functions are:

$$N_1 = \frac{\xi}{2}(1 - \xi), \tag{B.30}$$

$$N_2 = (1 - \xi)(1 + \xi), \tag{B.31}$$

and

$$N_3 = \frac{\xi}{2}(1 + \xi). \tag{B.32}$$

The geometry of the element is shown in Figure A.11 for the geometrical points $p1 = (0, 0), p2 = (1, 1), p3 = (2, 0)$. Some things worth noting is that the curve is not contained in the polygon formed by the geometrical points. Also worth noting is the fact that the geometrical points coincide with the nodes.

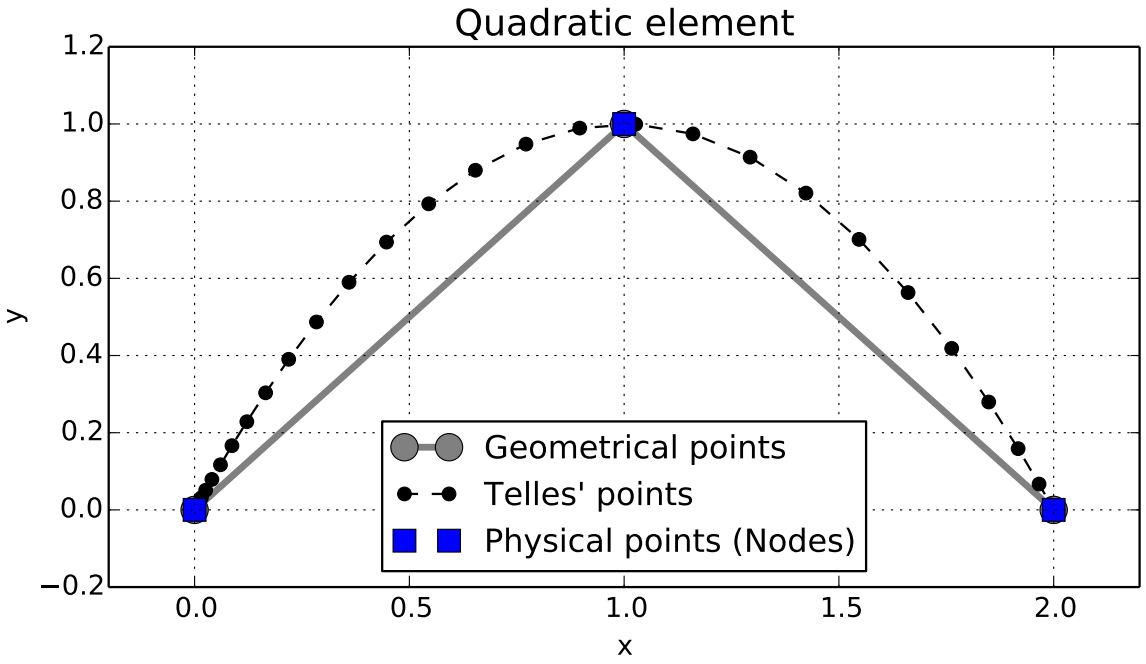

Figure B.12 – 2D quadratic element for 36 Gauss' points after the Telles transformation for the first physical node

To obtain the Jacobian of the transformation from the domain $[x, y]$ to ξ, it is necessary to derivate the shape functions with respect to ξ:

$$\frac{\partial N_1}{\partial \xi} = \frac{2\xi - 1}{2}, \tag{B.33}$$

$$\frac{\partial N_2}{\partial \xi} = -2\xi, \tag{B.34}$$

and

$$\frac{\partial N_3}{\partial \xi} = \frac{2\xi + 1}{2}. \tag{B.35}$$

Once this derivatives are obtained, one obtains the derivative of $[x, y]$ with respect to ξ, and calculate the Jacobian as

$$\frac{\partial x}{\partial \xi} = \frac{\partial N_1(\xi)}{\partial \xi} x_1 + \frac{\partial N_2(\xi)}{\partial \xi} x_2 + \frac{\partial N_3(\xi)}{\partial \xi} x_3, \tag{B.36}$$

$$\frac{\partial y}{\partial \xi} = \frac{\partial N_1(\xi)}{\partial \xi} y_1 + \frac{\partial N_2(\xi)}{\partial \xi} y_2 + \frac{\partial N_3(\xi)}{\partial \xi} y_3 \tag{B.37}$$

and, finally,

$$\frac{\partial \Gamma}{\partial \xi} = \sqrt{\left(\frac{\partial x}{\partial \xi}\right)^2 + \left(\frac{\partial y}{\partial \xi}\right)^2}. \tag{B.38}$$

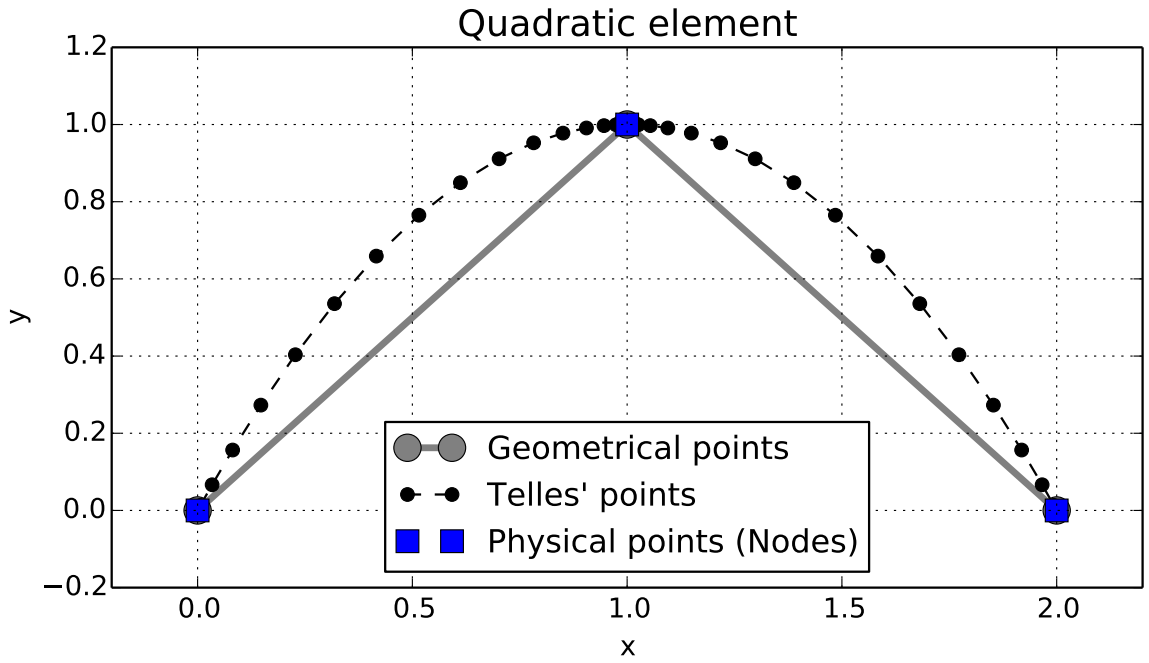

Figure B.13 – 2D quadratic element for 36 Gauss' points after the Telles transformation for the second physical node

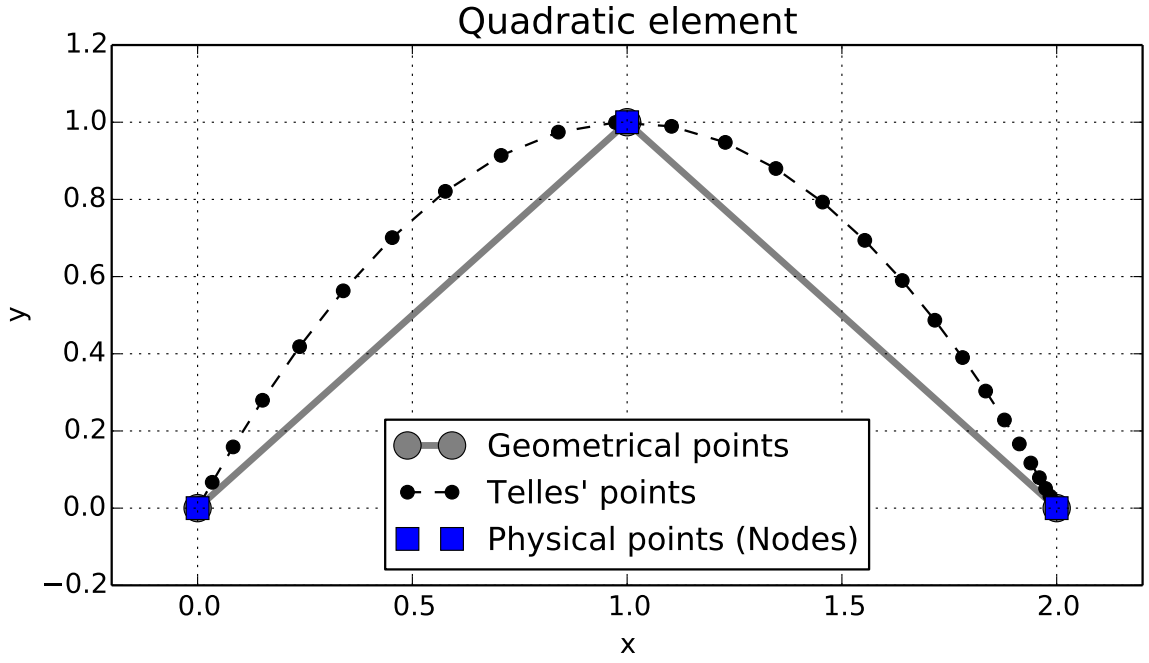

Figure B.14 – 2D quadratic element for 36 Gauss' points after the Telles transformation for the third physical node

