2022-05-31

An Iterative Numerical Method for Multiple Scattering Using High Order Local Absorbing Boundary Conditions

Jonathan Harriman Hale
Brigham Young University

An Iterative Numerical Method for Multiple Scattering Using High Order Local Absorbing Boundary Conditions

Jonathan Harriman Hale

A thesis submitted to the faculty of
Brigham Young University
in partial fulfillment of the requirements for the degree of

Master of Science

Vianey Villamizar, Chair
Jared Whitehead
Blake Barker

Department of Mathematics

Brigham Young University

ABSTRACT

An Iterative Numerical Method for Multiple Scattering Using High Order Local Absorbing Boundary Conditions

Jonathan Harriman Hale
Department of Mathematics, BYU
Master of Science

This thesis outlines an iterative approach for determining the scattered wave for two dimensional multiple acoustic scattering problems using high order local absorbing bound-ary conditions and second order finite difference. We seek to approximate the total wave as it is scattered off of multiple arbitrarily shaped obstacles. This is done by decomposing the scattered wave into the superposition of single scattered waves. We then repeatedly solve the single scattering system for each obstacle, while updating the boundary conditions based off the incident wave and the scattered wave off the other obstacles. We solve each single scattering by enclosing the obstacle in a circular artificial boundary and generating a curvilinear coordinate system for the computational region between the obstacle and the artificial boundary. We impose an absorbing boundary condition, specifically Karp's Farfield Expansion ABC, on the artificial boundary. We use a finite difference method to discretize the governing equations and to discretize the absorbing boundary conditions. This will cre-ate a linear system whose solution will approximate the single scattered wave. The forcing vector of the linear system is determined from the total influence on the obstacle boundary from the incident wave and the scattered waves from the other obstacles. In each iteration, we solve the singular acoustic scattering problem for each obstacle by using the scattered wave approximations from the other obstacles obtained from the previous iteration. The iterations continue until the solutions converge.

This iterative method scales well to multiple scattering configurations with many ob-stacles, and achieves errors on the order of 10^{-5} in less than five minutes. This is due to using LU factorization to solve the linear systems, paired with parallelization. I will include numerical results which demonstrate the accuracy and advantages of this iterative technique.

Keywords: multiple scattering, iterative multiple scattering, finite difference method, Helmholtz equation, high order absorbing boundary conditions, curvilinear coordinates, acoustic scat-tering, farfield pattern

Acknowledgements

I would like to acknowledge my mentor, Dr. Vianey Villamizar, for seeing my potential as an undergraduate and recruiting me during my Sophomore year. Dr. Villamizar has a positive attitude towards research that continually kept me going when our research results weren't what we expected. He has taught me the principle of enduring and having faith in our future success. He gave me encouragement when times were tough, and celebrated with me during our breakthroughs. What is even more amazing is that Dr. Villamizar did all of this remotely, due to the Covid-19 pandemic.

I also would like to acknowledge the other members of my graduate committee, Dr. Blake Barker and Dr. Jared Whitehead. These two professors have taught multiple math classes that I attended throughout my university experience. When I think of a math education at Brigham Young University, I think of these two professors. It has been a great honor having these two on my graduate committee.

I would like to thank my wife, Kaelena, for always believing in me and giving me the support I needed from home. Being a masters student was much busier than either of us expected, but Kaelena always encouraged me to go for it. I know there were many television episodes missed because of late nights working on research or homework, but she never complained.

I would like to acknowledge my parents for their emotional and financial support throughout my undergraduate and graduate degree. They helped me with tuition for my first two years of college before I was given scholarships my Junior year, and have subsidized my housing during all of my undergrad. Even after their explicit financial support ended, they still fed me and Kaelena frequent meals which was a greater help than they probably know. My father and mother have always believed and emphasized the importance of getting an education. This thesis is the current amalgamation of those beliefs that they have instilled in me.

I also would like to thank my parents-in-law. They have subsidized my housing during my

graduate degree, and also allowed Kaelena and I to live in their guest room very frequently.

I would like to acknowledge Brigham Young University for providing such stellar educational and research opportunities to people like me. Brigham Young University has also financially supported me my entire university experience, both in paying my tuition and employing me in many grading/teaching opportunities.

I also would like to acknowledge Gene and Gwen Wickes, who supplied me with a departmental mathematics scholarship my Junior year. Your generosity is greatly appreciated and your impact is felt.

Contents

List of Tables

List of Figures

Chapter 1. Introduction

The multiple scattering of waves is an important physical problem and has relevant applications in acoustic, elastic and electromagnetic waves, among others. In this work, we focus in the two-dimensional Helmholtz equation, which models the time-harmonic solution to the wave equation. We suppose that an acoustic sound plane wave, denoted by u_{inc}, bounces off of multiple objects and creates a scattered wave u. For u_{inc} with wavenumber k coming in at angle ϕ measured from the positive x-axis, we have

$$u_{inc}(x, y) = e^{ik(x\cos(\phi)+y\sin(\phi))}. \tag{1.1}$$

Given M obstacles, we denote the boundary of obstacle m by Γ_m, and the unbounded region in the exterior of Γ_m by Ω_m. The boundary condition on Γ_m can either be Dirichlet, Neumann or Robin, which correspond to acoustic softness, hardness, or intermediate hardness, respectively. Let $\Gamma = \bigcup_{m=1}^{M} \Gamma_m$ and $\Omega = \bigcap_{m=1}^{M} \Omega_m$. The boundary value problem (BVP) we are solving can be strongly formulated as

$$\triangle u + k^2 u = 0, \qquad \text{in } \Omega \tag{1.2}$$

$$Z\frac{\partial u}{\partial \nu} + (1-Z)u = -\left(Z\frac{\partial u_{inc}}{\partial \nu} + (1-Z)u_{inc}\right), \quad \text{in } \Gamma \tag{1.3}$$

$$\lim_{r\to\infty} r^{1/2}(\partial_r u - iku) = 0 \tag{1.4}$$

where we denote the acoustic hardness of an obstacle by $Z \in [0, 1]$, with $Z = 0$ being Dirichlet, $Z = 1$ being Neumann, and $0 < Z < 1$ being a Robin condition. Also $k > 0$ is the wavenumber, the $\triangle$ symbol represents the Laplacian operator and $\frac{\partial}{\partial \nu}$ is the normal derivative operator relative to the boundary Γ. Condition (1.4) is known as the Sommerfeld radiation condition, and the limit is assumed to hold uniformly for all directions $\mathbf{x}/|\mathbf{x}|$ with $r = |\mathbf{x}|$, for all $\mathbf{x} \in \Omega$. Essentially, our solution u must satisfy:

(i) The Helmoltz equation (1.2) in Ω .

(ii) The Dirichlet, Neumann or Robin boundary condition (1.3) on Γ .

(iii) The Sommerfeld radiation condition (1.4) when $r \to \infty$.

Under reasonable boundary conditions, this BVP has been shown to have a unique solution [1, 2]. There are also available analytical solutions for the scattering from several circular cylindrical obstacles based on eigenfunction expansion techniques combined with appropriate addition theorems, which are also called multipole methods [2]. In particular, if we have M circular cylinders of radius R^m centered at (b_x^m, b_y^m), the multipole method leads to a representation of the exact solution given by

$$u(r,\theta) = \sum_{m=1}^{M} \sum_{j=-\infty}^{\infty} c_j^m H_j(kr^m) e^{i(j\theta^m)}, \tag{1.5}$$

where the pair (r^m, θ^m) represents a local polar coordinate system for each circular cylinder centered at (b_x^m, b_y^m). The coefficients c_j^m are the solution to an infinite system of linear equations [2]. For practical purposes, the infinite summation in (1.5) is truncated by limiting j to go from $j = -N$ to $j = N$. Therefore, the representation (1.5) is no longer exact. However, the value of N can be increased to achieve greater accuracy in the approximation of the truncated series to the exact solution. We have found that $N = 125-150$ works very well for our purposes, and we will often refer to this approximation as the "true" solution, even though the true solution is reached in the limit as $N \to \infty$.

In [2], Martin also gives an analytical solution for what is called the "farfield pattern", or the scattered wave pattern. This is given by the angular function present in the dominant term in the asymptotic expansion of the scattered wave (1.5) when $r \to \infty$. It can be shown that

$$\lim_{r\to\infty} u(r,\theta) = \frac{e^{ikr}}{(kr)^{1/2}} u_\infty(\theta) + O(1/(kr)^{3/2}),$$

where

$$u_\infty(\theta) = \frac{1-i}{\sqrt{\pi}} \sum_{j=-N}^{N} (-i)^j e^{ij\theta} \left[\sum_{m=1}^{M} c_j^m e^{-ik(b_x^m \cos(\theta) + b_y^m \sin(\theta))} \right]. \tag{1.6}$$

The farfield pattern is an important physical property of the scatterers. It also allows to measure the accuracy of any numerical method for wave scattering. In Chapter 4, we will compare our numerical results to both the analytical scattered wave and the analytical farfield pattern.

Chapter 2. Formulation of the Iterative Multiple Scattering Problem Combined with Karp's Farfield Expansion ABC

The BVP defined by (1.2)-(1.4) has an unbounded domain. This is of common occurrence in many real problems in acoustics, electromagnetics, and elasticity. Numerical methods to solve problems in unbounded domains have been the subject of intensive work since the 1970s [3]. The general approach is to truncate the infinite domain Ω with an artificial boundary $\mathcal{C}$, separating it into the two regions Ω^- and Ω^+, where Ω^- is the bounded domain enclosed by the obstacle boundary Γ and the artificial boundary $\mathcal{C}$, while Ω^+ is the remaining unbounded region in the exterior of the artificial boundary. All computation is then done in Ω^-, and special boundary conditions are imposed on the artificial boundary $\mathcal{C}$ to minimize reflections or to absorb the outgoing waves. These boundary conditions are rightfully called Non Reflecting Boundary Conditions (NRBC) or Absorbing Boundary Conditions (ABC) [4].

2.1 Brief history of absorbing boundary conditions (ABC)

Starting in the 1970s, the first attempt at imposing a special boundary condition on the artificial boundary was to impose a condition much like the Sommerfeld radiation condition (1.4). Essentially, $\partial_r u - iku = 0$ was imposed on the artificial boundary $\mathcal{C}$, which looks much like (1.4) with the limit and $r^{1/2}$ removed [3]. This boundary condition is not very accurate,

and leads to poor approximations of the scattered field at the artificial boundary.

In the mid 1980s, the next attempt was to create linear operators that would annihilate terms in an asymptotic expansion of the outgoing waves due to Karp [5], $u \approx \sqrt{\frac{2}{\pi kr}} e^{i(kr-\pi/2)} \sum_{j=0}^{\infty} \frac{f_j(\theta)}{r^j}$, where f_j is an asymptotic family of farfield functions. We will discuss more in depth this representation in the next Sections. Bayliss and Turkel did this successfully, and created a family of operators $\mathbf{B_m}$ that, when applied to the Karp's asymptotic expansion at the artificial boundary, would annihilate the first m terms of it [6]. These conditions became very popular, but they are very difficult to implement in practice for $m \geq 3$, since the linear operators require higher derivatives of u in their definition. But, even just annihilating the first and second terms of the asymptotic expansion gives fair results, and this method is still used today when high accuracy and computational efficiency is not a priority.

In the late 1980s to mid 1990s, some exact absorbing boundary conditions were developed. The most prominent of the exact absorbing boundary conditions is based on the Dirichlet-to-Neumann map [7]. This condition can absorb the scattered wave to an arbitrary order and can be easily implemented. However, this boundary condition is based off of integrating over the absorbing boundary, so its finite difference approximation involves all points along the artificial boundary, resulting in dense sub-matrices [4]. Boundary conditions that require the computation of all points along the boundary for each boundary point are called non-local conditions.

The absorbing boundary condition we will be using is called Karp's farfield expansion ABC [8], and it is an exact local condition. It is exact because it represents the outgoing wave exactly in the exterior of circular artificial boundaries, and it is local in that the approximation of this boundary condition for a point on the boundary relies only on it's immediate neighboring points. This leads to sparse matrices, while still being an exact absorbing boundary condition. This ABC will be spoken of more in Section (2.3.1).

2.2 Reformulation of the multiple scattering BVP as single scattering BVPs for each scatterer

Decomposition of the scattered wave is crucial to the iterative method presented in this thesis. The BVP (1.2) - (1.4) has an alternate formulation proposed by Guizane et al. [9]. In this formulation, the total scattered wave is decomposed to be the sum of each obstacle scattered wave, so $u = \sum_{m=1}^{M} u_m$. Each u_m can be viewed as the solution of a single scattering problem with a boundary condition on Γ_m not only dependent on the original incident wave, but the outgoing waves $u_{m'}$ $(m' \neq m)$ from all other scatterers. We also adopt a local polar coordinate system, (r^m, θ^m), for each obstacle. The new formulation is as follows:

For all $m \in \{1, ..., M\}$

$$\triangle u_m + k^2 u_m = 0, \qquad \text{in } \Omega_m \tag{2.1}$$

$$\mathcal{B}u_m = -\mathcal{B}(u_{inc} + \sum_{m' \neq m} u_{m'}), \quad \text{in } \Gamma_m \tag{2.2}$$

$$\lim_{r^m \to \infty} (r^m)^{1/2} (\partial_{r^m} u_m - iku_m) = 0, \tag{2.3}$$

where Ω^m is the exterior region to the obstacle boundary Γ^m, and the boundary operator $\mathcal{B}$ is defined as $\mathcal{B}u = Z\frac{\partial u}{\partial \nu} + (1-Z)u$. Balabane [10] showed that this M coupled single scattering problems for u_m $(m = 1 \dots M)$ has a unique solution. Once the solution of (2.1)-(2.3) is obtained, the total scattered wave u, can be recovered by

$$u = \sum_{m=1}^{M} u_m, \qquad \text{in } \Omega. \tag{2.4}$$

2.3 Reformulation of (2.1)-(2.3) using Karp's Farfield Expansion

There is one last reformulation to perform before we are ready to decouple our BVP and solve it using the novel numerical iterative method. We will reformulate our BVP by encircling an artificial boundary around each scatterer and imposing an absorbing boundary condition (ABC) on that artificial boundary, as done in [11, 12]. We will first discuss the absorbing

boundary condition used in the single scattering, and then describe its extension to the multiple scattering problem (2.1) - (2.3).

2.3.1 Single Scattering Karp's farfield expansion ABC. The single scattering problem consists of only one obstacle. The symbol Γ is no longer representing the union of obstacle boundaries, but simply the single obstacle boundary. Likewise, Ω is just the exterior region outside of the single obstacle boundary. Villamizar, Acosta, and Dastrup [8] constructed an absorbing boundary condition for single scattering that they called *Karp's farfield expansion ABC* (KFE). This is based on Karp's theorem:

Theorem 1 (Karp [5]). Let u be an outgoing solution of the two–dimensional Helmholtz equation in the exterior region to a circle $r = R$. Then, u can be represented by a convergent expansion

$$u(r,\theta) = H_0(kr)\sum_{l=0}^{\infty}\frac{F_l(\theta)}{(kr)^l} + H_1(kr)\sum_{l=0}^{\infty}\frac{G_l(\theta)}{(kr)^l}, \qquad \text{for } r > R. \tag{2.5}$$

This series is uniformly and absolutely convergent for $r > R$ and can be differentiated term by term with respect to r and θ any number of times.

The functions H_0 and H_1 are Hankel functions of the first kind of order 0 and 1, respectively. The terms F_l and G_l $(l = 1, 2, \dots)$ can be computed recursively from F_0 and G_0. To accomplish this, we substituted the expansion (2.5) into the Helmholtz equation in polar coordinates and use the identities: $H_0(z)' = -H_1(z)$ and $H_1(z)' = H_0(z) - \frac{1}{z}H_1(z)$. In fact, by doing this and requiring the coefficients of H_0 and H_1 to vanish, we derived a recurrence formula for the coefficients F_l and G_l of the expansion (2.5). This result is stated in the following corollary.

Corollary 1. The coefficients $F_l(\theta)$ and $G_l(\theta)$ $(l > 1)$ of the expansion (2.5), can be determined from $F_0(\theta)$ and $G_0(\theta)$ by the recursion formulas

$$2lG_l(\theta) = (l-1)^2 F_{l-1}(\theta) + d_\theta^2 F_{l-1}(\theta), \qquad \text{for } l = 1, 2, \dots \tag{2.6}$$

$$2lF_l(\theta) = -l^2 G_{l-1}(\theta) - d_\theta^2 G_{l-1}(\theta), \qquad \text{for } l = 1, 2, \dots. \tag{2.7}$$

Using Theorem 1 and Corollary 1, Villamizar et. al. constructed their novel absorbing boundary condition KFE, which we now state in the following BVP formulation. First, the obstacle is surrounded with an artificial boundary $\mathcal{C}$ of radius R. This separates Ω into the two regions Ω^- and Ω^+, as we have discussed before. Then, the single scattering BVP is formulated as

$$\triangle u + k^2 u = 0, \qquad \text{in } \Omega^- \tag{2.8}$$

$$\mathcal{B}u = -\mathcal{B}u_{inc} \quad \text{in } \Gamma \tag{2.9}$$

with the KFE as an absorbing bounday condition given by:

$$u(R,\theta) = \mathcal{K}_L(R,\theta)) \text{ in } \mathcal{C}, \tag{2.10}$$

$$\frac{\partial u}{\partial r}(R,\theta) = \frac{\partial \mathcal{K}_L}{\partial r}(R,\theta) \text{ in } \mathcal{C}, \tag{2.11}$$

$$H_0(kR)[(L-1)^2 F_{L-1} + d_\theta^2 F_{L-1}] + H_1(kR)[L^2 G_{L-1} + d_\theta^2 G_{L-1}] = 0, \tag{2.12}$$

$$2lG_l = (l-1)^2 F_{l-1} + d_\theta^2 F_{l-1} \text{ for } l = 1, ..., L-1, \tag{2.13}$$

$$2lF_l = -l^2 G_{l-1} - d_\theta^2 G_{l-1} \text{ for } l = 1, ..., L-1. \tag{2.14}$$

The coordinates (r,θ) are the local polar coordinates for the region Ω^+. The symbol $\mathcal{K}_L$ is used to denote the truncated version of Karp's farfield expansion using L terms,

$$\mathcal{K}_L(r,\theta) = H_0(kr)\sum_{l=0}^{L-1}\frac{F_l(\theta)}{(kr)^l} + H_1(kr)\sum_{l=0}^{L-1}\frac{G_l(\theta)}{(kr)^l}, \text{ when } r > R. \tag{2.15}$$

We describe now some important facts about the KFE:

(i) The Sommerfeld radiation condition has been replaced by the set of equations (2.10)-(2.14).

(ii) In representing the outgoing wave as Karp's farfield expansion, $2L$ new unknown angular functions are introduced. They are the F and G families of angular functions.

(iii) To account for the unknown values of the scattered field u at the boundary, continuity of the scattered wave is imposed at $\mathcal{C}$.

(iv) If Karp's expansion were using only the first two angular functions F_0 and G_0, two

more conditions are needed in order to determine them. They are the continuity of the first radial derivative of the scattered wave across the artificial boundary, which are represented by condition (2.11) and condition (2.12), which imposes that $\mathcal{K}_L$ must satisfy the Helmholtz equation on $\mathcal{C}$ [13].

(v) Finally, anytime two new F_l and G_l angular functions are added to Karp's expansion, two more equations are needed. These two equations are supplied by the recurrence relations (2.6) and (2.7).

Also notice that when $r > R$, the scattered field $u(r,\theta) \approx \mathcal{K}_L(r,\theta)$. Specifically, $u(r,\theta) = \mathcal{K}_L(r,\theta) + O(r^{-(L+\frac{1}{2})})$. Thus, by numerically solving the single scattering problem with KFE, which is defined by (2.8)-(2.14), not only the scattered field inside the computational domain is obtained, but also the scattered field u anywhere in Ω^+. This is possible by simple evaluation of the Karp's expansion at every point outside the computational region Ω^-.

2.3.2 Extension of the local high order absorbing boundary condition KFE to Multiple Scattering. Our first step toward the formulation of the multiple scattering problem in terms of the KFE is to replace the Sommerfeld condition (2.3) by the KFE for each of the m scatterers. This is done by enclosing each obstacle by a circular artificial boundary $\mathcal{C}_m$ of radius R^m that separates each Ω_m into the bounded region Ω_m^- and the unbounded region Ω_m^+. We then apply a particular KFE absorbing boundary condition at each $\mathcal{C}_m$, which leads to the following family of BVPs ($m = 1 \dots M$), which are all coupled

through the boundary condition at the obstacle boundaries Γ_m:

$$\Delta u_m + k^2 u_m = 0, \quad \text{in } \Omega_m^-, \tag{2.16}$$

$$\mathcal{B} u_m = -\mathcal{B}\Big(u_{inc} + \sum_{m' \neq m} \mathcal{K}_{m',L}\Big), \quad \text{in } \Gamma_m \tag{2.17}$$

$$u_m(R^m, \theta^m) = \mathcal{K}_{m,L}(R^m, \theta^m), \quad \text{in } \mathcal{C}_m, \tag{2.18}$$

$$\frac{\partial u_m}{\partial r_m}(R^m, \theta^m) = \frac{\partial \mathcal{K}_{m,L}}{\partial r^m}(R^m, \theta^m) \quad \text{in } \mathcal{C}_m, \tag{2.19}$$

$$H_0(kR^m)\Big[(L-1)^2 F_{m,L-1} + d_{\theta^m}^2 F_{m,L-1}\Big] +$$
$$H_1(kR^m)\Big[L^2 G_{m,L-1} + d_{\theta^m}^2 G_{m,L-1}\Big] = 0, \tag{2.20}$$

$$2l\, G_{m,l} = (l-1)^2 F_{m,l-1} + d_{\theta^m}^2 F_{m,l-1} \tag{2.21}$$

$$2l\, F_{m,l} = -l^2 G_{m,l-1}(\theta) - d_\theta^2\, G_{m,l-1}, \tag{2.22}$$

$$\text{on } \mathcal{C}_m, \text{ for } m = 1, \dots M \text{ and } l = 1 \dots L-1,$$

At this point, we adopt an idea presented by Geuzaine et al. [9] to reformulate the BVP (2.16)-(2.22) as an iterative problem, which allows us to uncouple the BVP for one scatterer from the others. More precisely, by assuming that the outgoing waves from other scatterers are known from a previous iteration, it is possible to reduce the multiple scattering problem (2.17)-(2.22) to a family of single scattering problems. These can be solved individually in their respective local coordinate systems and then added together to obtain the total scattered field from all scatterers. Any iterative approach may serve our decoupling purposes, but we choose Jacobi and Gauss-Seidel for this end.

Gauss-Seidel iterative form:

$$\Delta u_m^{(n)} + k^2 u_m^{(n)} = 0, \quad \text{in } \Omega_m^-, \tag{2.23}$$

$$\mathcal{B}u_m^{(n)} = -\mathcal{B}u_{inc} + \begin{cases} -\sum_{m'=1}^{m-1} \mathcal{B}\mathcal{K}_{m',L}^{(n)}, & n = 1, \\ -\sum_{m'=1}^{m-1} \mathcal{B}\mathcal{K}_{m',L}^{(n)} - \sum_{m'=m+1}^{M} \mathcal{B}\mathcal{K}_{m',L}^{(n-1)}, & n > 1, \end{cases} \quad \text{on } \Gamma_m, \tag{2.24}$$

$$u_m^{(n)}(R^m, \theta^m) = \mathcal{K}_{m,L}^{(n)}(R^m, \theta^m)), \quad \text{in } \mathcal{C}_m, \tag{2.25}$$

$$\frac{\partial u_m^{(n)}}{\partial r_m}(R^m, \theta^m) = \frac{\partial \mathcal{K}_{m,L}^{(n)}}{\partial r^m}(R^m, \theta^m) \quad \text{in } \mathcal{C}_m, \tag{2.26}$$

$$H_0(kR^m)\Big[(L-1)^2 F_{m,L-1}^{(n)} + d_{\theta^m}^2 F_{m,L-1}^{(n)}\Big] +$$

$$H_1(kR^m)\Big[L^2 G_{m,L-1}^{(n)} + d_{\theta^m}^2 G_{m,L-1}^{(n)}\Big] = 0, \tag{2.27}$$

$$2l\, G_{m,l}^{(n)} = (l-1)^2 F_{m,l-1}^{(n)} + d_{\theta^m}^2 F_{m,l-1}^{(n)} \tag{2.28}$$

$$2l\, F_{m,l}^{(n)} = -l^2 G_{m,l-1}^{(n)} - d_\theta^2\, G_{m,l-1}^{(n)}, \tag{2.29}$$

$$\text{on } \mathcal{C}_m, \text{ for } m = 1, \dots M \text{ and } l = 1 \dots L-1, \ n = 1, \dots$$

For the Jacobi iterative form, the only change with respect to the Guass-Seidel form occurs at the scatterer boundary condition, as shown next.

Jacobi iterative form:

$$\mathcal{B}u_m^{(n)} = -\mathcal{B}u_{inc} + \begin{cases} 0, & n = 1, \\ -\sum_{m'=1}^{M} \mathcal{B}\mathcal{K}_{m',L}^{(n-1)}, & n > 1, \end{cases} \quad \text{on } \Gamma_m. \tag{2.30}$$

These iterative forms will continue until certain stopping criteria are met. If the iterations stop after N iterations, then we define $u_m = u_m^{(N)}$ and $\mathcal{K}_{m,L} = \mathcal{K}_{m,L}^{(N)}$. We can then reconstruct the scattered wave

$$u = \begin{cases} u_m + \sum_{m' \neq m} \mathcal{K}_{m',L}, & \text{in } \Omega_m^- \\ \sum_{m=1}^{M} \mathcal{K}_{m,L}, & \text{in } \Omega^+ \end{cases} \tag{2.31}$$

where $\Omega^+ = \bigcap_{m=1}^{M} \Omega_m^+$.

Chapter 3. Numerical Method

3.1 Elliptic-Polar Local Curvilinear Coordinates

As mentioned in equations (1.5) and (1.6), we have an analytical solution for the scattered wave off multiple cylinders. Thus, the purpose of numerically solving this BVP is to get an approximation for the scattered wave off of any obstacle shape, not just cylinders. In this section, we discuss two-dimensional generalized curvilinear coordinates, which are sufficiently smooth grids that conform to the obstacle boundary shape. I will follow closely with the development of this coordinate system found in Acosta-Villamizar [13].

We begin with a set of rectangular coordinates (ξ^m, η^m) defining a rectangular domain $\mathcal{R}^m \subset \mathbb{R}^2$, where $1 \leq \xi^m \leq N_1^m$, $1 \leq \eta^m \leq N_2^m$ for $m = 1, \ldots, M$. $\mathcal{R}^m$ represents the computational domain for each object m. Our local boundary conforming curvilinear coordinate system for each bounded subdomain Ω_m^- $(m = 1, \ldots, M)$ is given by an invertible and smooth transformation

$$T^m : \mathcal{R}^m \longrightarrow \Omega_m^-$$
$$(\xi^m, \eta^m) \mapsto \mathbf{x}(\xi^m, \eta^m)$$

where

$$\mathbf{x}(\xi^m, \eta^m) = (x(\xi^m, \eta^m), y(\xi^m, \eta^m)) \text{ and } 1 \leq \xi^m \leq N_1^m, 1 \leq \eta^m \leq N_2^m.$$

The coordinates (x, y) and (r, θ) denote the global cartesian and polar coordinates with respect to the fixed origin $\mathcal{O}$.

The transformation T^m is defined in [13] and is given as the solution to a BVP that is governed by the system of partial differential equations

$$\alpha x_{\xi\xi} - 2\beta x_{\xi\eta} + \gamma x_{\eta\eta} + \frac{1}{2}\alpha_\xi x_\xi + \frac{1}{2}\gamma_\eta x_\eta = 0, \text{ on } \mathcal{R}^m \tag{3.1}$$

$$\alpha y_{\xi\xi} - 2\beta y_{\xi\eta} + \gamma y_{\eta\eta} + \frac{1}{2}\alpha_\xi y_\xi + \frac{1}{2}\gamma_\eta y_\eta = 0, \text{ on } \mathcal{R}^m \tag{3.2}$$

$$\alpha = x_\eta^2 + y_\eta^2, \beta = x_\xi x_\eta + y_\xi y_\eta, \gamma = x_\xi^2 + y_\xi^2. \tag{3.3}$$

The boundary conditions the above PDE are the parametric curves $(x(\xi^m, 1), y(\xi^m, 1))$ and $(x(\xi^m, N_n^m), y(\xi^m, N_2^m))$, which describe the obstacle boundary and artificial boundary of obstacle m. The superscript m in (3.1) and (3.2) has been neglected for clarity.

3.1.1 Helmholtz Equation in Elliptic Polar Coordinates. The Helmholtz Equation in Cartesian coordinates is $\Delta f + k^2 f = 0$. The representation of the Helmholtz operator in generalized curvilinear coordinates is

$$\begin{aligned}\Delta f + k^2 f &= \frac{1}{J^2}(\alpha f_{\xi\xi} - 2\beta f_{\xi\eta} + \gamma f_{\eta\eta}) \\ &+ \frac{1}{J^3}(\alpha y_{\xi\xi} - 2\beta y_{\xi\eta} + \gamma y_{\eta\eta})(x_\eta f_\xi - x_\xi f_\eta) \\ &+ \frac{1}{J^3}(\alpha x_{\xi\xi} - 2\beta x_{\xi\eta} + \gamma x_{\eta\eta})(y_\xi f_\eta - y_\eta f_\xi) + k^2 f = 0 \end{aligned} \tag{3.4}$$

$$\alpha = x_\eta^2 + y_\eta^2, \beta = x_\xi x_\eta + y_\xi y_\eta, \gamma = x_\xi^2 + y_\xi^2, J = x_\xi y_\eta - x_\eta y_\xi. \tag{3.5}$$

We can simplify the Helmholtz equation in Elliptic Polar coordinates by substituting (3.1) and (3.2) into (3.4), with which we get

$$\Delta f + k^2 f = \frac{1}{J^2}(\alpha f_{\xi\xi} - 2\beta f_{\eta\eta} + \gamma f_{\eta\eta} + \frac{1}{2}(\alpha_\xi f_\xi + \gamma_\eta f_\eta)) + k^2 f = 0. \tag{3.6}$$

So, our scattered wave u^m must satisfy the Helmholtz equation in elliptic polar curvilinear coordinates in each Ω_m^-, and this is given by equation (3.6).