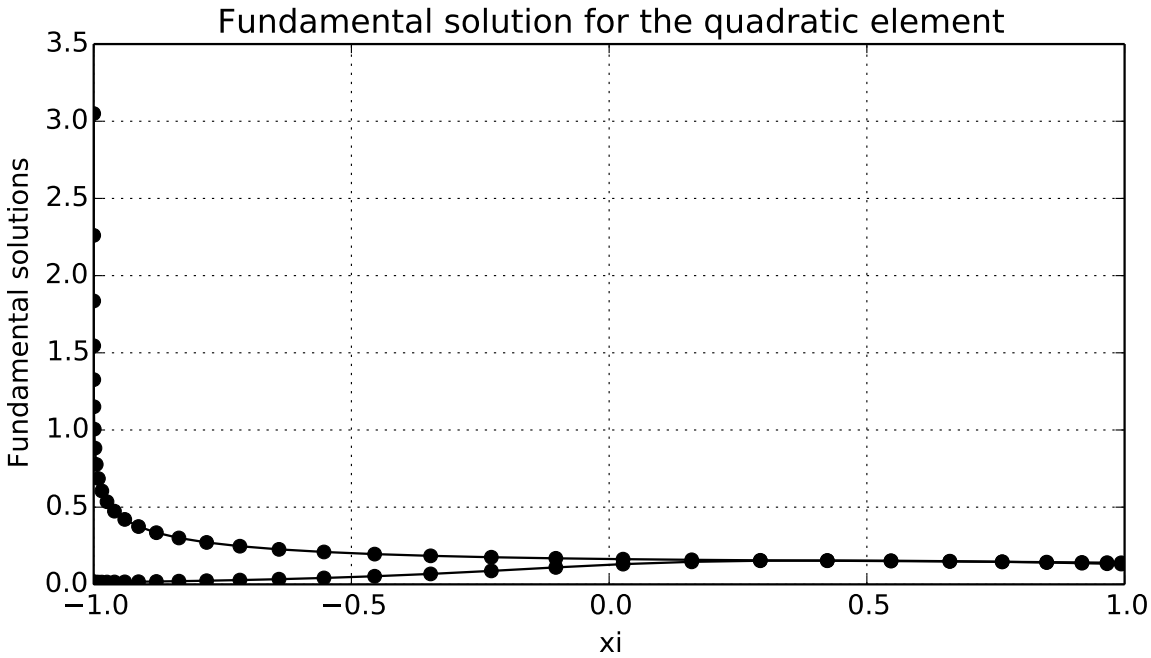

Figure B.15 – Fundamental solution of the Helmholtz equation for the quadratic element for 36 Gauss' points before and after the Telles transformation for the first node

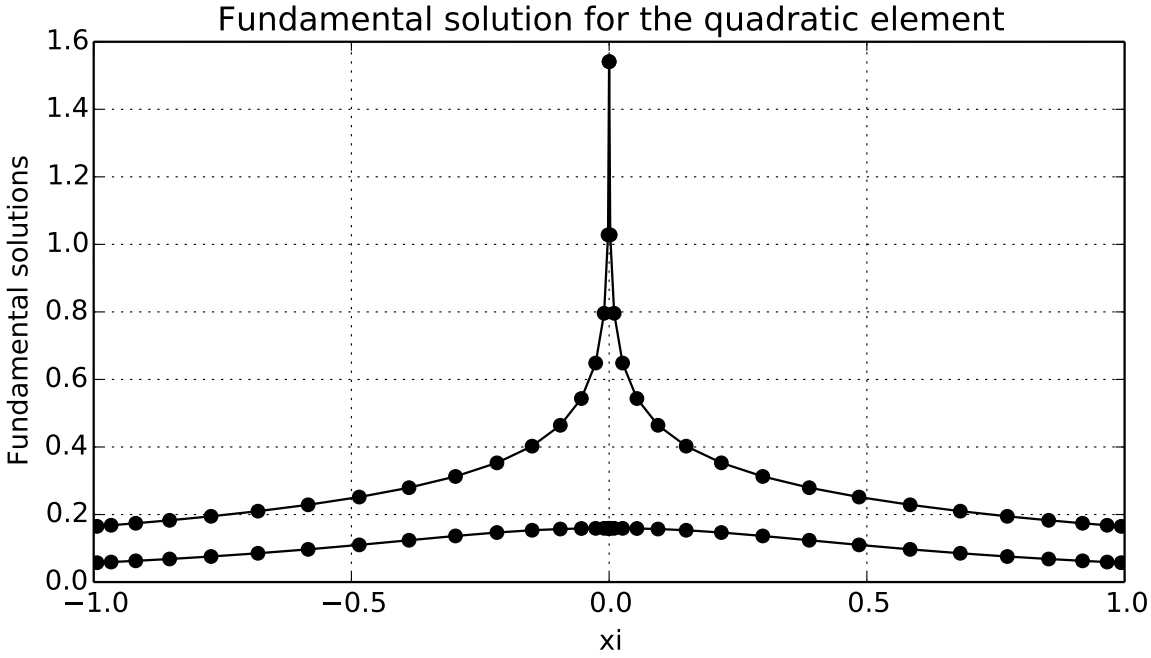

Figure B.16 – Fundamental solution of the Helmholtz equation for the quadratic element for 36 Gauss' points before and after the Telles transformation for the second node

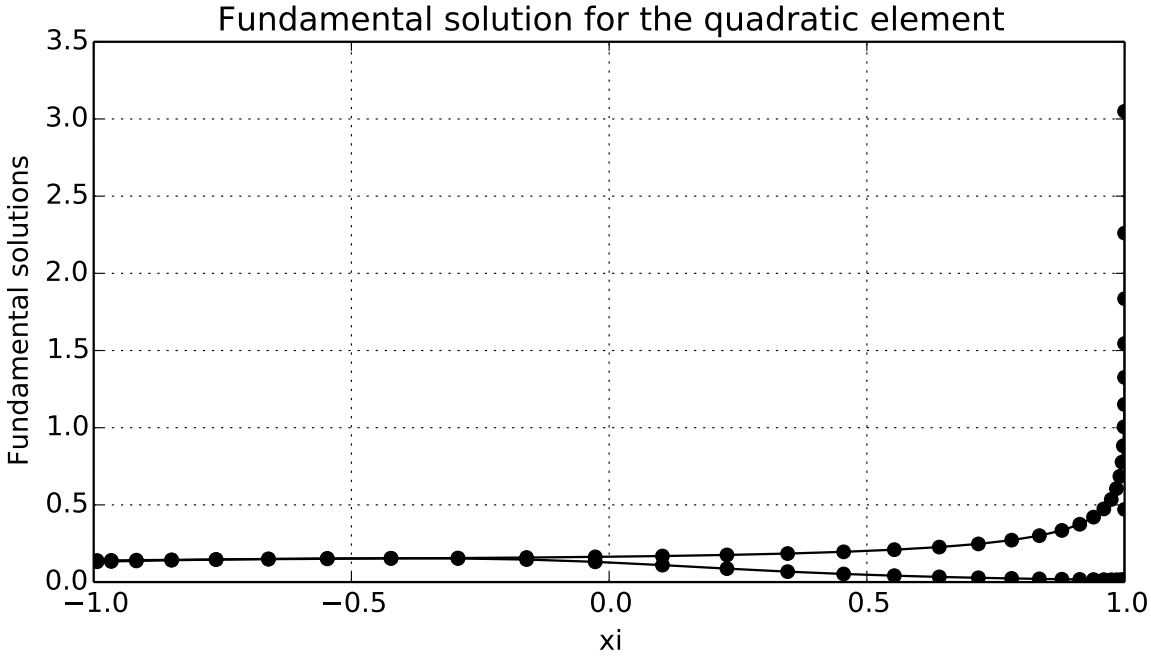

Figure B.17 – Fundamental solution of the Helmholtz equation for the quadratic element for 36 Gauss' points before and after the Telles transformation for the third node

The integration can now be performed in the domain of ξ using traditional numerical integration techniques such as Gaussian quadrature in Equations (A.39) and (A.40).

$$I_1 = \int_{-1}^{1} \frac{\partial \phi}{\partial \mathbf{n}} \phi^*(x', \xi, k) \frac{\partial \Gamma}{\partial \xi} d\xi \tag{B.39}$$

and

$$I_2 = \int_{-1}^{1} \phi \frac{\partial \phi^*}{\partial \mathbf{n}}(x', \xi, k) \frac{\partial \Gamma}{\partial \xi} d\xi, \tag{B.40}$$

where the Jacobian is given by Equation (A.38) and can be seen in Figure A.18.