3.1.2 Boundary Conditions in Elliptic Polar Coordinates. The boundary condition on Γ_m in our Cartesian coordinate system is given by $\mathcal{B}u_m = -\mathcal{B}(u_m + \sum_{m' \neq m} u_{m'})$, where $\mathcal{B}u = Z\frac{\partial u}{\partial \nu} + (1-Z)u$. All that needs to be done is to find the elliptic-polar representation of $\mathcal{B}$. This is done by replacing $\frac{\partial u}{\partial \nu}$ with it's elliptic polar counterpart. We first note that the partial derivative with respect to the normal direction can be written as $\frac{\partial u}{\partial \nu} = \nu \cdot \nabla u$. The

normal vector at the obstacle boundary in curvilinear coordinates is given by $\frac{1}{\gamma}[-y_\xi, x_\xi]^T$, and the gradient of u is given by $\frac{1}{J}[u_\xi y_\eta - u_\eta y_\xi, u_\eta x_\xi - u_\eta x_\eta]^T$ in Ω_m^-. At the artificial boundary at radius R^m, the normal vector is simply given by $\nu = \frac{1}{R^m}[x, y]^T$. These quantaties can be summed up as

$$\frac{\partial u}{\partial \nu} = \nu \cdot \nabla u = \begin{cases} \frac{1}{\gamma} \begin{bmatrix} -y_\xi \\ x_\xi \end{bmatrix} \cdot \frac{1}{J} \begin{bmatrix} u_\xi y_\eta - u_\eta y_\xi \\ u_\eta x_\xi - u_\eta x_\eta \end{bmatrix} & \text{in } \Gamma_m \\ \frac{1}{R^m} \begin{bmatrix} x \\ y \end{bmatrix} \cdot \frac{1}{J} \begin{bmatrix} u_\xi y_\eta - u_\eta y_\xi \\ u_\eta x_\xi - u_\eta x_\eta \end{bmatrix} & \text{in } \mathcal{C}_m \end{cases}. \tag{3.7}$$

The partial derivative in the operator $\mathcal{B}$ can be replaced by (3.7) to get the elliptic-polar representation.

3.2 Grids and Finite Difference Numerical Scheme

In this section, we will generate our grids and then set up our finite difference scheme.

3.2.1 Generating the Elliptic Polar Grids. We will have a global coordinate system with global origin $(0, 0)$. Each obstacle will be centered at local origin $\vec{b}_m = (\bar{x}_m, \bar{y}_m)$. For each obstacle, we will generate a grid to do computation on. The grid will be generated using obstacle m's local Cartesian coordinate system (x^m, y^m). In this subsection, we assume we will be working in the local Cartesian coordinate system of obstacle m, so we will drop many of the superscripts.

In the local Cartesian coordinate system, the boundary conditions for the grid generating ODE are determined by the parametric functions that define the boundary of the obstacle, $\mathcal{B}_{in}^m(\theta^m) = [B_x^m(\theta^m), B_y^m(\theta^m)]^T$ where θ^m is the local polar coordinate system. The outer boundary parametric equation is $\mathcal{B}_{out}^m(\theta^m) = [R^m \cos(\theta^m), R^m \sin(\theta^m)]^T$, where R^m is the radius of the outer boundary.

We select the values of N_1^m and N_2^m based on a desired amounts of points per wave-

length (PPW). Points per wavelength is a good measure of refinement for wave scattering problems, and is how many grid points are contained in a wavelength of the incoming wave. Let r_0^m denote the average radius of the inner boundary of obstacle m, that is $r_0^m = \frac{1}{2\pi}\int_0^{2\pi}\sqrt{(B_x^m(\theta^m))^2+(B_y^m(\theta^m))^2}d\theta^m$. Then

$$N_1^m = \left\lceil PPW(2\pi\frac{R^m+r_0^m}{2})\frac{k}{2\pi}\right\rceil \text{ and } N_2^m = \left\lceil PPW(R^m-r_0^m)\frac{k}{2\pi}\right\rceil \tag{3.8}$$

where $\lceil\ \rceil$ represents the ceiling function.

We will now descritize our rectangular computational domain $\mathcal{R}_m$ defined by $1 \le \xi^m \le N_1^m, 1 \le \eta^m \le N_2^m$. Our discrete steps will be integer steps starting at 1, so $\Delta\xi^m = 1$ and $\Delta\eta^m = 1$. Essentially, $\xi_i^m = i$ and $\eta_j^m = j$ for $1 \le i \le N_1^m, 1 \le j \le N_2^m$. To map this to the region Ω_m^-, we first make a branch cut along the positive x-axis. This branch cut will turn our multiply connected region into a simply connected region. We discretize θ^m from 2π to 0 into N_1^m equally spaced points, so $\theta_i^m = 2\pi - \frac{(i-1)*2\pi}{N_1^m-1}$. We then can set our boundary conditions for our grid generating PDE:

$$(x(\xi_i^m,1), y(\xi_i^m,1)) = (\mathcal{B}_x^m(\theta_i^m), \mathcal{B}_y^m(\theta_i^m)) \text{ for } i \in \{1,\dots,N_1^m\} \tag{3.9}$$

$$(x(\xi_i^m,N_n^m), y(\xi_i^m,N_2^m)) = (R^m\cos(\theta_i^m), R^m\sin(\theta_i^m)) \text{ for } i \in \{1,\dots,N_1^m\}. \tag{3.10}$$

Now that we have our boundary conditions for our grid generating PDE set, what is left is to solve for the interior points $\mathbf{x}(\xi_i^m,\eta_j^m)$ for $i \in \{2,\dots,N_1^m-1\}$ and $j \in \{1,\dots,N_2^m\}$. Once we solve for these, we will then have a sufficiently smooth grid to do finite difference computation on, as well as a discrete mapping from $\mathcal{R}^m$ to Ω_m^-. This is done by using SOR iteration on an initial grid. The initial grid we have used is a polar-like grid, where we have created a polar grid with the same θ^m discretization as before, and a linear discretization of the interval $[r_{min}^m, R^m]$, with $r_{min}^m = \sup_{\theta^m\in[0,2\pi]}\sqrt{(\mathcal{B}_x^m(\theta^m))^2+(\mathcal{B}_y^m(\theta^m))^2}$, excluding the $\eta^m = 1$ grid-points. So, the radius at $\eta^m = j$ for our initial grid is $r_{min}^m + \frac{(j-1)(R^m-r_{min}^m)}{N_2^m-1}$ for

$j \in \{2, \ldots, N_2^m\}$. Putting this together, the initial grid is

$$(x(\xi_i^m, 1), y(\xi_i^m, 1)) = (\mathcal{B}_x^m(\theta_i^m), \mathcal{B}_y^m(\theta_i^m)) \text{ for } i \in \{1, \ldots, N_1^m\} \tag{3.11}$$

$$(x(\xi_i^m, \eta_j^m), y(\xi_i^m, \eta_j^m)) = (r_{min}^m + \frac{(j-1)(R^m - r_{min}^m)}{N_2^m - 1})(\cos\theta_i^m, \sin\theta_i^m), \tag{3.12}$$

$$\text{for } i \in \{1, \ldots, N_1^m\}, j \in \{2, \ldots, N_2^m\}.$$

We then perform SOR iteration on this initial grid until it converges to a grid that satisfies (3.1) and (3.2). This SOR iteration will begin with the points $(x_{i,j}^{(0)}, y_{i,j}^{(0)})$ that are given by (3.11) and (3.12). The update equation for $x_{i,j}^{(k+1)}$ is given by

$$\begin{aligned} x_{i,j}^{(k+1)} = & \frac{1}{2(\alpha+\gamma)_{i,j}}[\alpha_{i,j}(x_{i+1,j}^{(k)} + x_{i-1,j}^{(k+1)}) + \gamma_{i,j}(x_{i,j+1}^{(k)} + x_{i,j-1}^{(k+1)}) \\ & + (2x_\xi x_\eta x_{\xi\eta} + y_{\xi\eta}(x_\xi y_\eta + x_\eta y_\xi))_{i,j} - 2\beta_{i,j}(x_{\xi\eta})_{i,j}] \end{aligned} \tag{3.13}$$

where

$$\alpha_{i,j} = (x_\eta)_{i,j}^2 + (y_\eta)_{i,j}^2, \quad \beta_{i,j} = (x_\xi)_{i,j}(x_\eta)_{i,j} + (y_\xi)_{i,j}(y_\eta)_{i,j}, \quad \gamma_{i,j} = (x_\xi)_{i,j}^2 + (y_\xi)_{i,j}^2$$

and

$$\begin{aligned} (x_\xi)_{i,j} &= (x_{i+1,j}^{(k)} - x_{i-1,j}^{(k+1)})/2, \quad (x_\eta)_{i,j} = (x_{i,j+1}^{(k)} - x_{i,j-1}^{(k+1)})/2 \\ (y_\xi)_{i,j} &= (y_{i+1,j}^{(k)} - y_{i-1,j}^{(k+1)})/2, \quad (y_\eta)_{i,j} = (y_{i,j+1}^{(k)} - y_{i,j-1}^{(k+1)})/2 \\ (x_{\xi\eta})_{i,j} &= (x_{i+1,j+1}^{(k)} - x_{i+1,j-1}^{(k)} - x_{i-1,j+1}^{(k+1)} + x_{i-1,j-1}^{(k+1)})/4 \\ (y_{\xi\eta})_{i,j} &= (y_{i+1,j+1}^{(k)} - y_{i+1,j-1}^{(k)} - y_{i-1,j+1}^{(k+1)} + y_{i-1,j-1}^{(k+1)})/4. \end{aligned}$$

The update formula for $y^{(k+1)}$ is the same as (3.13) but with the x's and y's switched. The SOR iteration uses a relaxation parameter ω to update $x_{i,j}^{(k+1)}$ as $x_{i,j}^{(k+1)} = \omega x_{i,j}^{(k+1)} + (1-\omega)x_{i,j}^{(k)}$. We will define the maximum pointwise error as

$$\varepsilon^{k+1} = \max_{\substack{1 \le i \le N_1^m \\ 1 \le j \le N_2^m}} \{|x_{i,j}^{(k+1)} - x_{i,j}^{(k)}|, |y_{i,j}^{(k+1)} - y_{i,j}^{(k)}|\}.$$

We will then iterate our grid update step until our maximum pointwise error falls below a specified tolerance, $\varepsilon^{k+1} < Tol$. We have then generated a sufficiently smooth computational grid for obstacle m. This is repeated for every obstacle.

At this point we have an Elliptic-Polar grid that represents the computational domain

Ω_m^- in each local Cartesian coordinate system. For the local discretized grid, let us call $\bar{X}^m$ the $N_1^m \times N_2^m$ sized array where $\bar{X}_{i,j}^m = x^m(\xi_i, \eta_j)$, and likewise $\bar{Y}_{i,j}^m = y^m(\xi_i, \eta_j)$.

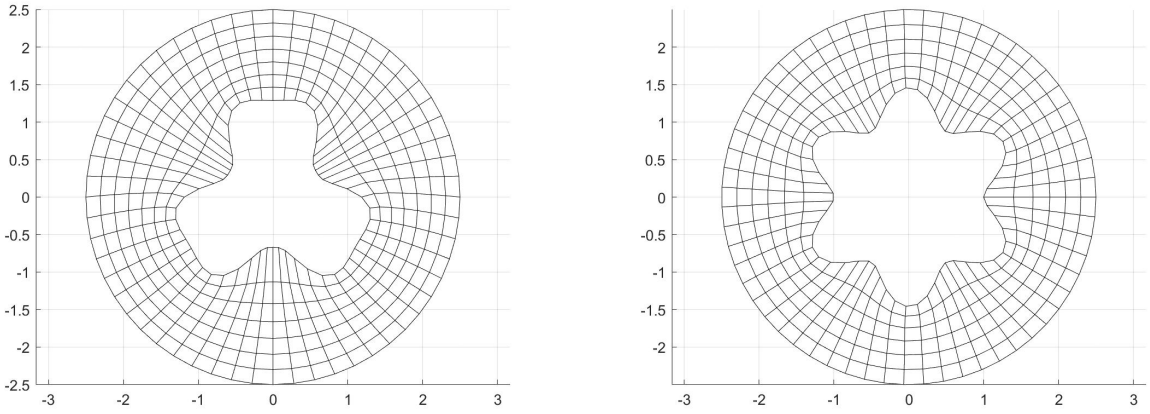
For an example, let us use $M = 2$ obstacles with complexly shaped inner boundaries. For this example, let

$$\mathcal{B}_{in}^1 = \begin{bmatrix} (1+\frac{1}{3}\cos(3\cos(1.75\cos(1.5\theta^1))))\cos(\theta^1) \\ (1+\frac{1}{3}\cos(3\cos(1.75\cos(1.5\theta^1))))\sin(\theta^1) \end{bmatrix} \text{ and}$$

$$\mathcal{B}_{in}^2 = \begin{bmatrix} (2-\cos(\sin(3\theta^2)))\cos(\theta^2) \\ (2-\cos(\sin(3\theta^2)))\sin(\theta^2) \end{bmatrix}.$$

Also, let the outer radii for the absorbing boundary be $R^m = 2.5$ for both obstacles. Figure 3.1 gives a possible grid that can be generated for these obstacles.

Figure 3.1: Computed computational domains for two example boundaries.



(a) Local $\bar{\Omega}_1^-$ (b) Local $\bar{\Omega}_2^-$

3.2.2 Grid representation in Global Cartesian and Local Polar. In our finite difference numerical scheme, we will be switching many times from global Cartesian coordinates to local Cartesian coordinates to local polar coordinates. In this section, we show how to calculate the various representations of our computational domain.

The center of obstacle m is located at $\vec{b}_m = (\bar{x}_m, \bar{y}_m)$. This point serves as the origin for the local coordinate system (x_m, y_m), which $\bar{X}^m$ and $\bar{Y}^m$ are given in. Let X^m and Y^m

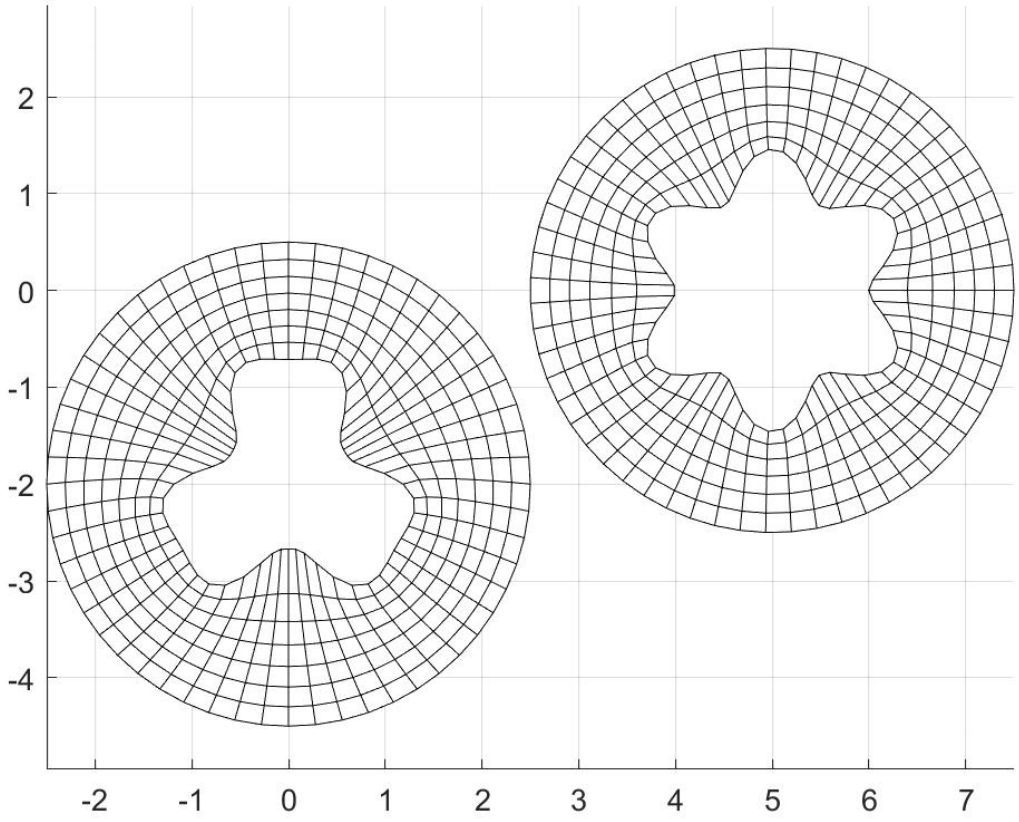
denote the global representation of the local grid $\bar{X}^m$ and $\bar{Y}^m$, respectfully.

$$X^m = \bar{X}^m + \bar{x}^m \tag{3.14}$$

$$Y^m = \bar{Y}^m + \bar{y}^m. \tag{3.15}$$

Continuing the example given in Figure 3.1, let's say $\vec{b}_1 = [0, -2]^T$ and $\vec{b}_2 = [5, 0]^T$. Then the global grid representation of the local grids in Figure 3.1 are given in Figure 3.2.