B.4 Two-dimensional quadratic Bézier elements

The quadratic Bézier element is very similar to the quadratic element, but the shape functions are slightly different. The curve is still described by 3 points in the $[x, y]$ domain, but this are now called 'control points' and aren't necessarily contained in the curve, only the first and last control points are. The parametric domain is $\xi \in [0, 1]$. In the case of the quadratic Bézier curve, there are only three control points,

$$x = N_1(\xi)x_1 + N_2(\xi)x_2 + N_3(\xi)x_3 \tag{B.41}$$

and

$$y = N_1(\xi)y_1 + N_2(\xi)y_2 + N_3(\xi)y_3, \tag{B.42}$$

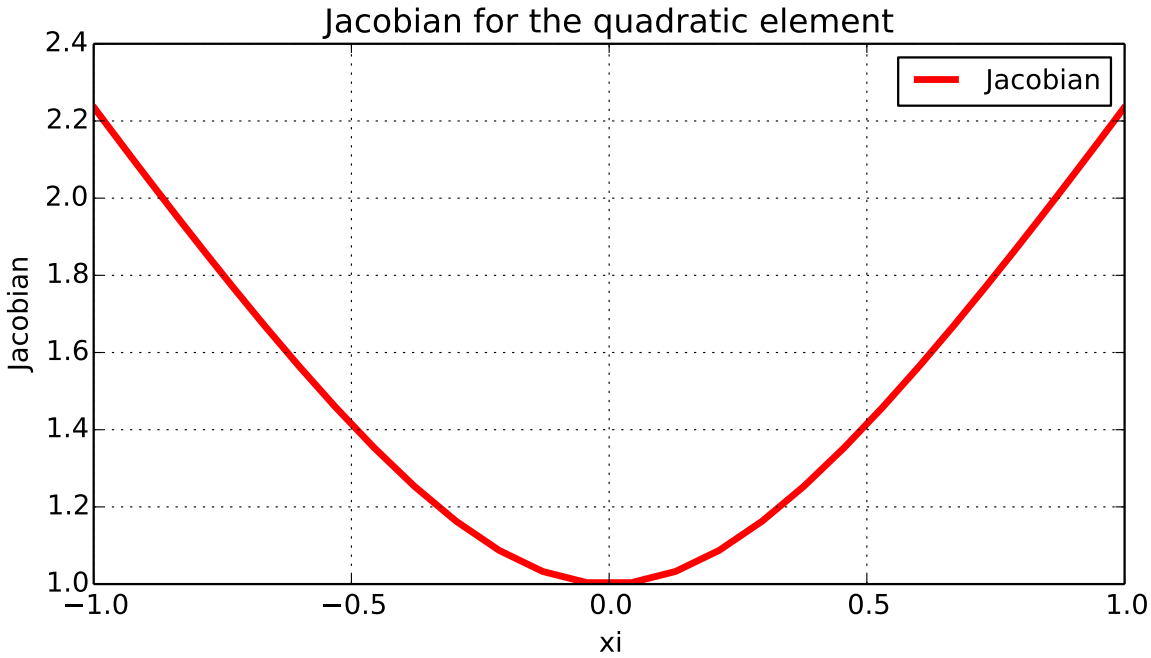

Figure B.18 – Jacobian for the quadratic element

where N_1 and N_2 and N_3 are the quadratic Bézier shape functions:

$$N_1 = (1 - \xi)^2, \tag{B.43}$$

$$N_2 = 2\xi(1 - \xi), \tag{B.44}$$

and

$$N_3 = \xi^2. \tag{B.45}$$

A visualization of the quadratic Bézier element is shown in Figure A.19. Note that the curve is contained in the polygon formed by the control points. It's interesting to compare Figures A.11 and A.19, given that they were constructed using the same geometrical points.

Another feature of the quadratic Bézier element is that the first and last geometrical points corresponds to the first and last nodes, but the intermediary node is not coincident with the intermediary geometrical point.

The behaviour of the shape functions for the quadratic Bézier element are shown in Figure A.20.

To obtain the Jacobian of the transformation from the domain $[x, y]$ to ξ, it is necessary to derivate the shape functions with respect to ξ:

$$\frac{\partial N_1}{\partial \xi} = 2(1 - \xi), \tag{B.46}$$

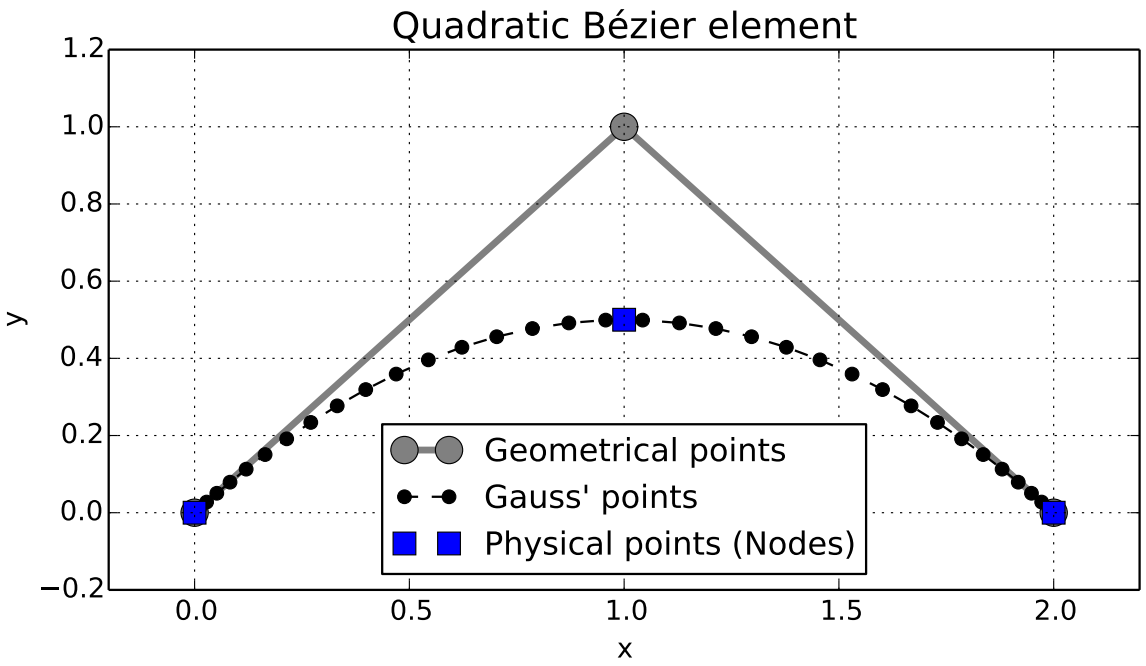

Figure B.19 – 2D quadratic Bézier element for 36 Gauss' points

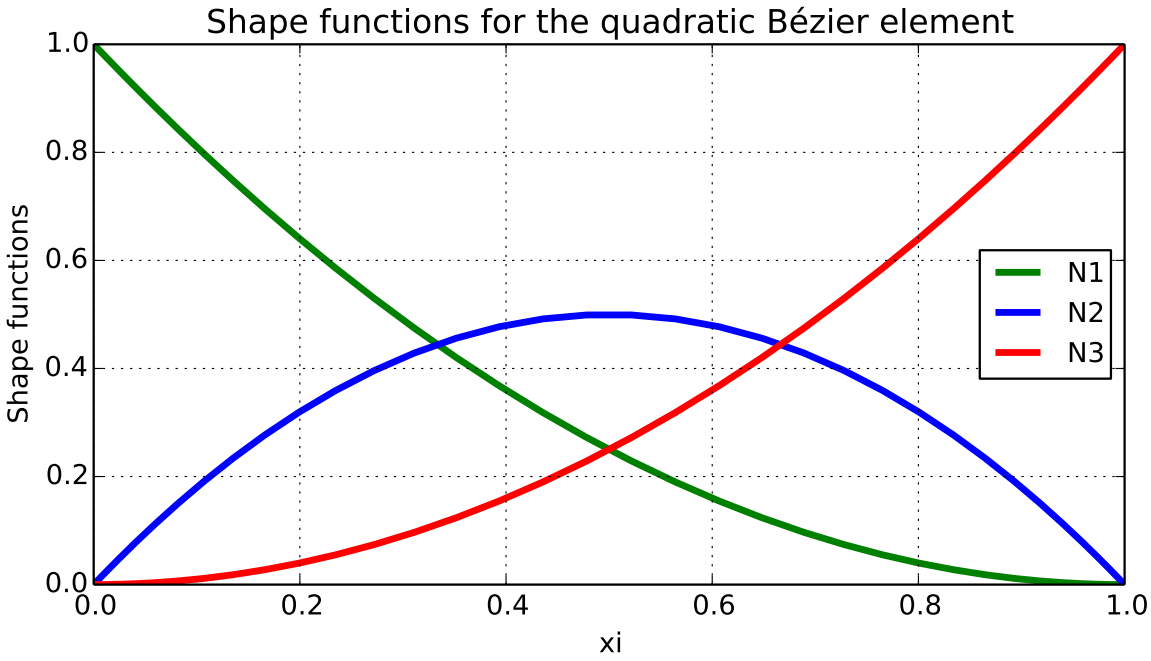

Figure B.20 – Shape functions for the quadratic Bézier element

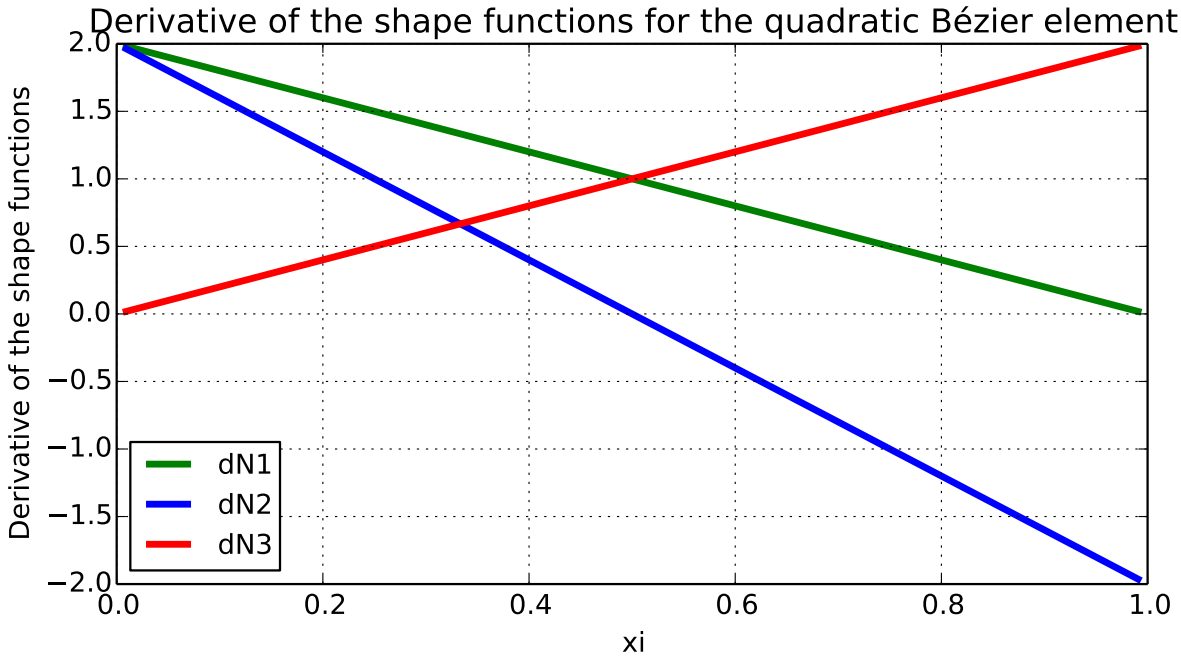

Figure B.21 – Derivative of the shape functions for the quadratic Bézier element

$$\frac{\partial N_2}{\partial \xi} = 2(1 - 2\xi), \tag{B.47}$$

and

$$\frac{\partial N_3}{\partial \xi} = 2\xi. \tag{B.48}$$

The derivatives of the shape functions for the quadratic Bézier element are shown in Figure A.21.

Once this derivatives are obtained, one obtains the derivative of $[x, y]$ in relation to ξ,

$$\frac{\partial x}{\partial \xi} = \frac{\partial N_1(\xi)}{\partial \xi} x_1 + \frac{\partial N_2(\xi)}{\partial \xi} x_2 + \frac{\partial N_3(\xi)}{\partial \xi} x_3, \tag{B.49}$$

$$\frac{\partial y}{\partial \xi} = \frac{\partial N_1(\xi)}{\partial \xi} y_1 + \frac{\partial N_2(\xi)}{\partial \xi} y_2 + \frac{\partial N_3(\xi)}{\partial \xi} y_3, \tag{B.50}$$

the Jacobian is then

$$\frac{\partial \Gamma}{\partial \xi} = \sqrt{\left(\frac{\partial x}{\partial \xi}\right)^2 + \left(\frac{\partial y}{\partial \xi}\right)^2}. \tag{B.51}$$

The integration can now be performed in the domain of ξ using traditional numerical integration techniques such as Gaussian quadrature in equations (A.52) and (A.53).

$$I_1 = \int_{-1}^{1} \frac{\partial \phi}{\partial \mathbf{n}} \phi^*(x', \xi, k) \frac{\partial \Gamma}{\partial \xi} d\xi \tag{B.52}$$

and

$$I_2 = \int_{-1}^{1} \phi \frac{\partial \phi^*}{\partial \mathbf{n}}(x', \xi, k) \frac{\partial \Gamma}{\partial \xi} d\xi, \tag{B.53}$$

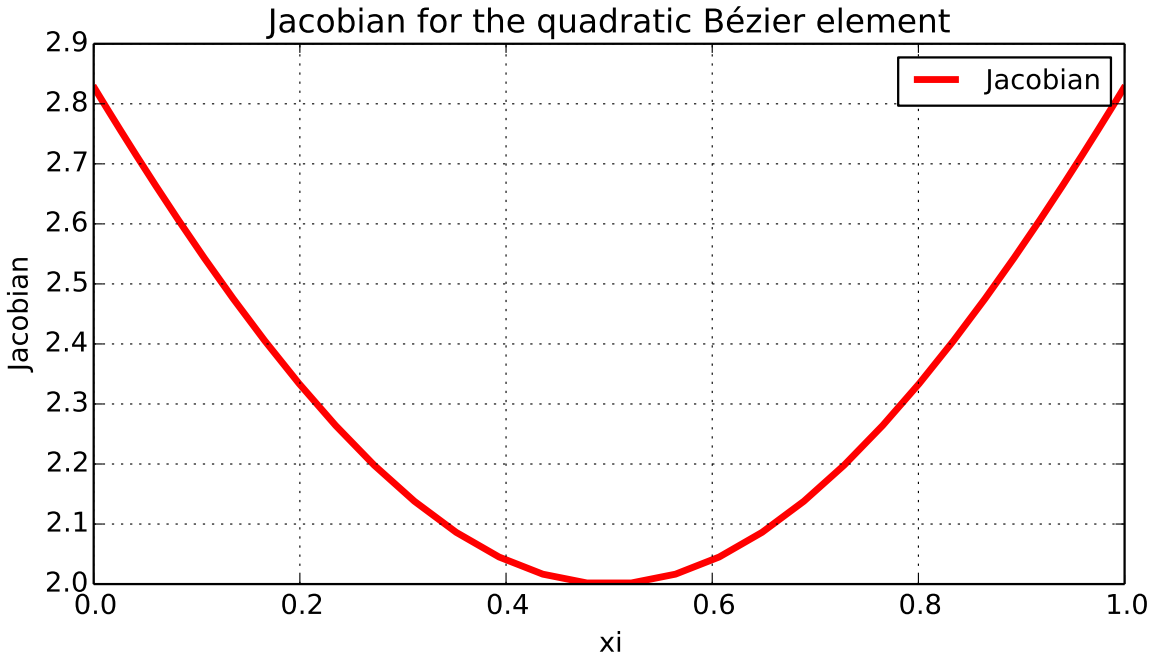

Figure B.22 – Jacobian for the quadratic Bézier element

where the Jacobian is given by equation (A.51). The Jacobian of the quadratic Bézier element is shown in Figure A.22.