Figure 3.2: Global $\bar{\Omega}_1^-$ and $\bar{\Omega}_2^-$ for our example grids.



There are also interactions between obstacle m and obstacle m'. These interactions occur in the local polar coordinate system of obstacle m', so it is necessary to get a representation of our grid (X^m, Y^m) in terms of the local polar coordinate system $(r^{m'}, \theta^{m'})$, which is centered at $(\bar{x}^{m'}, \bar{y}^{m'})$. Let us define $\tilde{X}^{(m,m')}$ and $\tilde{Y}^{(m,m')}$ to be the representation of X^m and Y^m in

the local Cartesian coordinate system $(x^{m'}, y^{m'})$. These are given by

$$\tilde{X}^{(m,m')} = X^m - \bar{x}^{m'},$$

$$\tilde{Y}^{(m,m')} = Y^m - \bar{y}^{m'}.$$

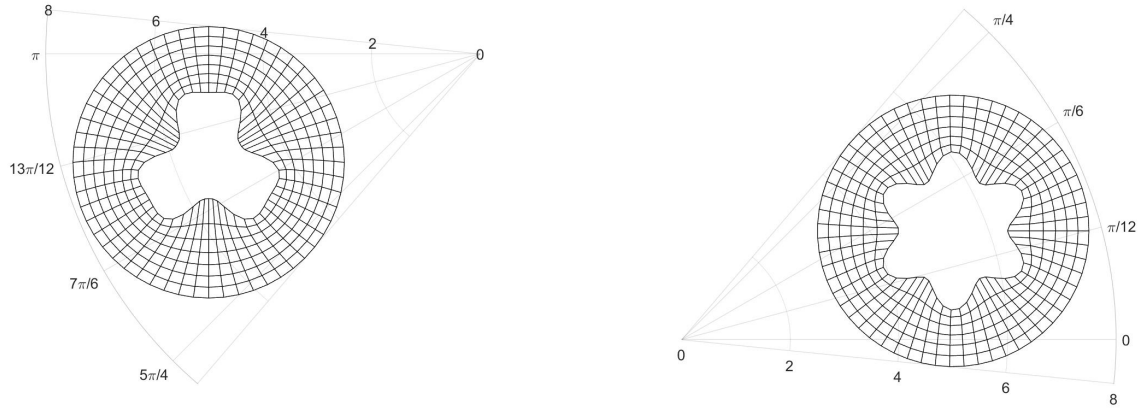
We then can get the local polar representation

$$R^{(m,m')} = \sqrt{(\tilde{X}^{(m,m')})^2 + (\tilde{Y}^{(m,m')})^2} \tag{3.16}$$

$$\Theta^{(m,m')} = \arctan \frac{\tilde{Y}^{(m,m')}}{\tilde{X}^{(m,m')}}. \tag{3.17}$$

The local polar representations for our example grids are found in Figure 3.3. The grid shapes in Figures 3.3a and 3.3b are identical to the grids in Figures 3.1a and 3.1b. However, the grids in 3.3 are represented by arrays of radii and arrays of angle values, instead of arrays of x values and y values.

Figure 3.3: Polar grid representations for our example grids.



(a) Polar representation of $\bar{\Omega}_1^-$ in (r^2, θ^2).

(b) Polar representation of $\bar{\Omega}_2^-$ in (r^1, θ^1).

These grid representations (X^m, Y^m), $(\bar{X}^m, \bar{Y}^m)$, and $(R^{(m,m')}, \Theta^{(m,m')})$ are all representations of Ω_m^- in various coordinate systems, and will all come in handy in our finite difference numerical scheme.

3.2.3 Finite Difference Discretizations. Thus far, we have discretized our computational domains Ω_m^- for all obstacles $m \in \{1, \ldots, M\}$, and now have representations of that discretized domain in local Cartesian (x^m, y^m), global Cartesian (x, y), and every local polar

$(r^{m'}, \theta^{m'})$. It is now time to discretize our governing equations (2.23) - (2.29). We will use second order finite difference. Throughout this subsection, we will derive everything for one obstacle and one iteration, and thus we will drop the sub- and super-scripts that denote the obstacle and iteration. We will assume we are working with obstacle m. Thus, $u_m^{(n)}$,$\mathcal{K}_{m,L}^{(n)}$, $F_{m,l}^{(n)}$ and $G_{m,l}^{(n)}$ will be called u, $\mathcal{K}_L$, F_l and G_l, respectfully. We will add a sub- or superscript of m' where appropriate.

At this point, we will discretize our solution u to have function values at each of the points (X^m, Y^m). We will call this discretization U^m or just U for most cases, where U is an $(N_1-1)\times(N_2+1)$ array where $U_{i,j}$ represents our function value of u at the point $(X_{i,j}^m, Y_{i,j}^m)$. We eliminate the N_1 row from U because $U_{1,\eta} = U_{N_1,\eta}$ for all $\eta \in \{1, \ldots, N_2+1\}$. Thus, in many of our discretized formulas, we assume $U_{-1,\eta} = U_{N_1-1,\eta}$ and $U_{N_1,\eta} = U_{1,\eta}$ for all $\eta \in \{1, \ldots, N_2+1\}$. We also add a column of ghost points at the $\eta = N_2+1$ level to simplify some calculations. We also will discretize the family of farfield functions F_l^m and G_l^m to be $\bar{F}_l^m$ and $\bar{G}_l^m$, which are vectors of length N_1-1 such that $(\bar{F}_l^m)_i$ represents $F_l^m(\theta_i^m)$ and $(\bar{G}^m)_i$ represents $G_l^m(\theta_i^m)$ for all $i \in \{1, \ldots, N_1-1\}$ and all $m \in \{1, \ldots, M\}$. As a reminder, each θ^m is discretized with $\theta_i^m = 2\pi - \frac{(i-1)*2\pi}{N_1-1}$ for all $m \in \{1, \ldots, M\}$. We also will introduce discretized boundaries and computational domains. Let $(\bar{\Omega}_m^-)_{i,j} = (X_{i,j}^m, Y_{i,j}^m)$, and $(\bar{\Gamma}_m)_i = (X_{i,1}^m, Y_{i,1}^m)$.

3.2.4 Finding $\mathcal{K}_{m',L}$ on $\bar{\Gamma}_m$ and in $\bar{\Omega}_m$. Given discretized $\bar{F}_{m',l}$ and $\bar{G}_{m',l}$, which are discretized as equally spaced intervals in $\theta^{m'}$ for $l \in \{0, \ldots, L-1\}$, we will oftentimes need to find the value of $\mathcal{K}_{m',L}$ on the discretized domains $\bar{\Gamma}_m$ and in the region $\bar{\Omega}_m^-$. As a reminder, $(R^{(m,m')}, \Theta^{(m,m')})$ is the representation of $\bar{\Omega}_m^-$ in the local polar coordinates of obstacle m'. This means $(R_{i,1}^{(m,m')}, \Theta_{i,1}^{(m,m')})$ represents $(\bar{\Gamma}_m)_i$. Since $\bar{F}_{m',l}$ and $\bar{G}_{m',l}$ are at equally spaced intervals θ_i^m, we will need to create an interpolating function to approximate the values of $F_{m',l}$ and $G_{m',l}$ at intermediate angles θ^m. Let $\tilde{F}_{m',l}$ and $\tilde{G}_{m',l}$ be the interpolating function based off the arrays $\bar{F}_{m',l}$ and $\bar{G}_{m',l}$, respectfully. This interpolating function can be any interpolating scheme for equally spaced intervals that is at least second order accurate, so

that our overall scheme stays second order. We've had success with third order Lagrangian Interpolation. When evaluating $\mathcal{K}_{m',L}$ at a point $(\bar{\Omega}_m^-)_{i,j}$, we will use the representation of $\bar{\Omega}_m^-$ in the local polar coordinate system of m', so $\bar{\Omega}_m^-$ is represented by $(R^{(m',m)}, \Theta^{(m',m)})$. Once you have the interpolating functions $\tilde{F}_{m',l}$ and $\tilde{G}_{m',l}$, then

$$\begin{aligned}(\mathcal{K}_{m',L}(\bar{\Omega}_m^-))_{i,j} &= \mathcal{K}_{m',L}(R_{i,j}^{(m',m)}, \Theta_{i,j}^{(m',m)}) \\ &= H_0(kR_{i,j}^{(m',m)}) \sum_{l=0}^{L-1} \frac{\tilde{F}_{m',l}(\Theta_{i,j}^{(m',m)})}{(kR_{i,j}^{(m',m)})^l} + H_1(kR_{i,j}^{(m',m)}) \sum_{l=0}^{L-1} \frac{\tilde{G}_{m',l}(\Theta_{i,j}^{(m',m)})}{(kR_{i,j}^{(m',m)})^l} \end{aligned} \tag{3.18}$$

and

$$(\mathcal{K}_{m',L}(\bar{\Gamma}_m))_i = (\mathcal{K}_{m',L}(\bar{\Omega}_m^-))_{i,1}. \tag{3.19}$$

3.2.5 Discretizing the Helmholtz Operator and (2.23). Equation (2.23) shows that the single scattering solution $u_m^{(n)}$ satisfies the Helmholtz equation in the region Ω_m^-. Since (X^m, Y^m) is an elliptic-polar grid, then we will discretize the elliptic-polar Helmholtz equation given by (3.6). As a recap, the scaling factors are

$$\alpha = x_\eta^2 + y_\eta^2, \quad \beta = x_\xi x_\eta + y_\xi y_\eta, \quad \gamma = x_\xi^2 + y_\xi^2, \quad J = x_\xi y_\eta - x_\eta y_\xi$$

and the Helmholtz operator in elliptic-polar coordinates, which we will call $\mathcal{H}^{ep}$ applied to u is

$$\mathcal{H}^{ep}u = \frac{1}{J^2}(\alpha u_{\xi\xi} - 2\beta u_{\xi\eta} + \gamma u_{\eta\eta} + \frac{1}{2}(\alpha_\xi u_\xi + \gamma_\eta u_\eta)) + k^2 u. \tag{3.20}$$

We will use second order finite difference to discretize $x_\xi, y_\xi, x_\eta,$ and y_η:

$$\begin{aligned}(X_\xi)_{i,j} &= (X_{i+1,j} - X_{i-1,j})/2, \quad (X_\eta)_{i,j} = (X_{i,j+1} - X_{i,j-1})/2 \\ (Y_\xi)_{i,j} &= (Y_{i+1,j} - Y_{i-1,j})/2, \quad (Y_\eta)_{i,j} = (Y_{i,j+1} - Y_{i,j-1})/2 \\ (X_{\xi\eta})_{i,j} &= (X_{i+1,j+1} - X_{i+1,j-1} - X_{i-1,j+1} + X_{i-1,j-1})/4 \\ (Y_{\xi\eta})_{i,j} &= (Y_{i+1,j+1} - Y_{i+1,j-1} - Y_{i-1,j+1} + Y_{i-1,j-1})/4.\end{aligned}$$

The discretized scaling factors and specific scaling factor derivatives are

$$\alpha_{i,j} = (X_\eta)^2_{i,j} + (Y_\eta)^2_{i,j}, \quad \beta_{i,j} = (X_\xi)_{i,j}(X_\eta)_{i,j} + (Y_\xi)_{i,j}(Y_\eta)_{i,j},$$
$$\gamma_{i,j} = (X_\xi)^2_{i,j} + (Y_\xi)^2_{i,j}, \quad J_{i,j} = (X_\xi)_{i,j}(Y_\eta)_{i,j} - (X_\eta)_{i,j}(Y_\xi)_{i,j}$$
$$(\alpha_\xi)_{i,j} = 2(X_\eta X_{\xi\eta} + Y_\eta Y_{\xi\eta})_{i,j}, \quad (\gamma_\eta)_{i,j} = 2(X_\xi X_{\xi\eta} + Y_\xi Y_{\xi\eta})_{i,j}.$$

We also discretize the derivatives of $U_{i,j}$ as such:

$$(U_\xi)_{i,j} = (U_{i+1,j} - U_{i-1,j})/2, \quad (U_{\xi\xi})_{i,j} = (U_{i+1,j} - 2U_{i,j} + U_{i-1,j})$$
$$(U_\eta)_{i,j} = (U_{i,j+1} - U_{i,j-1})/2, \quad (U_{\eta\eta})_{i,j} = (U_{i,j+1} - 2U_{i,j} + U_{i,j-1})$$
$$(U_{\xi\eta})_{i,j} = (U_{i+1,j+1} - U_{i+1,j-1} - U_{i-1,j+1} + U_{i-1,j-1})/4.$$

When we substitute the finite difference approximations into (3.20), we arrive at the discretized Helmholtz operator, which we will call $\hat{\mathcal{H}}^{ep}$, in elliptic polar coordinates

$$\begin{aligned}
\hat{\mathcal{H}}^{ep}U_{i,j} &= (k^2 - \frac{2}{J^2_{i,j}}(\alpha_{i,j} + \gamma_{i,j}))U_{i,j} + (\frac{1}{J^2_{i,j}}(\alpha_{i,j} - \frac{1}{4}(\alpha_\xi)_{i,j}))U_{i-1,j} \\
&+ (\frac{1}{J^2_{i,j}}(\alpha_{i,j} + \frac{1}{4}(\alpha_\xi)_{i,j}))U_{i+1,j} + (\frac{1}{J^2_{i,j}}(\gamma_{i,j} - \frac{1}{4}(\gamma_\eta)_{i,j}))U_{i,j-1} \\
&+ (\frac{1}{J^2_{i,j}}(\gamma_{i,j} + \frac{1}{4}(\gamma_\eta)_{i,j}))U_{i,j+1} - \frac{\beta_{i,j}}{2J^2_{i,j}}(U_{i+1,j+1} - U_{i+1,j-1} - U_{i-1,j+1} + U_{i-1,j-1}) \\
&\text{for } 1 \le i \le N_1^m - 1 \text{ and } 2 \le j \le N_2^m - 1. \qquad (3.21)
\end{aligned}$$

Then the full discretization of (2.23) is

$$\hat{\mathcal{H}}^{ep}U_{i,j} = 0 \quad \text{for} \quad 1 \le i \le N_1^m - 1 \text{ and } 2 \le j \le N_2^m \qquad (3.22)$$
$$\iff J^2_{i,j}\hat{\mathcal{H}}^{ep}U_{i,j} = 0 \quad \text{for} \quad 1 \le i \le N_1^m - 1 \text{ and } 2 \le j \le N_2^m \qquad (3.23)$$

(3.22) is the discretization of (2.23), but we often get scaling issues if we use this formulation. In practice, we use the equivalent discretization (3.23).