B.5 Three-dimensional triangular linear elements

These elements are described by the points which form a triangular shape. The points are the vertices of the element and the position within the elemnt is described using two parametric variables $\xi, \eta \in [0, 1]$ into the shape functions shown below.

$$x = N_1(\xi, \eta)x_1 + N_2(\xi, \eta)x_2 + N_3(\xi, \eta)x_3, \tag{B.54}$$

$$y = N_1(\xi, \eta)y_1 + N_2(\xi, \eta)y_2 + N_3(\xi, \eta)y_3 \tag{B.55}$$

and

$$z = N_1(\xi, \eta)z_1 + N_2(\xi, \eta)z_2 + N_3(\xi, \eta)z_3, \tag{B.56}$$

where N_1 and N_2 and N_3 are the triangular linear shape functions which will describe the geometry of the boundary. Suitable shape functions are

$$N_1 = \xi, \tag{B.57}$$

$$N_2 = \eta, \tag{B.58}$$

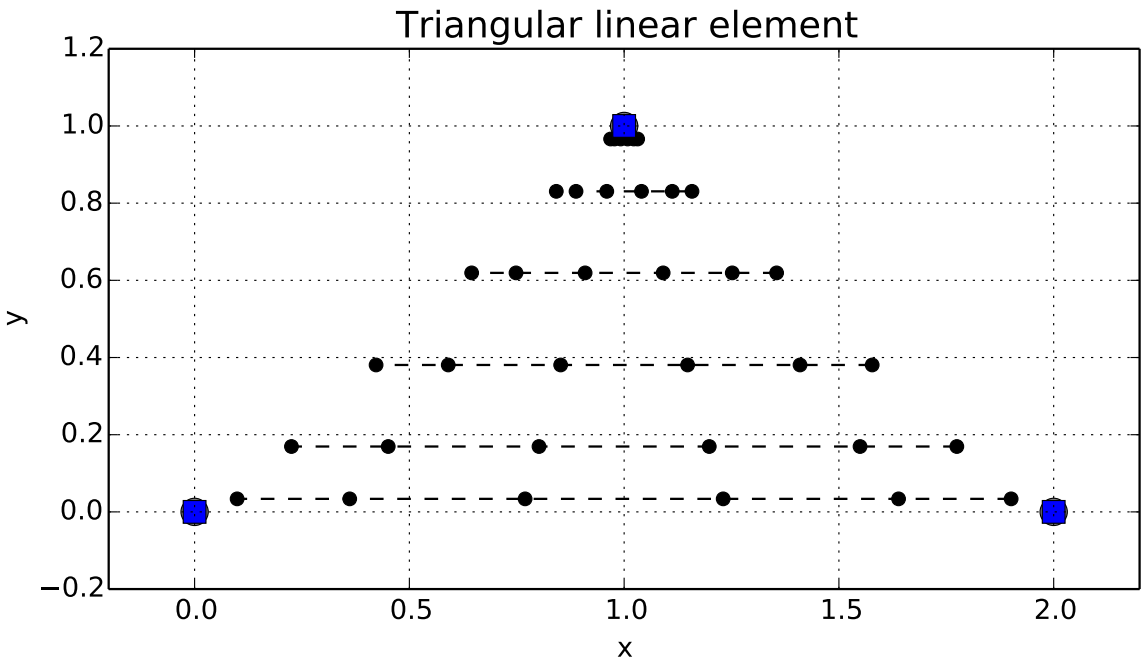

Figure B.23 – 3D triangular linear element for 6 Gauss' points for each parametric variable $[\xi, \eta]$

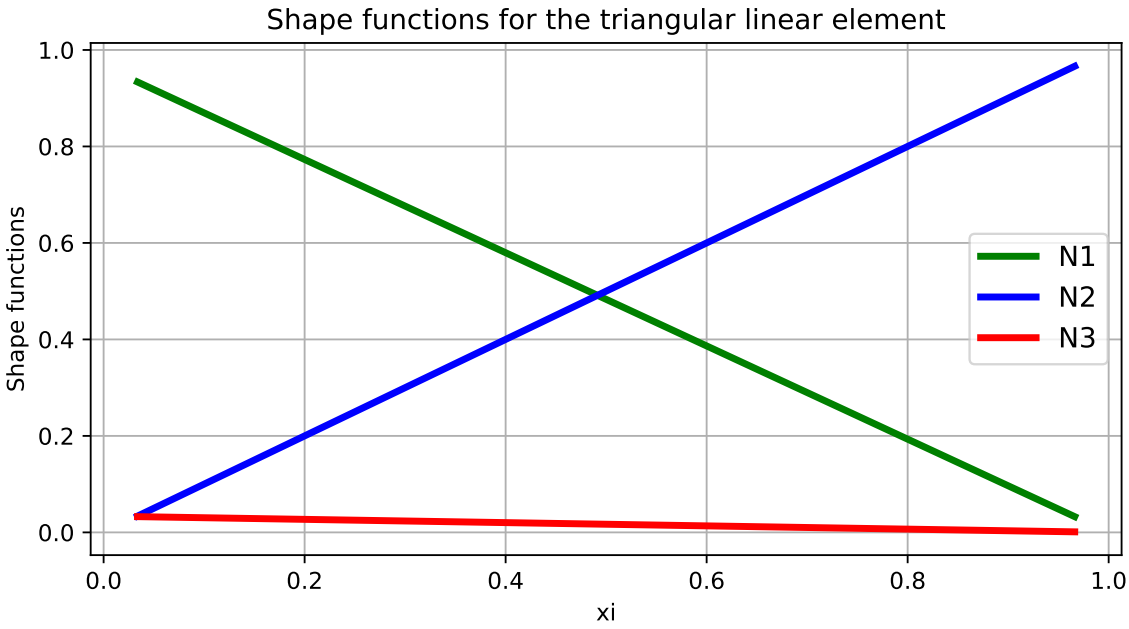

Figure B.24 – Shape functions for the triangular linear element

and

$$N_3 = 1 - \xi - \eta. \tag{B.59}$$

The element can be seen in Figure A.23, which shows the geometrical points, node and the Gauss' points for a quadrature of 6×6 points.

The shape functions are shown in Figure A.24.

The derivative of the shape functions must be obtained to formulate the Jacobian

of the transformation, which will be used to perform the integration. Thus,

$$\frac{\partial N_1}{\partial \xi} = 1, \tag{B.60}$$

$$\frac{\partial N_1}{\partial \eta} = 0, \tag{B.61}$$

$$\frac{\partial N_2}{\partial \xi} = 0, \tag{B.62}$$

$$\frac{\partial N_2}{\partial \eta} = 1, \tag{B.63}$$

$$\frac{\partial N_3}{\partial \xi} = -1 \tag{B.64}$$

and

$$\frac{\partial N_3}{\partial \eta} = -1. \tag{B.65}$$

To obtain the Jacobian, first one obtains the derivative of the position $[x, y, z]$, using the derivative of the shape functions:

$$\frac{\partial x}{\partial \xi} = \frac{\partial N_1}{\partial \xi} x_1 + \frac{\partial N_2}{\partial \xi} x_2 + \frac{\partial N_3}{\partial \xi} x_3, \tag{B.66}$$

$$\frac{\partial x}{\partial \eta} = \frac{\partial N_1}{\partial \eta} x_1 + \frac{\partial N_2}{\partial \eta} x_2 + \frac{\partial N_3}{\partial \eta} x_3 \tag{B.67}$$

and similarly for y and z (not shown). Now, the Jacobian can be obtained by applying

$$J = \sqrt{\left(\frac{\partial y}{\partial \xi}\frac{\partial z}{\partial \eta} - \frac{\partial z}{\partial \xi}\frac{\partial y}{\partial \eta}\right)^2 + \left(\frac{\partial z}{\partial \xi}\frac{\partial x}{\partial \eta} - \frac{\partial x}{\partial \xi}\frac{\partial z}{\partial \eta}\right)^2 + \left(\frac{\partial x}{\partial \xi}\frac{\partial y}{\partial \eta} - \frac{\partial y}{\partial \xi}\frac{\partial x}{\partial \eta}\right)^2} \tag{B.68}$$

The integral equations can be derived using the shape functions as

$$I_1 = \int_{-1}^{1}\int_{-1}^{1} \phi^*(x', x, k)\frac{\partial \phi(x)}{\partial \mathbf{n}} J(\xi, \eta) d\xi d\eta \tag{B.69}$$

and

$$I_2 = \int_{-1}^{1}\int_{-1}^{1} \phi(x)\frac{\partial \phi^*}{\partial \mathbf{n}}(x', x, k) J(\xi, \eta) d\xi d\eta. \tag{B.70}$$

B.6 Three-dimensional quadrilateral bilinear elements

The boundary is described by a patchwork of quadrilateral elements. The element is defined by four geometric points [x,y,z], at each the vertex of the quadrilateral. Two parametric variables are used to describe points inside the element, $\xi, \eta \in [-1, 1]$, and the position is given by applying the shape functions into the geometric points:

$$x = N_1(\xi, \eta)x_1 + N_2(\xi, \eta)x_2 + N_3(\xi, \eta)x_3 + N_4(\xi, \eta)x_4, \tag{B.71}$$

$$y = N_1(\xi, \eta)y_1 + N_2(\xi, \eta)y_2 + N_3(\xi, \eta)y_3 + N_4(\xi, \eta)y_4 \tag{B.72}$$

and

$$z = N_1(\xi, \eta)z_1 + N_2(\xi, \eta)z_2 + N_3(\xi, \eta)z_3 + N_4(\xi, \eta)z_4. \tag{B.73}$$

Shape functions N_1, N_2, N_3 and N_4 which describe this element are given by

$$N_1 = \frac{1}{4}(1 - \xi)(1 - \eta), \tag{B.74}$$

$$N_2 = \frac{1}{4}(1 + \xi)(1 - \eta), \tag{B.75}$$

$$N_3 = \frac{1}{4}(1 + \xi)(1 + \eta) \tag{B.76}$$

and

$$N_4 = \frac{1}{4}(1 - \xi)(1 + \eta). \tag{B.77}$$

A visualization of the element is shown in Figure A.25.