3.2.6 Discretizing the Inner Boundary Condition (2.24). To discretize (2.24), we need to discretize the boundary operator $\mathcal{B}$ defined by $\mathcal{B}u = Z\frac{\partial u}{\partial \nu} + (1-Z)u$. As given in

(3.7), we have that

$$\frac{\partial u}{\partial \nu} = \nu \cdot \nabla u = \frac{1}{\gamma} \begin{bmatrix} -y_\xi \\ x_\xi \end{bmatrix} \cdot \frac{1}{J} \begin{bmatrix} u_\xi y_\eta - u_\eta y_\xi \\ u_\eta x_\xi - u_\eta x_\eta \end{bmatrix}.$$

Expanding the dot product out and substituting the scaling factors where appropriate, we have

$$\frac{\partial u}{\partial \nu} = \frac{1}{J\gamma}(-\beta u_\xi + \gamma u_\eta). \tag{3.24}$$

We can then discretize the operator $\frac{\partial}{\partial \nu}$ by substituting the proper second order finite difference approximations. However, we are at the $\eta = 1$ level at the obstacle boundary, so we cannot use centered difference in η. We then use second order one-sided difference,

$$(U_\eta)_{i,1} = (-3U_{i,1} + 4U_{i,2} - U_{i,3})/2$$

$$(X_{\xi\eta})_{i,1} = (-3X_{i+1,1} + 4x_{i+1,2} - X_{i+1,3} + 3X_{i-1,1} - 4X_{i-1,2} + X_{i-1,3})/4$$

with $(X_\eta)_{i,1}$, $(Y_\eta)_{i,1}$ and $(Y_{\xi\eta})_{i,1}$ defined similarly. Then $\hat{\mathcal{B}}U_{i,1} = Z\frac{\hat{\partial}}{\partial \nu}U_{i,1} + (1-Z)U_{i,1} = Z\frac{1}{(J\gamma)_{i,1}}(-\beta_{i,1}(U_\xi)_{i,1} + \gamma_{i,1}(U_\eta)_{i,1}) + (1-Z)U_{i,1}$.

Substituting in the discretizations, we arrive at

$$\begin{aligned} \hat{\mathcal{B}}U_{i,1} = {} & (\frac{Z\beta_{i,1}}{2(J\gamma)_{i,1}})U_{i-1,1} + (-\frac{3Z}{2J_{i,1}} + 1 - Z)U_{i,1} \\ & + (-\frac{Z\beta_{i,1}}{2(J\gamma)_{i,1}})U_{i+1,1} + (\frac{2Z}{J_{i,1}})U_{i,2} + (-\frac{Z}{2J_{i,1}})U_{i,3}. \end{aligned} \tag{3.25}$$

The discretized inner boundary condition is then given by

$$\hat{\mathcal{B}}U_{i,1} = -\hat{\mathcal{B}}U_{inc,i,1} + \begin{cases} -\sum_{m'=1}^{m-1} \hat{\mathcal{B}}(\mathcal{K}^{(n)}_{m',L}((\bar{\Gamma}_m))_i, & n = 1, \\ -\sum_{m'=1}^{m-1} \hat{\mathcal{B}}(\mathcal{K}^{(n)}_{m',L}(\bar{\Gamma}_m))_i - \sum_{m'=m+1}^{M} \hat{\mathcal{B}}(\mathcal{K}^{(n-1)}_{m',L}(\bar{\Gamma}_m))_i, & n > 1, \end{cases}$$

$$\text{for } i \in \{1, \ldots, N_1 - 1\} \tag{3.26}$$

where $(\mathcal{K}^{(n)}_{m',L}((\bar{\Gamma}_m))_i$ is given by (3.19).

3.2.7 Discretizing the Continuity Condition at the Artificial Boundary, (2.25).

This condition states that $u(R,\theta) = \mathcal{K}_L(R,\theta)$. Discretized, this condition is

$$U_{i,N_2} - H_0(kR)\sum_{l=0}^{L-1}\frac{F_{l,i}}{(kR)^l} - H_1(kR)\sum_{l=0}^{L-1}\frac{G_{l,i}}{(kR)^l} = 0. \tag{3.27}$$

3.2.8 Discretizing the Continuity of the First Derivative Condition at the Artificial Boundary, (2.26).

Condition (2.26) states that $\frac{\partial u}{\partial r}(R,\theta) = \frac{\partial \mathcal{K}_L}{\partial r}(R,\theta)$. We can analytically take the derivative of $\mathcal{K}_L$ with respect to r, and then evaluate at $r = R$ and $\theta = \theta_i$ to get

$$\begin{aligned}
&\frac{\partial}{\partial r}\mathcal{K}_L(R,\theta_i) = \sum_{l=0}^{L-1} A_l(kR)F_{l,i} + \sum_{l=0}^{L-1} B_l(kR)G_{l,i} \\
&Al(kR) = \frac{kH_1(kR)}{(kR)^l} + \frac{klH_0(kR)}{(kR)^{l+1}} \quad \text{and} \quad B_l(kR) = \frac{k(l+1)H_1(kR)}{(kR)^{l+1}} - \frac{kH_0(kR)}{(kR)^l}.
\end{aligned} \tag{3.28}$$

The derivative $\frac{\partial}{\partial r}$ is actually the normal derivative on the artificial boundary, so

$$\begin{aligned}
&\frac{\partial}{\partial r}u = \frac{\partial}{\partial \nu}u = \nu\cdot\nabla u = \frac{1}{R}\begin{bmatrix} x \\ y \end{bmatrix}\cdot\frac{1}{J}\begin{bmatrix} u_\xi y_\eta - u_\eta y_\xi \\ u_\eta x_\xi - u_\eta x_\eta \end{bmatrix} = \frac{1}{RJ}(\mu u_\xi + \lambda u_\eta) \\
&\text{where } \mu = xy_\eta - yx_\eta \quad \text{and} \quad \lambda = yx_\xi - xy_\xi.
\end{aligned} \tag{3.29}$$

We then substitute the second order finite difference approximations for u_η and u_ξ into (3.29). We use centered finite difference for u_ξ. The ghost points we have introduced, U_{i,N_2+1}, allows us to use centered finite difference for u_η as well.

$$\begin{aligned}
&\frac{\hat{\partial}}{\partial r}U_{i,N_2} = (\frac{\mu_i}{2RJ_{i,N_2}})U_{i+1,N_2} + (\frac{-\mu_i}{2RJ_{i,N_2}})U_{i-1,N_2} + \\
&(\frac{\lambda_i}{2RJ_{i,N_2}})U_{i,N_2+1} + (\frac{-\lambda_i}{2RJ_{i,N_2}})U_{i,N_2-1}.
\end{aligned} \tag{3.30}$$

Our final discretized condition is

$$\begin{aligned}
&\frac{\hat{\partial}}{\partial r}U_{i,N_2} = \frac{\partial}{\partial r}\mathcal{K}_L(R,\theta_i) \iff \frac{\hat{\partial}}{\partial r}U_{i,N_2} - \frac{\partial}{\partial r}\mathcal{K}_L(R,\theta_i) = 0 \\
&\text{for all } i \in \{1,\ldots,N_1-1\}.
\end{aligned} \tag{3.31}$$

We gather all of the unknowns we are trying to solve onto one side, so we use the equivalent formulation in (3.31).

3.2.9 The Final Three Conditions. The discretized conditions we have done thus far are the more difficult discretizations. The final 3 conditions, (2.27)-(2.29), are relatively simple since the only variable we are discretizing with respect to is θ. The operator we are working with the most is the operator d_θ^2. We can discretize this operator with second order finite difference, and we get

$$\hat{d_\theta^2} F_{l,i} = (F_{l,i-1} - 2F_{l,i} + F_{l,i+1})/(\Delta\theta)^2 \tag{3.32}$$

where $\Delta\theta = -\frac{2\pi}{N_1 - 1}$.

For (2.27), we insert (3.32) where appropriate and evaluate the functions at θ_i. We arrive at

$$\begin{aligned} &H_0(kR)[((L-1)^2 - \frac{2}{(\Delta\theta)^2})F_{L-1,i} + (\frac{1}{(\Delta\theta)^2})F_{L-1,i-1} + (\frac{1}{(\Delta\theta)^2})F_{L-1,i+1}] \\ &+ H_1(KR)[(L^2 - \frac{2}{(\Delta\theta)^2})G_{L-1,i} + (\frac{1}{(\Delta\theta)^2})G_{L-1,i-1} + (\frac{1}{(\Delta\theta)^2})G_{L-1,i+1}] = 0 \\ &\quad \text{for all} \quad i \in \{1, \ldots, N_1 - 1\}. \end{aligned} \tag{3.33}$$

For (2.28), we substitute (3.32) where appropriate and evaluate the functions at $\theta = \theta_i$. We then get

$$\begin{aligned} &2lG_{l,i} = (l-1)^2 F_{l-1,i} + \frac{1}{(\Delta\theta)^2}(F_{l-1,i-1} - 2F_{l-1,i} + F_{l-1,i+1}) \\ &\iff 2lG_{l,i} + (-(l-1)^2 + \frac{2}{(\Delta\theta)^2})F_{l-1,i} - (\frac{1}{(\Delta\theta)^2})F_{l-1,i-1} - (\frac{1}{(\Delta\theta)^2})F_{l-1,i+1} = 0 \\ &\quad \text{for all} \quad i \in \{1, \ldots, N_1 - 1\}, l \in \{1, \ldots, L-1\}. \end{aligned} \tag{3.34}$$

For (2.29), we again substitute (3.32) where appropriate and evaluate the functions at $\theta = \theta_i$. We then get

$$\begin{aligned} &2lF_{l,i} = -l^2 G_{l-1,i} - \frac{1}{(\Delta\theta)^2}(G_{l-1,i-1} - 2G_{l-1,i} + G_{l-1,i+1}) \\ &\iff 2lF_{l,i} + (l^2 - \frac{2}{(\Delta\theta)^2})G_{l-1,i} + (\frac{1}{(\Delta\theta)^2})G_{l-1,i-1} + (\frac{1}{(\Delta\theta)^2})G_{l-1,i+1} = 0 \\ &\quad \text{for all} \quad i \in \{1, \ldots, N_1 - 1\}, l \in \{1, \ldots, L-1\}. \end{aligned} \tag{3.35}$$

3.3 Finite Difference Matrix Form

We have discretized all of our conditions (2.23)-(2.29). Each of these conditions are linear combinations of the discretized points $U_{i,j}$, $F_{l,i}$ and $G_{l,i}$ for $i \in \{1,\ldots,N_1-1\}, j \in \{1,\ldots,N_2+1\}$ and $l \in \{0,\ldots,L-1\}$. Let $\mathbf{U_j}$ be the vector $[U_{1,j}, U_{2,j}, \ldots, U_{N_1-1,j}]^T$ for $j \in \{1,\ldots,N_2+1\}$. Also let $\mathbf{F_l} = [F_{l,1},\ldots,F_{l,N_1-1}]^T$ and $\mathbf{G_l} = [G_{l,1},\ldots,G_{l,N_1-1}]^T$ for $l \in \{1,\ldots,L-1\}$. We will create a vector of unknowns

$$\mathbf{V} = [\mathbf{U_1}, \mathbf{U_2}, \ldots, \mathbf{U_{N_2+1}}, \mathbf{F_0}, \mathbf{G_0}, \mathbf{F_1}, \mathbf{G_1}, \ldots, \mathbf{F_{L-1}}, \mathbf{G_{L-1}}]^T. \tag{3.36}$$

This unknowns vector corresponds to $(N_1-1)(N_2+1+2L)$ values. To have our system be determined, we need $(N_1-1)(N_2+1+2L)$ equations. We will now build a linear system of equations such that $\mathbf{AV} = \mathbf{b}$ is equivalent to our discretized system given by (3.23), (3.26), (3.27), (3.31), (3.33) - (3.35).

We will represent $\mathbf{A}$ as an $(N_2+1+2L)\times(N_2+1+2L)$ block matrix, where each block is $(N_1-1)\times(N_1-1)$. Each block will be indexed $\mathbf{A_{j,index}}$ where j will be the row of the block matrix $\mathbf{A}$. The index *index* will indicate which portion of the unknowns vector the block will be applied to.

$$index = \begin{cases} j & \text{if block is applied to } \mathbf{U_j} \\ f_l & \text{if block is applied to } \mathbf{F_l} \\ g_l & \text{if block is applied to } \mathbf{G_l} \end{cases}.$$

3.3.1 Block Matrix Form for (3.26). The first N_1-1 equations will come from (3.26). All the unknowns in (3.26) are on the left side, and on the right side are our known values. Define $\hat{U}^{(n)}_{inc,i,j}$ to be

$$\hat{U}^{(n)}_{inc,i,j} = U_{inc,i,j} + \begin{cases} \sum_{m'=1}^{m-1} (\mathcal{K}^{(n)}_{m',L}((\bar{\Omega}_m))_{i,j}, & n=1, \\ \sum_{m'=1}^{m-1} (\mathcal{K}^{(n)}_{m',L}(\bar{\Omega}_m))_{i,j} + \sum_{m'=m+1}^{M} (\mathcal{K}^{(n-1)}_{m',L}(\bar{\Omega}_m))_{i,j} & \end{cases}.$$

Essentially, $\hat{U}^{(n)}_{inc,i,j}$ is the sum of the incident wave, u_{inc}, and the other $M-1$ scattered waves approximated by $\mathcal{K}^{(n)}_{m',L}$ or $\mathcal{K}^{(n-1)}_{m',L}$ on the point $(\bar{\Omega}_m)_{i,j}$. Equation (3.26) can thus be written

as

$$\hat{\mathcal{B}}U_{i,1} = -\hat{\mathcal{B}}\hat{U}^{(n)}_{inc,i,1} \text{ for all } i \in \{1,\ldots,N_1-1\}.$$

If we expand this out, we get

$$\begin{aligned}&(\frac{Z\beta_{i,1}}{2(J\gamma)_{i,1}})U_{i-1,1}+(-\frac{3Z}{2J_{i,1}}+1-Z)U_{i,1}+(-\frac{Z\beta_{i,1}}{2(J\gamma)_{i,1}})U_{i+1,1}+(\frac{2Z}{J_{i,1}})U_{i,2}+(-\frac{Z}{2J_{i,1}})U_{i,3}\\&= -\hat{\mathcal{B}}\hat{U}^{(n)}_{inc,i,1} \text{ for all } i \in \{1,\ldots,N_1-1\}. \qquad (3.37)\end{aligned}$$

We know the quantity $\hat{\mathcal{B}}U^{(n)}_{inc,i,1}$, so this becomes the ith element of $\mathbf{b}$. In the matrix $\mathbf{A}$, we place non-zero terms where appropriate so that the dot product of the ith row of $\mathbf{A}$ and the unknowns vector $\mathbf{V}$ results in (3.37). The unknowns that are affected by these equations are $[\mathbf{U_1}, \mathbf{U_2}, \mathbf{U_3}]^T$, so there will be three $(N_1-1)\times(N_1-1)$ matrices in a row for this condition. The first matrix of the three will be $\mathbf{A_{1,1}}$.

$$\mathbf{A_{1,1}} = \begin{bmatrix} \frac{-3Z}{2J_{1,1}}+1-Z & \frac{-Z\beta_{1,1}}{2(J\gamma)_{1,1}} & \cdots & \cdots & \frac{Z\beta_{1,1}}{2(J\gamma)_{1,1}} \\ \frac{Z\beta_{2,1}}{2(J\gamma)_{2,1}} & \frac{-3Z}{2J_{2,1}}+1-Z & \frac{-Z\beta_{2,1}}{2(J\gamma)_{2,1}} & & \\ & \ddots & \ddots & \ddots & \\ & & \frac{Z\beta_{N_1-2,1}}{2(J\gamma)_{N_1-2,1}} & \frac{-3Z}{2J_{N_1-2,1}}+1-Z & \frac{-Z\beta_{N_1-2,1}}{2(J\gamma)_{N_1-2,1}} \\ \frac{-Z\beta_{N_1-1,1}}{2(J\gamma)_{N_1-1,1}} & & & \frac{Z\beta_{N_1-1,1}}{2(J\gamma)_{N_1-1,1}} & \frac{-3Z}{2J_{N_1-1,1}}+1-Z \end{bmatrix}.$$

The next block, $\mathbf{A_{1,2}}$, will be a diagonal matrix where the ith diagonal element is $\frac{2Z}{J_{i,1}}$. The last block will be $\mathbf{A_{1,3}}$ and will also be a diagonal matrix whose ith diagonal element is $\frac{-Z}{2J_{i,1}}$.