The derivative for each shape function is given below:

$$\frac{\partial N_1}{\partial \xi} = -\frac{1}{4}(1 - \eta), \tag{B.78}$$

$$\frac{\partial N_2}{\partial \xi} = \frac{1}{4}(1 - \eta), \tag{B.79}$$

$$\frac{\partial N_3}{\partial \xi} = \frac{1}{4}(1 + \eta), \tag{B.80}$$

$$\frac{\partial N_4}{\partial \xi} = -\frac{1}{4}(1 + \eta), \tag{B.81}$$

$$\frac{\partial N_1}{\partial \eta} = -\frac{1}{4}(1 - \xi), \tag{B.82}$$

$$\frac{\partial N_2}{\partial \eta} = -\frac{1}{4}(1 + \xi), \tag{B.83}$$

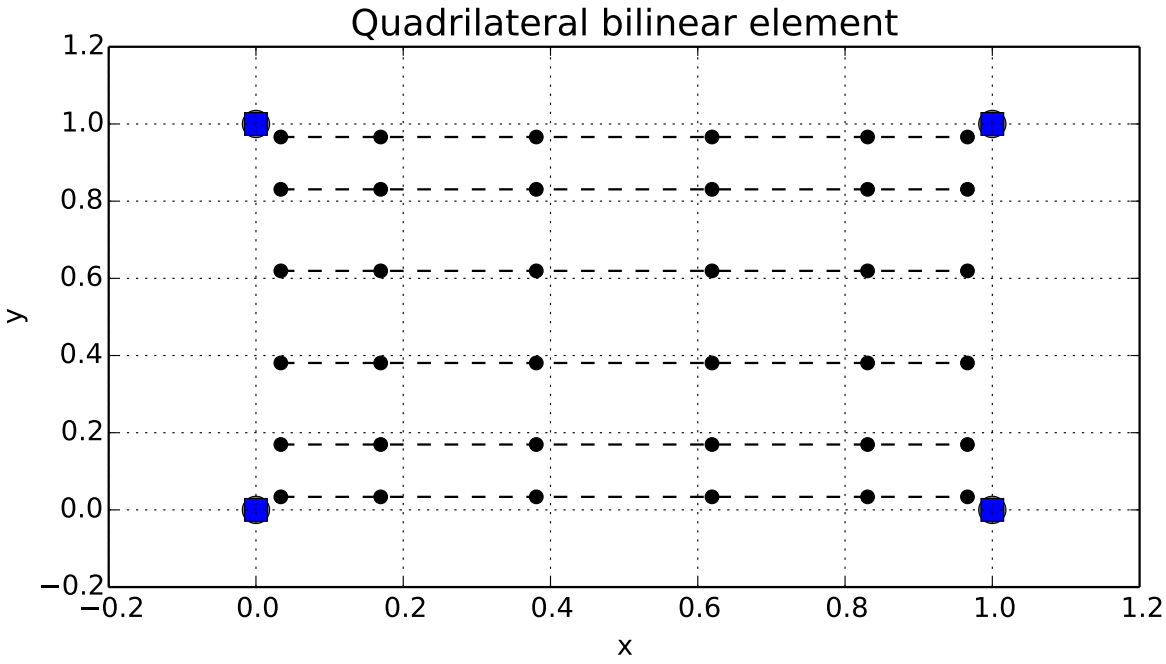

Figure B.25 – 3D quadrilateral linear element for 6 Gauss' points for each parametric variable $[\xi, \eta]$

$$\frac{\partial N_3}{\partial \eta} = \frac{1}{4}(1+\xi), \tag{B.84}$$

and

$$\frac{\partial N_4}{\partial \eta} = \frac{1}{4}(1-\xi). \tag{B.85}$$

To obtain the Jacobian, first one obtains the derivative of the position $[x, y, z]$, using the derivative of the shape functions:

$$\frac{\partial x}{\partial \xi} = \frac{\partial N_1}{\partial \xi}x_1 + \frac{\partial N_2}{\partial \xi}x_2 + \frac{\partial N_3}{\partial \xi}x_3 + \frac{\partial N_4}{\partial \xi}x_4, \tag{B.86}$$

$$\frac{\partial x}{\partial \eta} = \frac{\partial N_1}{\partial \eta}x_1 + \frac{\partial N_2}{\partial \eta}x_2 + \frac{\partial N_3}{\partial \eta}x_3 + \frac{\partial N_4}{\partial \eta}x_4 \tag{B.87}$$

and similarly for y and z (not shown). Now, the Jacobian can be obtained by applying

$$J = \sqrt{\left(\frac{\partial y}{\partial \xi}\frac{\partial z}{\partial \eta} - \frac{\partial z}{\partial \xi}\frac{\partial y}{\partial \eta}\right)^2 + \left(\frac{\partial z}{\partial \xi}\frac{\partial x}{\partial \eta} - \frac{\partial x}{\partial \xi}\frac{\partial z}{\partial \eta}\right)^2 + \left(\frac{\partial x}{\partial \xi}\frac{\partial y}{\partial \eta} - \frac{\partial y}{\partial \xi}\frac{\partial x}{\partial \eta}\right)^2} \tag{B.88}$$

The integral equations can be derived using the shape functions as

$$I_1 = \int_{-1}^{1}\int_{-1}^{1} \phi^*(x', x, k)\frac{\partial \phi(x)}{\partial \mathbf{n}} J(\xi, \eta) d\xi d\eta \tag{B.89}$$

and

$$I_2 = \int_{-1}^{1}\int_{-1}^{1} \phi(x)\frac{\partial \phi^*}{\partial \mathbf{n}}(x', x, k) J(\xi, \eta) d\xi d\eta. \tag{B.90}$$

B.7 Three-dimensional triangular quadratic elements

This element is described by 6 geometric points. The 3 first points are at the vertex of the triangular shape, and the 3 others are in located between the first. The shape functions describe a parabola, using a quadratic function to interpolate the shape. The shape functions are parametrized by the variables $\xi, \eta \in [0, 1]$, but the dummy variable $\mu = 1 - \xi - \eta$ is used to simplify the written form of the equations.

The position of any point [x,y,z] on the surface of the element is given by

$$x = \sum_{i=1}^{6} N_i(\xi, \eta) x_i, \tag{B.91}$$

$$y = \sum_{i=1}^{6} N_i(\xi, \eta) y_i, \tag{B.92}$$

and

$$z = \sum_{i=1}^{6} N_i(\xi, \eta) z_i, \tag{B.93}$$

where N_i are the quadratic shape functions which will describe the geometry of the boundary. The shape functions used are

$$N_1 = \xi(2\xi - 1), \tag{B.94}$$

$$N_2 = \eta(2\eta - 1), \tag{B.95}$$

$$N_3 = \mu(2\mu - 1), \tag{B.96}$$

$$N_4 = 4\xi\eta, \tag{B.97}$$

$$N_5 = 4\eta\mu \tag{B.98}$$

and

$$N_6 = 4\xi\mu. \tag{B.99}$$

A visualization of the element is shown in Figure A.26.