The condition (3.37) can then we written in matrix form as

$$\begin{bmatrix}\mathbf{A_{1,1}} & \mathbf{A_{1,2}} & \mathbf{A_{1,3}}\end{bmatrix}\begin{bmatrix}\mathbf{U_1}\\ \mathbf{U_2}\\ \mathbf{U_3}\end{bmatrix} = -\hat{\mathcal{B}}\hat{\mathbf{U}}^{(\mathbf{n})}_{\mathbf{inc}} \qquad (3.38)$$

where $(\hat{\mathcal{B}}\hat{\mathbf{U}}^{(\mathbf{n})}_{\mathbf{inc}})_i = \hat{\mathcal{B}}\hat{U}^{(n)}_{inc,i,1}$ for all $i \in \{1,\ldots,N_1-1\}$.

3.3.2 Block Matrix Form for (3.23). The next set of linear equations we will cover is (3.23), which is that

$$J_{i,j}\hat{\mathcal{H}}^{ep}U_{i,j} = 0 \quad \text{for} \quad 1 \le i \le N_1^m - 1 \text{ and } 2 \le j \le N_2^m.$$

For the $\eta = j$ level, this set of equations can be rewritten using block matrix multiplication with the vector $[\mathbf{U_{j-1}}, \mathbf{U_j}, \mathbf{U_{j+1}}]^T$. There will be three blocks of size $(N_1-1)\times(N_1-1)$. We will call them $\mathbf{A_{j,j-1}}, \mathbf{A_{j,j}}$ and $\mathbf{A_{j,j+1}}$ for $j \in \{2, \ldots, N_2\}$.

$$\mathbf{A_{j,j-1}} = \begin{bmatrix} \gamma_{1,j} - \frac{(\gamma_\eta)_{1,j}}{4} & \frac{\beta_{1,j}}{2} & & & & \frac{-\beta_{1,j}}{2} \\ \frac{-\beta_{2,j}}{2} & \gamma_{2,j} - \frac{(\gamma_\eta)_{2,j}}{4} & \frac{\beta_{2,j}}{2} & & & \\ & \ddots & \ddots & \ddots & & \\ & & \frac{-\beta_{N_1-2,j}}{2} & \gamma_{N_1-2,j} - \frac{(\gamma_\eta)_{N_1-2,j}}{4} & \frac{\beta_{N_1-2,j}}{2} & \\ \frac{\beta_{N_1-1,j}}{2} & & & \frac{-\beta_{N_1-1,j}}{2} & \gamma_{N_1-1,j} - \frac{(\gamma_\eta)_{N_1-1,j}}{4} \end{bmatrix}.$$

$\mathbf{A_{j,j}} =$

$$\begin{bmatrix}
\frac{J_{1,j}^2 k^2}{-2(\alpha_{1,j}+\gamma_{1,j})} & \alpha_{1,j}+\frac{(\alpha_\xi)_{1,j}}{4} & & & \alpha_{1,j}-\frac{(\alpha_\xi)_{1,j}}{4} \\
\alpha_{2,j}-\frac{(\alpha_\xi)_{2,j}}{4} & \frac{J_{2,j}^2 k^2}{-2(\alpha_{2,j}+\gamma_{2,j})} & \alpha_{2,j}+\frac{(\alpha_\xi)_{2,j}}{4} & & \\
& \ddots & \ddots & \ddots & \\
& & \alpha_{N_1-2,j}-\frac{(\alpha_\xi)_{N_1-2,j}}{4} & \frac{J_{N_1-2,j}^2 k^2}{-2(\alpha_{N_1-2,j}+\gamma_{N_1-2,j})} & \alpha_{N_1-2,j}+\frac{(\alpha_\xi)_{N_1-2,j}}{4} \\
\alpha_{N_1-1,j}+\frac{(\alpha_\xi)_{N_1-1,j}}{4} & & & \alpha_{N_1-1,j}-\frac{(\alpha_\xi)_{N_1-1,j}}{4} & \frac{J_{N_1-1,j}^2 k^2}{-2(\alpha_{N_1-1,j}+\gamma_{N_1-1,j})}
\end{bmatrix}.$$

$\mathbf{A_{j,j+1}} =$

$$\begin{bmatrix}
\gamma_{1,j}+\frac{(\gamma_\eta)_{1,j}}{4} & \frac{-\beta_{1,j}}{2} & & & \frac{\beta_{1,j}}{2} \\
\frac{\beta_{2,j}}{2} & \gamma_{2,j}+\frac{(\gamma_\eta)_{2,j}}{4} & \frac{-\beta_{2,j}}{2} & & \\
& \ddots & \ddots & \ddots & \\
& & \frac{\beta_{N_1-2,j}}{2} & \gamma_{N_1-2,j}+\frac{(\gamma_\eta)_{N_1-2,j}}{4} & \frac{-\beta_{N_1-2,j}}{2} \\
\frac{-\beta_{N_1-1,j}}{2} & & & \frac{+\beta_{N_1-1,j}}{2} & \gamma_{N_1-1,j}+\frac{(\gamma_\eta)_{N_1-1,j}}{4}
\end{bmatrix}.$$

Then, for a fixed $\eta = j$ level, the system (3.23) is

$$\begin{bmatrix} \mathbf{A_{j,j-1}} & \mathbf{A_{j,j}} & \mathbf{A_{j,j+1}} \end{bmatrix} \begin{bmatrix} \mathbf{U_{j-1}} \\ \mathbf{U_j} \\ \mathbf{U_{j+1}} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}.$$

This is true for all $j \in \{2, \ldots, N_2\}$, so the full system of equations representing (3.23) is

$$\begin{bmatrix} \mathbf{A_{2,1}} & \mathbf{A_{2,2}} & \mathbf{A_{2,3}} & & & & \\ & & & & & & \\ & \mathbf{A_{3,2}} & \mathbf{A_{3,3}} & \mathbf{A_{3,4}} & & & \\ & & & & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & & & & \\ & & & & \mathbf{A_{N_2,N_2-1}} & \mathbf{A_{N_2,N_2}} & \mathbf{A_{N_2,N_2+1}} \end{bmatrix} \begin{bmatrix} \mathbf{U_1} \\ \mathbf{U_2} \\ \mathbf{U_3} \\ \vdots \\ \vdots \\ \vdots \\ \mathbf{U_{N_2+1}} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \vdots \\ \vdots \\ \vdots \\ \mathbf{0} \end{bmatrix}, \tag{3.39}$$

which represents $(N_1-1)(N_2-1)$ equations.

3.3.3 Block Matrix Form for (3.27). The next set of equations we will put in matrix form is (3.27), which is that

$$U_{i,N_2} - H_0(kR)\sum_{l=0}^{L-1}\frac{F_{l,i}}{(kR)^l} - H_1(kR)\sum_{l=0}^{L-1}\frac{G_{l,i}}{(kR)^l} = 0$$

for all $\{1, \ldots, N_1-1\}$. Let $\mathbf{A_{N_2+1,N_2}} = \mathbf{I}$, $\mathbf{A_{N_2+1,f_l}} = \frac{-H_0(kR)}{(kR)^l}\mathbf{I}$ and $\mathbf{A_{N_2+1,g_l}} = \frac{-H_1(kR)}{(kR)^l}\mathbf{I}$ for $l \in \{0, \ldots, L-1\}$. Then

$$\begin{bmatrix} \mathbf{A_{N_2+1,N_2}} & \mathbf{A_{N_2+1,f_0}} & \mathbf{A_{N_2+1,g_0}} & \cdots & \mathbf{A_{N_2+1,f_{L-1}}} & \mathbf{A_{N_2+1,g_{L-1}}} \end{bmatrix} \begin{bmatrix} \mathbf{U_{N_2}} \\ \mathbf{F_0} \\ \mathbf{G_0} \\ \vdots \\ \mathbf{F_{L-1}} \\ \mathbf{G_{L-1}} \end{bmatrix} = \mathbf{0}. \tag{3.40}$$

3.3.4 Block Matrix Form for (3.31). Now, for (3.31), which is

$$\frac{\hat{\partial}}{\partial r}U_{i,N_2} - \frac{\partial}{\partial r}\mathcal{K}_L(R, \theta_i) = 0 \text{ for all } i \in \{1, \ldots, N_1-1\},$$

we will create blocks at the $j = N_2+2$ level. Recall that

$$\frac{\hat{\partial}}{\partial r}U_{i,N_2} = (\frac{\mu_i}{2RJ_{i,N_2}})U_{i+1,N_2} + (\frac{-\mu_i}{2RJ_{i,N_2}})U_{i-1,N_2} + (\frac{\lambda_i}{2RJ_{i,N_2}})U_{i,N_2+1} + (\frac{-\lambda_i}{2RJ_{i,N_2}})U_{i,N_2-1}$$

and

$$\frac{\partial}{\partial r}\mathcal{K}_L(R,\theta_i) = \sum_{l=0}^{L-1} A_l(kR)F_{l,i} + \sum_{l=0}^{L-1} B_l(kR)G_{l,i}$$

where $\mu, \lambda, A_l(kR)$ and $B_l(kR)$ are defined in (3.29) and (3.28).

When putting these linear equations into block matrix multiplication form, there ends up being a lot of diagonal matrices. I will cover those first.

$\mathbf{A_{N_2+2,N_2-1}}$ is a diagonal matrix with the ith diagonal element equal to $\frac{-\lambda_i}{2RJ_{i,N_2}}$.

$\mathbf{A_{N_2+2,N_2+1}}$ is a diagonal matrix with the ith diagonal element equal to $\frac{\lambda_i}{2RJ_{i,N_2}}$.

$\mathbf{A_{N_2+2,f_l}} = A_l(kR)\mathbf{I}$ for $l \in \{0,\ldots,L-1\}$.

$\mathbf{A_{N_2+2,g_l}} = B_l(kR)\mathbf{I}$ for $l \in \{0,\ldots,L-1\}$.

The only block matrix that isn't diagonal is $\mathbf{A_{N_2+2,N_2}}$, given by

$$\mathbf{A_{N_2+2,N_2}} = \begin{bmatrix} 0 & \frac{\mu_1}{2RJ_{1,N_2}} & & & \frac{-\mu_1}{2RJ_{1,N_2}} \\ \frac{-\mu_2}{2RJ_{2,N_2}} & 0 & \frac{\mu_2}{2RJ_{2,N_2}} & & \\ & \ddots & \ddots & \ddots & \\ & & \frac{-\mu_{N_2-2}}{2RJ_{N_2-2,N_2}} & 0 & \frac{\mu_{N_2-2}}{2RJ_{N_2-2,N_2}} \\ \frac{\mu_{N_2-1}}{2RJ_{N_2-1,N_2}} & & & \frac{-\mu_{N_2-1}}{2RJ_{N_2-1,N_2}} & 0 \end{bmatrix}. \tag{3.41}$$

The full matrix form for (3.31) is then

$$\begin{bmatrix} \mathbf{A_{N_2+2,N_2-1}} & \mathbf{A_{N_2+2,N_2}} & \mathbf{A_{N_2+2,N_2+1}} & \mathbf{A_{N_2+2,f_0}} & \cdots & \mathbf{A_{N_2+2,g_{L-1}}} \end{bmatrix} \begin{bmatrix} \mathbf{U_{N_2-1}} \\ \mathbf{U_{N_2}} \\ \mathbf{U_{N_2+1}} \\ \mathbf{F_0} \\ \cdots \\ \mathbf{G_{L-1}} \end{bmatrix} = \mathbf{0}. \tag{3.42}$$

3.3.5 Block Matrix From for (3.33). We will now get the matrix form for (3.33). We get

$$\mathbf{A_{N_2+3,f_{L-1}}} =$$

$$H_0(kR)\begin{bmatrix} (L-1)^2-\frac{2}{(\Delta\theta)^2} & \frac{1}{(\Delta\theta)^2} & & & \frac{1}{(\Delta\theta)^2} \\ \frac{1}{(\Delta\theta)^2} & (L-1)^2-\frac{2}{(\Delta\theta)^2} & \frac{1}{(\Delta\theta)^2} & & \\ & \ddots & \ddots & \ddots & \\ & & \frac{1}{(\Delta\theta)^2} & (L-1)^2-\frac{2}{(\Delta\theta)^2} & \frac{1}{(\Delta\theta)^2} \\ \frac{1}{(\Delta\theta)^2} & & & \frac{1}{(\Delta\theta)^2} & (L-1)^2-\frac{2}{(\Delta\theta)^2} \end{bmatrix}. \tag{3.43}$$

and

$$\mathbf{A_{N_2+3,g_{L-1}}} = H_1(kR)\begin{bmatrix} L^2-\frac{2}{(\Delta\theta)^2} & \frac{1}{(\Delta\theta)^2} & & & \frac{1}{(\Delta\theta)^2} \\ \frac{1}{(\Delta\theta)^2} & L^2-\frac{2}{(\Delta\theta)^2} & \frac{1}{(\Delta\theta)^2} & & \\ & \ddots & \ddots & \ddots & \\ & & \frac{1}{(\Delta\theta)^2} & L^2-\frac{2}{(\Delta\theta)^2} & \frac{1}{(\Delta\theta)^2} \\ \frac{1}{(\Delta\theta)^2} & & & \frac{1}{(\Delta\theta)^2} & L^2-\frac{2}{(\Delta\theta)^2} \end{bmatrix}. \tag{3.44}$$

Thus the matrix form of (3.33) is

$$\begin{bmatrix} \mathbf{A_{N_2+3,f_{L-1}}} & \mathbf{A_{N_2+3,g_{L-1}}} \end{bmatrix} \begin{bmatrix} \mathbf{F_{L-1}} \\ \mathbf{G_{L-1}} \end{bmatrix} = \mathbf{0}. \tag{3.45}$$

3.3.6 Block Matrix Form for (3.34) and (3.35). We will first do the block matrix form for (3.34). For a given $l \in \{1, \ldots, L-1\}$ and a given $i \in \{1, \ldots, N_1-1\}$, we have that

$$2lG_{l,i}+(-(l-1)^2+\frac{2}{(\Delta\theta)^2})F_{l-1,i}-(\frac{1}{(\Delta\theta)^2})F_{l-1,i-1}-(\frac{1}{(\Delta\theta)^2})F_{l-1,i+1}=0.$$

The blocks that will make this condition will be at the $j = N_2+3+2l$ level in the block matrix $\mathbf{A}$.