The derivatives of the shape functions are given by

$$\frac{\partial N_1}{\partial \xi} = 4\xi - 1, \tag{B.100}$$

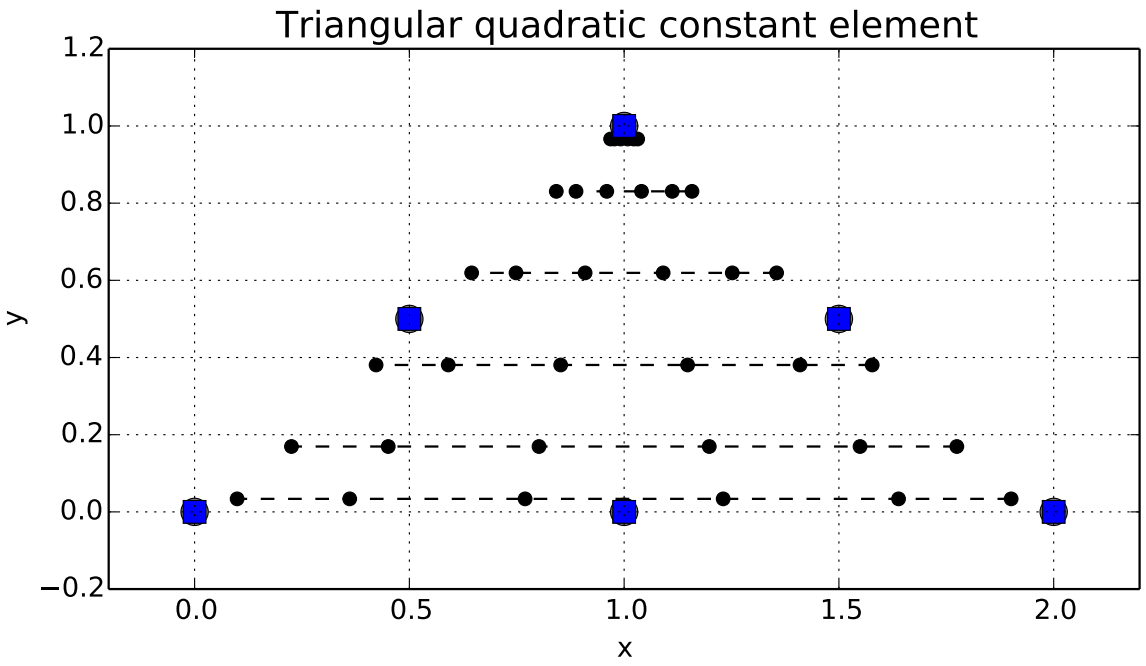

Figure B.26 – 3D triangular quadratic element for 6 Gauss' points for each parametric variable $[\xi, \eta]$

$$\frac{\partial N_1}{\partial \eta} = 0, \tag{B.101}$$

$$\frac{\partial N_2}{\partial \xi} = 0, \tag{B.102}$$

$$\frac{\partial N_2}{\partial \eta} = 4\eta - 1, \tag{B.103}$$

$$\frac{\partial N_3}{\partial \xi} = 1 - 4\mu, \tag{B.104}$$

$$\frac{\partial N_3}{\partial \eta} = 1 - 4\mu, \tag{B.105}$$

$$\frac{\partial N_4}{\partial \xi} = 4\eta, \tag{B.106}$$

$$\frac{\partial N_4}{\partial \eta} = 4\xi, \tag{B.107}$$

$$\frac{\partial N_5}{\partial \xi} = -4\eta, \tag{B.108}$$

$$\frac{\partial N_5}{\partial \eta} = 4(\mu - \eta), \tag{B.109}$$

$$\frac{\partial N_6}{\partial \xi} = 4(\mu - \xi), \tag{B.110}$$

and

$$\frac{\partial N_6}{\partial \eta} = -4\xi. \tag{B.111}$$

To obtain the Jacobian, first one obtains the derivative of the position $[x, y, z]$, using the derivative of the shape functions:

$$\frac{\partial x}{\partial \xi} = \frac{\partial N_1}{\partial \xi}x_1 + \frac{\partial N_2}{\partial \xi}x_2 + \frac{\partial N_3}{\partial \xi}x_3 + \frac{\partial N_4}{\partial \xi}x_4 + \frac{\partial N_5}{\partial \xi}x_5 + \frac{\partial N_6}{\partial \xi}x_6, \tag{B.112}$$

$$\frac{\partial x}{\partial \eta} = \frac{\partial N_1}{\partial \eta}x_1 + \frac{\partial N_2}{\partial \eta}x_2 + \frac{\partial N_3}{\partial \eta}x_3 + \frac{\partial N_4}{\partial \eta}x_4 + \frac{\partial N_5}{\partial \eta}x_5 + \frac{\partial N_6}{\partial \eta}x_6 \tag{B.113}$$

and similarly for y and z (not shown). Now, the Jacobian can be obtained by applying

$$J = \sqrt{\left(\frac{\partial y}{\partial \xi}\frac{\partial z}{\partial \eta} - \frac{\partial z}{\partial \xi}\frac{\partial y}{\partial \eta}\right)^2 + \left(\frac{\partial z}{\partial \xi}\frac{\partial x}{\partial \eta} - \frac{\partial x}{\partial \xi}\frac{\partial z}{\partial \eta}\right)^2 + \left(\frac{\partial x}{\partial \xi}\frac{\partial y}{\partial \eta} - \frac{\partial y}{\partial \xi}\frac{\partial x}{\partial \eta}\right)^2} \tag{B.114}$$

The integral equations can be derived using the shape functions as

$$I_1 = \int_{-1}^{1}\int_{-1}^{1} \phi^*(x', x, k)\frac{\partial \phi(x)}{\partial \mathbf{n}} J(\xi, \eta) d\xi d\eta \tag{B.115}$$

and

$$I_2 = \int_{-1}^{1}\int_{-1}^{1} \phi(x)\frac{\partial \phi^*}{\partial \mathbf{n}}(x', x, k) J(\xi, \eta) d\xi d\eta. \tag{B.116}$$

B.8 Three-dimensional quadrilateral quadratic elements

The boundary is described by a Lagrangian quadrilateral quadratic element. The element consists of 8 geometric points [x,y,z] which describes 4 quadratic curves. Two parametric variables $\xi, \eta \in [-1, 1]$ are used to describe the geometry inside the element. The 4 first points are the outer most corners of the element and the others are between those, as shown in Figure A.27.

This will be the first approximation of the half cylinder studied. It's important to note that quadratic polynomials can never describe conic sections such as circles, and this geometrical approximation is visible in Figure A.27.

The position of any point [x,y,z] on the surface of the element is given by

$$x = \sum_{i=1}^{8} N_i(\xi, \eta)x_i, \tag{B.117}$$

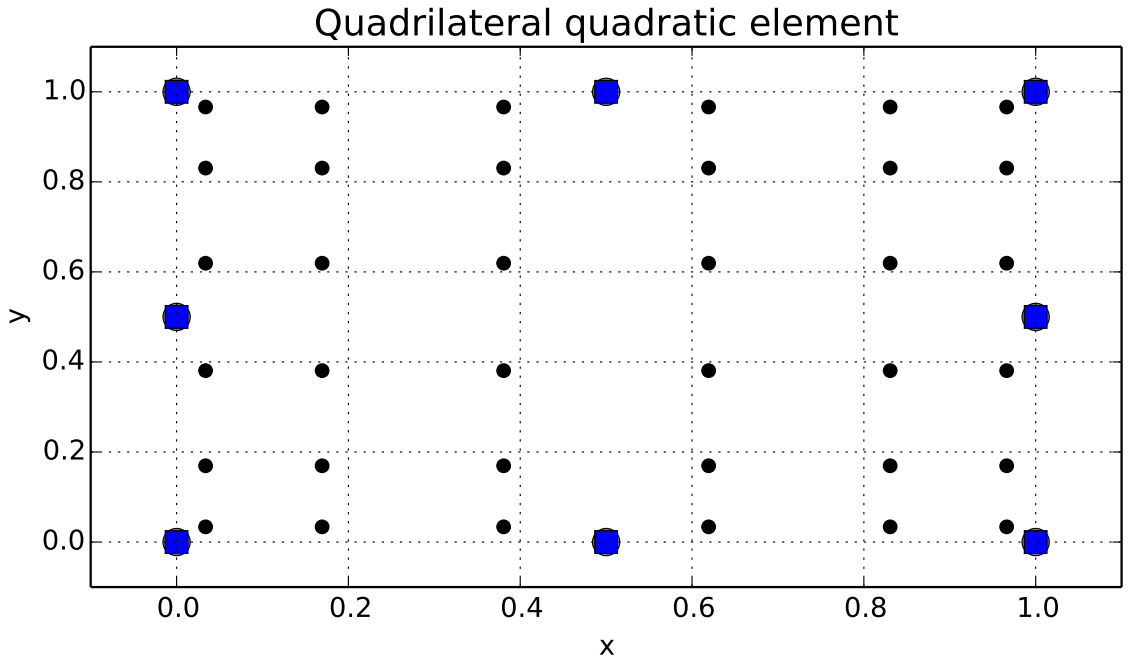

Figure B.27 – 3D quadrilateral quadratic element for 6 Gauss' points for each parametric variable $[\xi, \eta]$

$$y = \sum_{i=1}^{8} N_i(\xi, \eta) y_i, \tag{B.118}$$

and

$$z = \sum_{i=1}^{8} N_i(\xi, \eta) z_i, \tag{B.119}$$

where N_i are the quadratic shape functions which will describe the geometry of the boundary. The shape functions used are

$$N_1 = -\frac{1}{4}(1-\xi)(1-\eta)(\xi+1+\eta), \tag{B.120}$$

$$N_2 = \frac{1}{4}(1+\xi)(1-\eta)(\xi-1-\eta), \tag{B.121}$$

$$N_3 = \frac{1}{4}(1+\xi)(1+\eta)(\eta-1+\xi), \tag{B.122}$$

$$N_4 = \frac{1}{4}(1-\xi)(1+\eta)(\eta-1-\xi), \tag{B.123}$$

$$N_5 = \frac{1}{2}(1+\xi)(1-\xi)(1-\eta), \tag{B.124}$$

$$N_6 = \frac{1}{2}(1+\xi)(1+\eta)(1-\eta), \tag{B.125}$$

$$N_7 = \frac{1}{2}(1+\eta)(1+\xi)(1-\xi), \tag{B.126}$$

and

$$N_8 = \frac{1}{2}(1-\xi)(1+\eta)(1-\eta). \tag{B.127}$$

Now the derivatives are obtained.