The matrix $\mathbf{A_{N_2+3+2l,g_l}} = 2l\mathbf{I}$. We also have the matrix $\mathbf{A_{N_2+3+2l,f_{l-1}}}$. This is more complicated, and is given by

$$\mathbf{A_{N_2+3+2l,f_{l-1}}} = \tag{3.46}$$

$$\begin{bmatrix} \frac{2}{(\Delta\theta)^2}-(l-1)^2 & \frac{-1}{(\Delta\theta)^2} & & & \frac{-1}{(\Delta\theta)^2} \\ \frac{-1}{(\Delta\theta)^2} & \frac{2}{(\Delta\theta)^2}-(l-1)^2 & \frac{-1}{(\Delta\theta)^2} & & \\ & \ddots & \ddots & \ddots & \\ & & \frac{-1}{(\Delta\theta)^2} & \frac{2}{(\Delta\theta)^2}-(l-1)^2 & \frac{-1}{(\Delta\theta)^2} \\ \frac{-1}{(\Delta\theta)^2} & & & \frac{-1}{(\Delta\theta)^2} & \frac{2}{(\Delta\theta)^2}-(l-1)^2 \end{bmatrix}. \tag{3.47}$$

The system of equations (3.34) is then given by

$$\begin{bmatrix} \mathbf{A_{N_2+3+2l,f_{l-1}}} & \mathbf{0} & \mathbf{0} & \mathbf{A_{N_2+3+2l,g_l}} \end{bmatrix} \begin{bmatrix} \mathbf{F_{l-1}} \\ \mathbf{G_{l-1}} \\ \mathbf{F_l} \\ \mathbf{G_l} \end{bmatrix} = \mathbf{0}. \tag{3.48}$$

We will now do the block matrix form for (3.35) For a given $l \in \{1, \ldots, L-1\}$ and a given $i \in \{1, \ldots, N_1-1\}$, we have that

$$2lF_{l,i}+(l^2-\frac{2}{(\Delta\theta)^2})G_{l-1,i}+(\frac{1}{(\Delta\theta)^2})G_{l-1,i-1}+(\frac{1}{(\Delta\theta)^2})G_{l-1,i+1}=0.$$

The blocks that will represent this condition will be at the $j = N_2+2+2l$ level in the block

matrix $\mathbf{A}$.

The matrix $\mathbf{A_{N_2+2+2l,f_l}} = 2l\mathbf{I}$. We also have the matrix $\mathbf{A_{N_2+2+2l,g_{l-1}}}$. This is more complicated, and is given by

$$\mathbf{A_{N_2+2+2l,g_{l-1}}} = \begin{bmatrix} l^2-\frac{2}{(\Delta\theta)^2} & \frac{1}{(\Delta\theta)^2} & & & \frac{1}{(\Delta\theta)^2} \\ \frac{1}{(\Delta\theta)^2} & l^2-\frac{2}{(\Delta\theta)^2} & \frac{1}{(\Delta\theta)^2} & & \\ & \ddots & \ddots & \ddots & \\ & & \frac{1}{(\Delta\theta)^2} & l^2-\frac{2}{(\Delta\theta)^2} & \frac{1}{(\Delta\theta)^2} \\ \frac{1}{(\Delta\theta)^2} & & & \frac{1}{(\Delta\theta)^2} & l^2-\frac{2}{(\Delta\theta)^2} \end{bmatrix}.$$

The system of equations (3.34) is then given by

$$\begin{bmatrix} \mathbf{A_{N_2+2+2l,g_{l-1}}} & \mathbf{A_{N_2+2+2l,f_l}} \end{bmatrix} \begin{bmatrix} \mathbf{G_{l-1}} \\ \mathbf{F_l} \end{bmatrix} = \mathbf{0}.$$

A full implementation of (3.34) and (3.35) for all $i \in \{1, \ldots, N_1-1\}$ and $l \in \{1, \ldots, L-1\}$ is then

$$\begin{bmatrix} \mathbf{0} & \mathbf{A_{N_2+4,g_0}} & \mathbf{A_{N_2+4,f_1}} & \mathbf{0} & & & \\ & & & & & & \\ \mathbf{A_{N_2+5,f_0}} & \mathbf{0} & \mathbf{0} & \mathbf{A_{N_2+5,g_1}} & & & \\ & \ddots & \ddots & \ddots & \ddots & & \\ & & \mathbf{0} & \mathbf{A_{N_2+2L,g_{L-2}}} & \mathbf{A_{N_2+2L,f_{L-1}}} & \mathbf{0} & \\ & & \mathbf{A_{N_2+1+2L,f_{L-2}}} & \mathbf{0} & \mathbf{0} & \mathbf{A_{N_2+1+2L,g_{L-1}}} & \end{bmatrix} \begin{bmatrix} \mathbf{F_0} \\ \mathbf{G_0} \\ \mathbf{F_1} \\ \mathbf{G_1} \\ \vdots \\ \mathbf{F_{L-2}} \\ \mathbf{G_{L-2}} \\ \mathbf{F_{L-1}} \\ \mathbf{G_{L-1}} \end{bmatrix} = \mathbf{0}. \tag{3.49}$$

3.3.7 The Full IBVP Representation. We have gone through each of the second order discretized equations for our iterative BVP for a given obstacle m and iteration (n) and have written each of the systems of equations in block matrix form. $\mathbf{A}$ is a $(N_2+1+2L)\times(N_2+1+2L)$ block matrix where each block is a $(N_1-1)\times(N_1-1)$ matrix. We have described each block in the previous sections using indexing. We have also described the forcing vector $\mathbf{b}$. In summary, the vector $\mathbf{b}$ is a vector of all zeros, except for the first N_1-1 values which contain the vector $-\hat{\mathcal{B}}\hat{\mathbf{U}}^{(\mathbf{n})}_{\mathbf{inc}}$ All that we would need to do is then solve $\mathbf{AV}=\mathbf{b}$ using a linear solver of choice.

We will go back to using notation to represent the obstacle m and iteration (n). We will denote the matrix for obstacle m as $\mathbf{A_m}$, the unknowns vector for obstacle m and iteration (n) as $\mathbf{V}^{(\mathbf{n})}_{\mathbf{m}}$, and the forcing vector as $\mathbf{b}^{(\mathbf{n})}_{\mathbf{m}}$. The entire iterative process can be summed up by

$$\begin{aligned} &\text{Solve } \mathbf{A_m}\mathbf{V}^{(\mathbf{n})}_{\mathbf{m}} = \mathbf{b}^{(\mathbf{n})}_{\mathbf{m}} \text{ for } \mathbf{V}^{(\mathbf{n})}_{\mathbf{m}} \\ &\text{for } n = 1, \dots \text{ and } m = 1, \dots, M. \end{aligned} \tag{3.50}$$

Note that $\mathbf{A_m}$ doesn't have a superscript (n). This is because $\mathbf{A_m}$ is independent of the iteration process. The moving parts of the iteration are the unknowns vector $\mathbf{V_m^{(n)}}$ and the forcing vector $\mathbf{b_m^{(n)}}$. This means that the matrix $\mathbf{A_m}$ needs to be generated only once at the beginning of the iteration process.

3.4 Numerical Implementation

For our algorithm, the following are parameters that are needed to start the process.

- *centers*, a list of 2-D vectors $\vec{b}_m$ which contain the centers of our M obstacles.
- $IntxFun$, a cell array of functions that define $B_x^m(\theta^m)$.
- $IntyFun$, a cell array of functions that define $B_y^m(\theta^m)$.
- ϕ, the angle of the incident wave.
- k, the wavenumber of the incident wave.
- Z, the acoustic hardness of the M obstacles.
- PPW, points per wavelength, a measure of how refined the grids are.
- L, how many terms to use in KFE.
- Rs, a vector of outer radii R^m, where we place the absorbing boundary condition.
- Tol, part of our stopping criteria.
- *maxiter*, part of our stopping criteria.

With these variables initialized, we then are ready to begin the algorithm.

(i) Calculate N_1^m and N_2^m using (3.8). Using N_1^m and N_2^m and the interior boundary functions B_x^m and B_y^m and the outer radii $Rs(m) = R^m$, generate the computational grid $\bar{\Omega}_m^-$ using the methods described in section (3.2.1), and then get the various representations of $\bar{\Omega}_m^-$ using section (3.2.2).

(ii) Build A_m for each obstacle using section (3.3). While generating the matrix A_m, also save the values of γ, β and J on the obstacle boundary. We can call these $\gamma_{b,m}$, $\beta_{b,m}$, and $J_{b,m}$.

(iii) Create cell arrays F, G, $newF$, $newG$, and initialize $F\{m,l\}$ to be a zero $(N_1^m-1)\times 1$ vector for all $m \in \{1,\ldots,M\}$ and $l \in \{0,\ldots,L-1\}$. Repeat for G, $newF$ and $newG$.

(iv) Create a cell array U and $newU$, and initialize $U\{m\}$ and $newU\{m\}$ to be a $(N_1-1)\times N_2$ zero array for all $m \in \{1,\ldots,M\}$.

With these definitions of variables, I will now present the iterative process in Algorithm 1.

3.4.1 Speeding Up The Iterative Process. Now that the iterative process is outlined in Algorithm 1, there are a few things that can be done to speed up the iteration process and make it so the algorithm can compete with other algorithms out there.

Parallelization. There are a few processes that can be done in parallel that can really speed up the algorithm. The first few things that can be done in parallel are items (1) - (4) from the list in the previous section. What is significant to note is that each grid Ω_m^- and each matrix A_m can be created in parallel. This is a huge time saver. There is also parallelization that can be done within the iteration process outlined in Algorithm 1, which will save a lot of time. For Gauss-Seidel iteration (the iteration outlined in Algorithm 1), we can replace the for-loop on line 9 with a parallelized for-loop. For a given obstacle m, the parallelized for-loop will sum up the scattered waves of the other $M-1$ obstacles in parallel. If we are doing Jacobi iteration, then we can instead parallelize the for-loop on line 7. In this case, we are doing each obstacle in parallel.

Initializing The Scattered Waves. In item (3) from the list at the beginning of the section, we initialize all of the $F\{m,l\}$ and $G\{m,l\}$ to be a zero vector. Since the scattered wave from obstacle m outside Ω_m^- is given by (3.18), this is essentially initializing the iteration with no scattered wave from any of the obstacles.

Algorithm 1 The Gauss-Seidel Iterative Process

1: $n \leftarrow 0$
2: **do**
3: $n \leftarrow n+1$
4: $F \leftarrow newF$
5: $G \leftarrow newG$
6: $U \leftarrow newU$
7: **for** $m = 1 : M$ **do**
8: $\hat{U}_{inc} \leftarrow \mathbf{0}((N_1^m - 1) \times 3)$ ▷ This is a zero matrix with $N_1^m - 1$ rows and 3 columns
9: **for** $m' \neq m$ **do**
10: **for** $l = 0 : L-1$ **do**
11: $\tilde{F}_{m',l} \leftarrow$ Interpolating function based on $newF\{m, l\}$
12: $\tilde{G}_{m',l} \leftarrow$ Interpolating function based on $newG\{m, l\}$ ▷ To change to Jacobi iteration, base the interpolating functions off of F and G instead of $newF$ and $newG$
13: $f\_interps \leftarrow \tilde{F}_{m',l}(\Theta^{(m',m)}[:, 1:3])$
14: $g\_interps \leftarrow \tilde{G}_{m',l}(\Theta^{(m',m)}[:, 1:3])$
▷ We interpolate the scattered wave at the first 3 η levels of Ω_m^-. We do the first 3 levels so that we can do one-sided second order finite difference(line 20).
15: $\hat{U}_{inc} \leftarrow \hat{U}_{inc} + H_0(kR^{(m,m')}) \frac{f\_interps}{(kR^{(m,m')})^l} + H_1(kR^{(m,m')}) \frac{g\_interps}{(kR^{(m,m')})^l}$
16: **end for**
17: **end for**
18: $\hat{U}_{inc} \leftarrow \hat{U}_{inc} + u_{inc}((\bar{\Omega}_m^-)[:, 1:3])$
19: $b \leftarrow \mathbf{0}((N_1^m - 1)(N_2^m + 1 + 2L) \times 1)$
20: $b[1 : N_1^m - 1] \leftarrow \hat{\mathcal{B}}\hat{U}_{inc}[1 : N_1^m - 1, 1]$
21: $V \leftarrow \text{LinearSolve}(A_m, b)$
22: $V \leftarrow \text{reshape}(V, N_1^m - 1, N_2^m + 1 + 2L)$
23: $newU\{m\} \leftarrow (V[:, 1 : N_2^m])$
24: **for** $l = 0 : L-1$ **do**
25: $newF\{m, l\} \leftarrow V[:, N_2^m + 2 + 2l]$
26: $newG\{m, l\} \leftarrow V[:, N_2^m + 3 + 2l]$
27: **end for**
28: **end for**
29: **while** $\max_m |newUm - Um|_\infty > Tol \ \ \& \ \ n < maxiter$

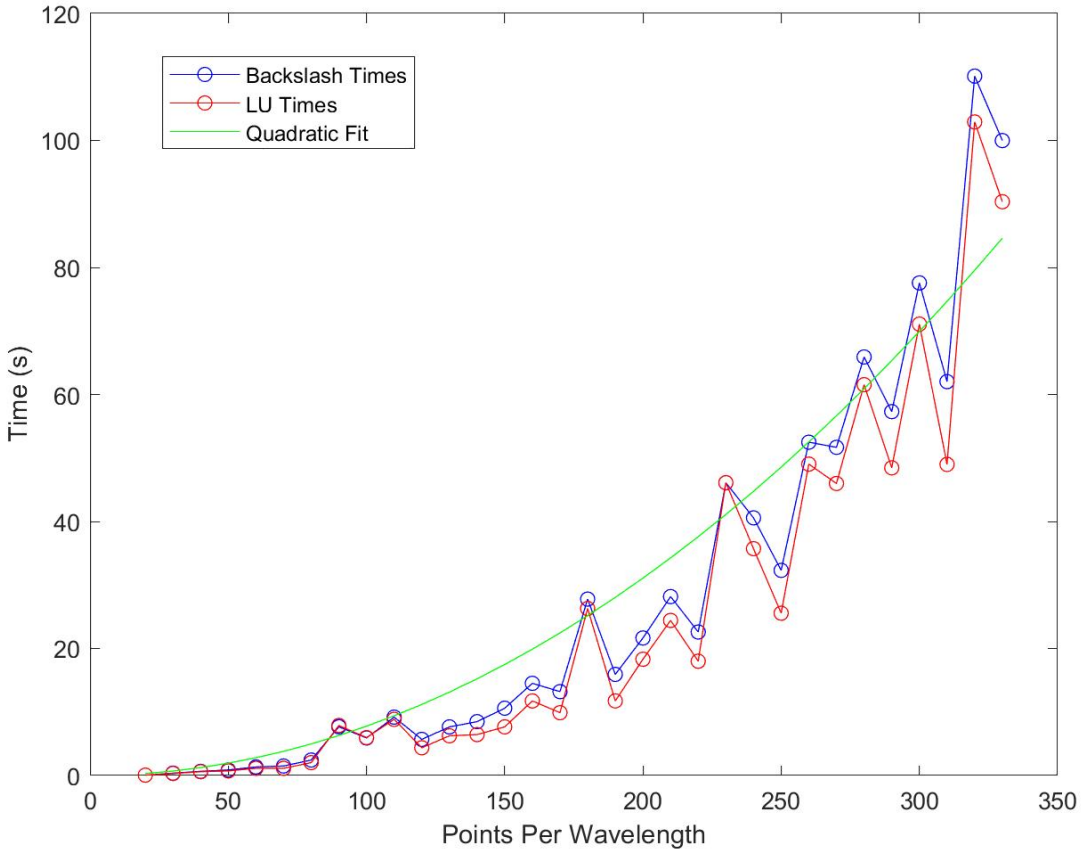
We can instead initialize the scattered wave for each obstacle to be a decent approximation of the scattered wave. This means we would be starting the iteration process at a point that is closer to the true solution. This would be like skipping a few iterations in the iteration process.