$$\frac{\partial N_1}{\partial \xi} = \frac{1}{4}(1-\eta)(2\xi+\eta), \tag{B.128}$$

$$\frac{\partial N_1}{\partial \eta} = \frac{1}{4}(1-\xi)(2\eta+\xi), \tag{B.129}$$

$$\frac{\partial N_2}{\partial \xi} = \frac{1}{4}(1-\eta)(2\xi-\eta), \tag{B.130}$$

$$\frac{\partial N_2}{\partial \eta} = \frac{1}{4}(1+\xi)(2\eta-\xi), \tag{B.131}$$

$$\frac{\partial N_3}{\partial \xi} = \frac{1}{4}(1+\eta)(2\xi+\eta), \tag{B.132}$$

$$\frac{\partial N_3}{\partial \eta} = \frac{1}{4}(1+\xi)(2\eta+\xi), \tag{B.133}$$

$$\frac{\partial N_4}{\partial \xi} = \frac{1}{4}(1+\eta)(2\xi-\eta), \tag{B.134}$$

$$\frac{\partial N_4}{\partial \eta} = \frac{1}{4}(1-\xi)(2\eta-\xi), \tag{B.135}$$

$$\frac{\partial N_5}{\partial \xi} = -\xi(1-\eta), \tag{B.136}$$

$$\frac{\partial N_5}{\partial \eta} = -\frac{1}{2}(1+\xi)(1-\xi), \tag{B.137}$$

$$\frac{\partial N_6}{\partial \xi} = \frac{1}{2}(1+\eta)(1-\eta), \tag{B.138}$$

$$\frac{\partial N_6}{\partial \eta} = -\eta(1+\xi), \tag{B.139}$$

$$\frac{\partial N_7}{\partial \xi} = -\xi(1+\eta), \tag{B.140}$$

$$\frac{\partial N_7}{\partial \eta} = \frac{1}{2}(1+\xi)(1-\xi), \tag{B.141}$$

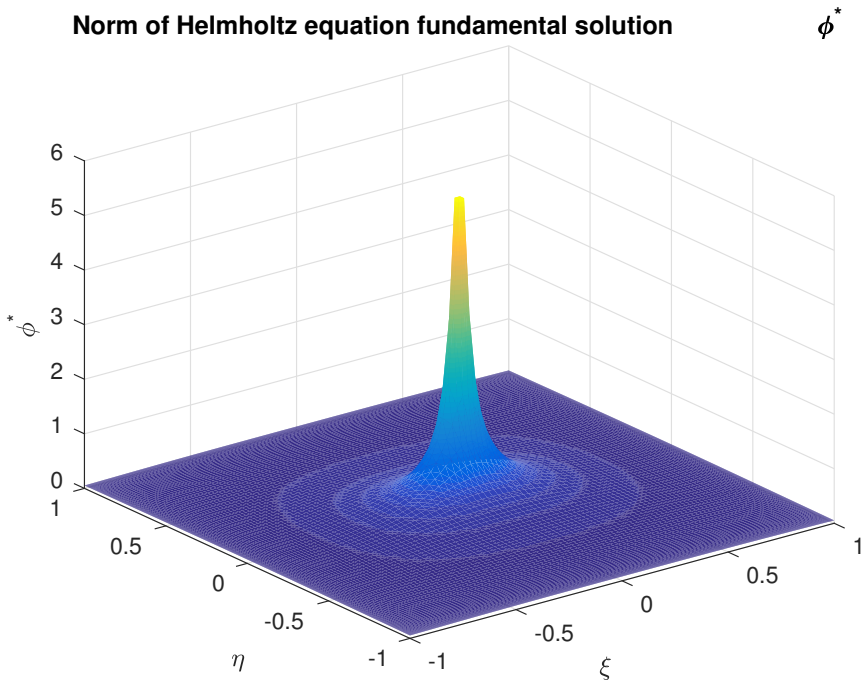

Figure B.28 – Absolute value of the fundamental solution for the Helmholtz equation in parametric space for quadrilateral quadratic elements

$$\frac{\partial N_8}{\partial \xi} = -\frac{1}{2}(1+\eta)(1-\eta) \tag{B.142}$$

and

$$\frac{\partial N_8}{\partial \eta} = -\eta(1-\xi). \tag{B.143}$$

The Jacobian can be obtained by

$$J = \sqrt{\left(\frac{\partial y}{\partial \xi}\frac{\partial z}{\partial \eta} - \frac{\partial z}{\partial \xi}\frac{\partial y}{\partial \eta}\right)^2 + \left(\frac{\partial z}{\partial \xi}\frac{\partial x}{\partial \eta} - \frac{\partial x}{\partial \xi}\frac{\partial z}{\partial \eta}\right)^2 + \left(\frac{\partial x}{\partial \xi}\frac{\partial y}{\partial \eta} - \frac{\partial y}{\partial \xi}\frac{\partial x}{\partial \eta}\right)^2} \tag{B.144}$$

Now the integral equations can be derived using the shape functions as

$$I_1 = \int_{-1}^{1}\int_{-1}^{1} \phi^*(x', x, k)\frac{\partial \phi(x)}{\partial \mathbf{n}} J(\xi, \eta) d\xi d\eta \tag{B.145}$$

and

$$I_2 = \int_{-1}^{1}\int_{-1}^{1} \phi(x)\frac{\partial \phi^*}{\partial \mathbf{n}}(x', x, k) J(\xi, \eta) d\xi d\eta, \tag{B.146}$$

where J is the Jacobian of the coordinate transformation into parametric space defined by $[\xi, \eta] \in [-1, 1]$ given by equation A.144.

For the Helmholtz equation, the value of the fundamental solution and its normal derivative in the parametric space is shown in Figure A.28 and A.29.

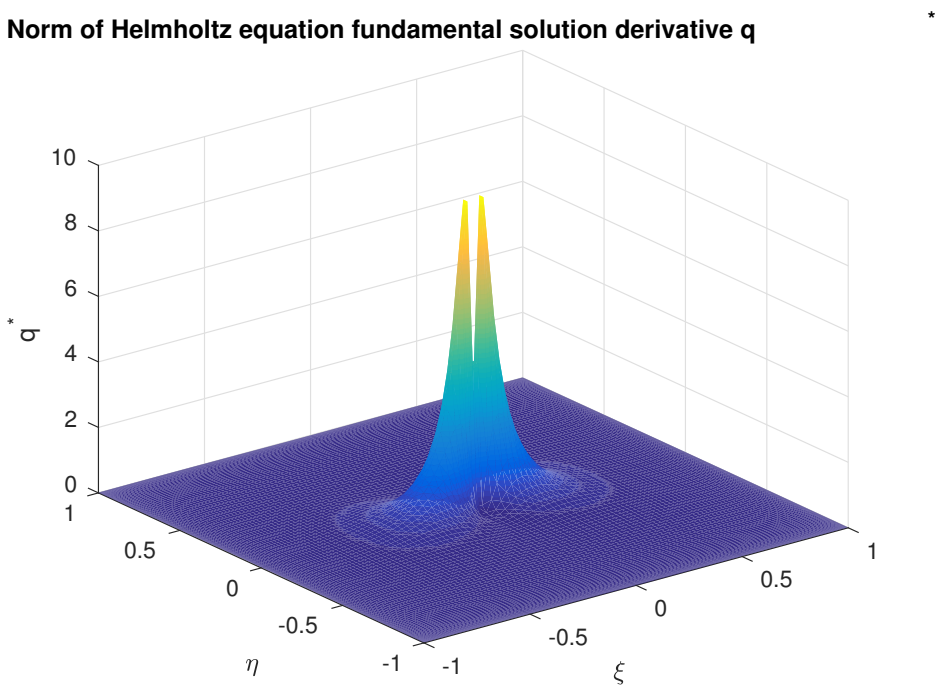

Figure B.29 – Absolute value of the normal derivative of the fundamental solution for the Helmholtz equation in parametric space for quadrilateral quadratic elements