We could first run the algorithm for a small PPW that won't take much time to complete. For the BYU configuration, using $k = 2\pi$, $Tol =$ 1E-8 and $PPW = 25$, the entire algorithm takes less than 10 seconds to converge. Once the algorithm is finished, we have a decent approximation for the scattered waves of each obstacle with a typical L2 error on the order of 10E-3. This means that we have the vectors $F\{m, l\}$ and $G\{m, l\}$ for all $m \in \{1, \ldots, M\}$ and $l \in \{0, \ldots, L-1\}$ for this less-refined grid. We can base interpolating functions off these less-refined vectors from the $PPW = 25$ solution, and then initialize the vectors $F\{m, l\}$ and $G\{m, l\}$ by evaluating the interpolating function at the discretized θ values for $PPW = 200$.

For the BYU configuration (see Figure 4.2a), the algorithm took a total of 29 iterations to converge under a 1E-8 tolerance. However, if I initialize it with the $PPW = 25$ solution, then it only took 17 iterations.

Factoring The Matrices. The most computationally costly part of the iteration process is solving the matrix system $\mathbf{A_m V_m^{(n)}} = \mathbf{b_m^{(n)}}$. This is done on line 21 in Algorithm 1. The matrix $\mathbf{A_m}$ is a $(N_1^m-1)(N_2^m+1+2L)\times(N_1^m-1)(N_2^m+1+2L)$ sparse matrix with $(N_1^m-1)(9N_2^m+12L-1)$ non-zero terms. A typical sparse linear solver should theoretically have a time complexity of $O(nnz)$ where nnz is the number of non-zero terms in the sparse matrix. We then should expect the time complexity to solve our system to be proportional to $nnz = (N_1^m-1)(9N_2^m+12L-1)$. Substituting the definitions (3.8) and isolating the leading order term, we have that the number of non-zero terms is asymptotic to $9k^2PPW^2((R^m)^2-(r_{min}^m)^2)/(4\pi)$. Thus, for a fixed R^m and r_{min}^m, $nnz \in O(PPW^2)$. We then expect the temporal complexity of solving $\mathbf{A_m V_m^{(n)}} = \mathbf{b_m^{(n)}}$ to grow proportional to PPW^2. Looking at Figure 3.4, this does seem to be the case, at least for the range of points per wavelength we are dealing with.

Figure 3.4: Times to run backslash and LU decomposition for a matrix $\mathbf{A}$, averaged over 6 trials. We used $R = 2$ and $r_0 = 1$. The quadratic fit gives $t = 0.0007769 * PPW^2$ as the best fit.



When I first began coding this algorithm, I was using Matlab's backslash operator to solve the systems. Looking at figure 3.4, we can see that solving this system gets more and more time consuming as we increase the points per wavelength. Fitting the data I got for time to a quadratic function, the leading order term is $0.0007769 PPW^2$. It can easily get to a point where solving this single system can take around 5 minutes. If the configuration takes N iterations to converge within a specified tolerance, and there are M obstacles, then the algorithm will have to solve $N * M$ of these systems sequentially. If solving a single system takes 5 minutes, then we expect convergence after more than $\frac{N*M}{12}$ hours of computational time. For 7 obstacles and 30 iterations, as an example, convergence would take over 17 hours. Luckily, we have devised a way to get around this.

The way I have solved this is to do an LU factorization for each of the matrices $\mathbf{A_m}$ with full column and row pivoting before I enter into the do-while loop on line 2 of Algorithm 1. So, before anything else, we will find the matrices $[\bar{\mathbf{L}}_\mathbf{m}, \bar{\mathbf{U}}_\mathbf{m}, \bar{\mathbf{P}}_\mathbf{m}, \bar{\mathbf{Q}}_\mathbf{m}, \bar{\mathbf{D}}_\mathbf{m}]$ such that

$$\bar{\mathbf{P}}_\mathbf{m}((\bar{\mathbf{D}}_\mathbf{m})^{-1}\mathbf{A_m})\bar{\mathbf{Q}}_\mathbf{m} = \bar{\mathbf{L}}_\mathbf{m}\bar{\mathbf{U}}_\mathbf{m} \tag{3.51}$$

where $\bar{\mathbf{P}}_\mathbf{m}$ and $\bar{\mathbf{Q}}_\mathbf{m}$ are row and column permutation matrices, respectfully, and $\bar{\mathbf{D}}_\mathbf{m}$ is a diagonal scaling matrix. This decomposition is very efficiently found for sparse matrices using Matlab's "lu" function.

Once we have the decomposition of each matrix $\mathbf{A_m}$, we can then efficiently solve the system (3.50) like so:

$$\begin{aligned} &\text{Solve } \bar{\mathbf{L}}_\mathbf{m}\mathbf{y_1} = \bar{\mathbf{P}}_\mathbf{m}(\bar{\mathbf{D}}_\mathbf{m})^{-1}\mathbf{b_m^{(n)}}) \text{ for } \mathbf{y_1} \text{ using forward substitution} \\ &\text{Solve } \bar{\mathbf{U}}_\mathbf{m}\mathbf{y_2} = \mathbf{y_1} \text{ for } \mathbf{y_2} \text{ using backward substitution} \\ &\mathbf{V_m^{(n)}} = \mathbf{Q_m}\mathbf{y_2}. \end{aligned} \tag{3.52}$$

What allows us to take advantage of this factorization is that the solution to the system (3.50) is only dependent on the forcing vector $b_m^{(n)}$. So, for the obstacle m, we end up solving

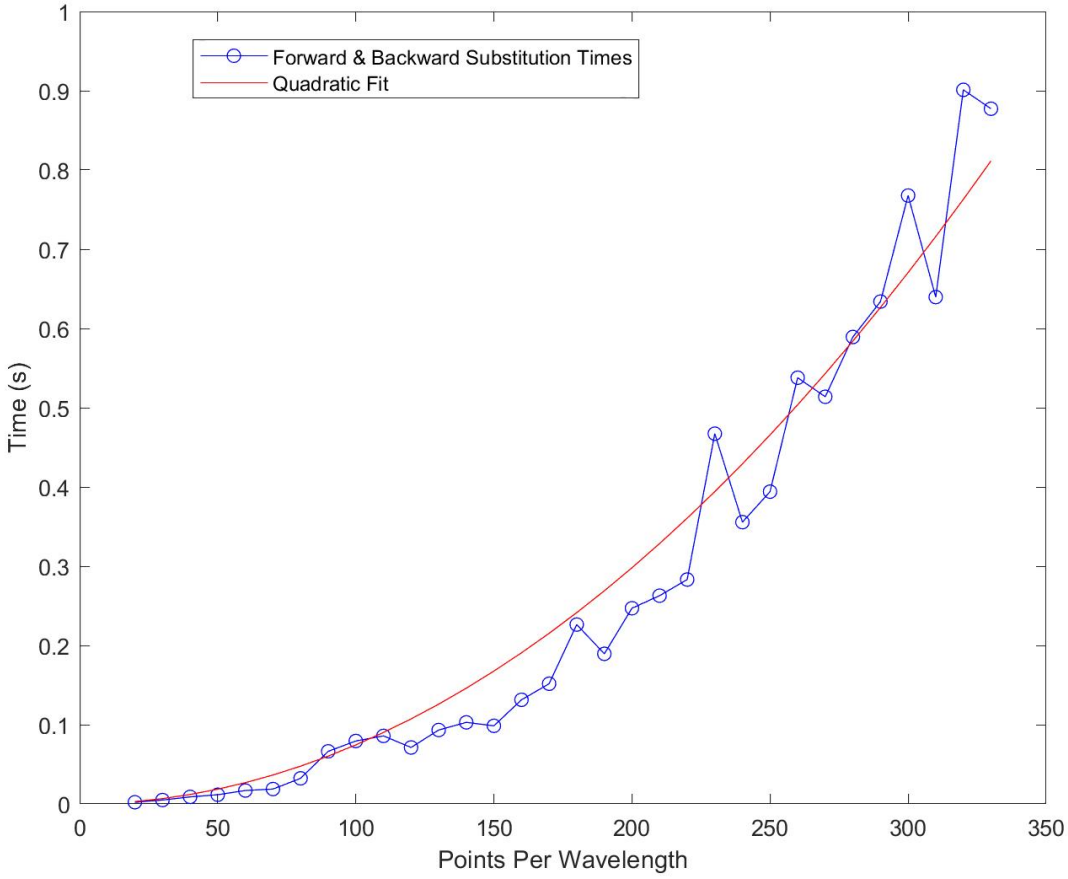
$$\mathbf{A_m}\mathbf{V_m^{(1)}} = \mathbf{b_m^{(1)}}, \mathbf{A_m}\mathbf{V_m^{(2)}} = \mathbf{b_m^{(2)}}, \mathbf{A_m}\mathbf{V_m^{(3)}} = \mathbf{b_m^{(3)}}, \ldots$$

until convergence is reached. Since the matrix $\mathbf{A_m}$ doesn't change, then substituting the LU factorization of $\mathbf{A_m}$ doesn't change the linear system, but it makes it easier to solve using forward and backward substitution.

Notice from Figure 3.4 that the times for an LU decomposition are extremely close to the times for a backslash solution. This is because Matlab calls the LU decomposition within backslash, and then does a similar forward and backward substitution like the one I've described in (3.52). The times are so close because the time it takes to do the forward and backward substitution is very small compared to the time it takes to decompose the matrix into it's LU factors. To see some numerical results on forward and backward substitution times, look at Figure 3.5. Up near the high points per wavelength value of $PPW = 320$, it still takes less than a second to do the forward and backward substitution. The average ratio

of time to forward-backward solve to time to LU factor is 0.0135, with standard deviation 0.0049. So, a decent upper bound for our ratio is 0.02. This means that we can forward-backward solve in 2% of the time it takes to do an LU factorization.

Figure 3.5: Times for forward and backward substitution solving the system (3.50) using the LU factors, averaged over 6 trials. We used $R = 2$ and $r_0 = 1$. The quadratic fit gives $t = 7.451{*}10^{-6}{*}PPW^2$ as the best fit.



When solving the systems repeatedly using Matlab's backslash operator on our sparse matrix, the operator automatically chooses the LU solver to solve the system. So, every time we call backslash on the system $\mathbf{A_m V_m^{(n)}} = \mathbf{b_m^{(n)}}$, Matlab does an LU decomposition on $\mathbf{A_m}$ anyway and then does forward and backward substitution to solve the system. If we look at it that way, then when we iterate using the backslash operator, we are doing $N{*}M$ sequential LU factorizations and $N{*}M$ forward and backward substitutions. If we

instead do the LU decomposition ourselves before the do-while loop, then we only do M sequential LU factorizations and $N*M$ sequential forward and backward substitutions. We then have successfully eliminated $(N-1)*M$ costly LU factorizations from our iteration process. What's even more amazing, is that we can do the M LU factorizations in parallel before the do-while loop, so we could potentially do all M LU factorizations in the time it takes to do one LU factorization, if you have enough processors. So, we've gone from doing $N*M$ LU factorizations and forward and backward substitutions sequentially, to doing M LU factorizations in parallel and then $N*M$ forward and backward substitutions sequentially.

The example we used before was 5 minutes per backslash, over $N = 30$ iterations and $M = 7$ obstacles. Our estimate for the time to converge was $\frac{N*M}{12} = 17.5$ hours. If instead we do M LU factorizations in parallel, and then $N*M$ forward and backward substitutions sequentially, our estimate would be $\frac{1}{12} + \frac{30*7}{12} * 0.02 = 0.433$ hours. Thus, we have reduced our estimated 17 hours of computational time to less than 30 minutes.

For the 3 cylinders configuration, here are the times where we use backslash vs. LU factorization. The ratio row shows the LU method is around 10% of the backslash method. This is a significant time saver (3.1).

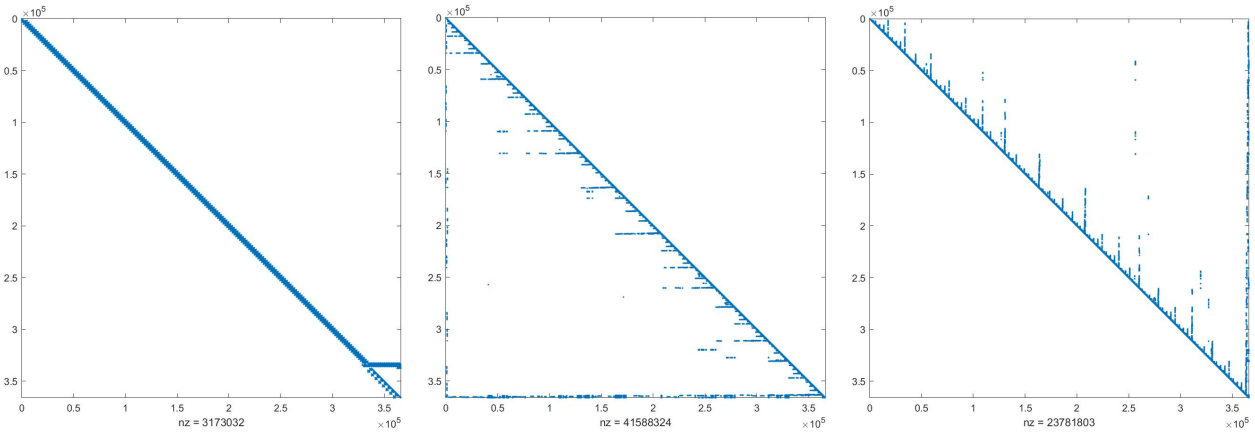
Table 3.1: Comparison of convergence times for the 3 cylinders configuration. Entries are in seconds.

PPW	80	100	150	180	250
LU Factoring	10.1	17.0	33.0	48.8	262.4
Backslash	105.9	214.5	403.6	652.6	2410.6
Ratio	0.095	0.0792	0.082	0.0748	0.109

One downside to the LU decomposition is that the memory requirements grow rapidly. Though the factors $\bar{\mathbf{L}}_\mathbf{m}$ and $\bar{\mathbf{U}}_\mathbf{m}$ are sparse themselves, they are much less sparse than the matrix $\mathbf{A_m}$, as seen in Figure 3.6. We know that the number of non-zero terms of $\mathbf{A_m}$ is asymptotic to $9k^2PPW^2((R^m)^2-(r^m_{min})^2)/(4\pi)$. What we don't know is how this reflects in the non-zero terms of the LU factors. Plotted in Figure 3.7 are the number of non-zero elements in various matrices as a function of PPW. We can see that the number of non-zero

terms of the LU factors grows much faster than the number of non-zero factors of $\mathbf{A}$. An LU decomposition of a $PPW = 600, L = 9, R = 2, r_{min} = 1$ matrix $\mathbf{A}$ exceeds 32 gigabytes of memory, so a normal laptop with 16-32 gigabytes of memory doesn't have enough resources to run this algorithm for such high PPW. This algorithm will run for higher PPW, it just needs to be run on a computer that has access to more memory.

Figure 3.6: Sparse Matrix Plots for $PPW = 150, L = 9, R = 2, r_{min} = 1, k = 2\pi$.



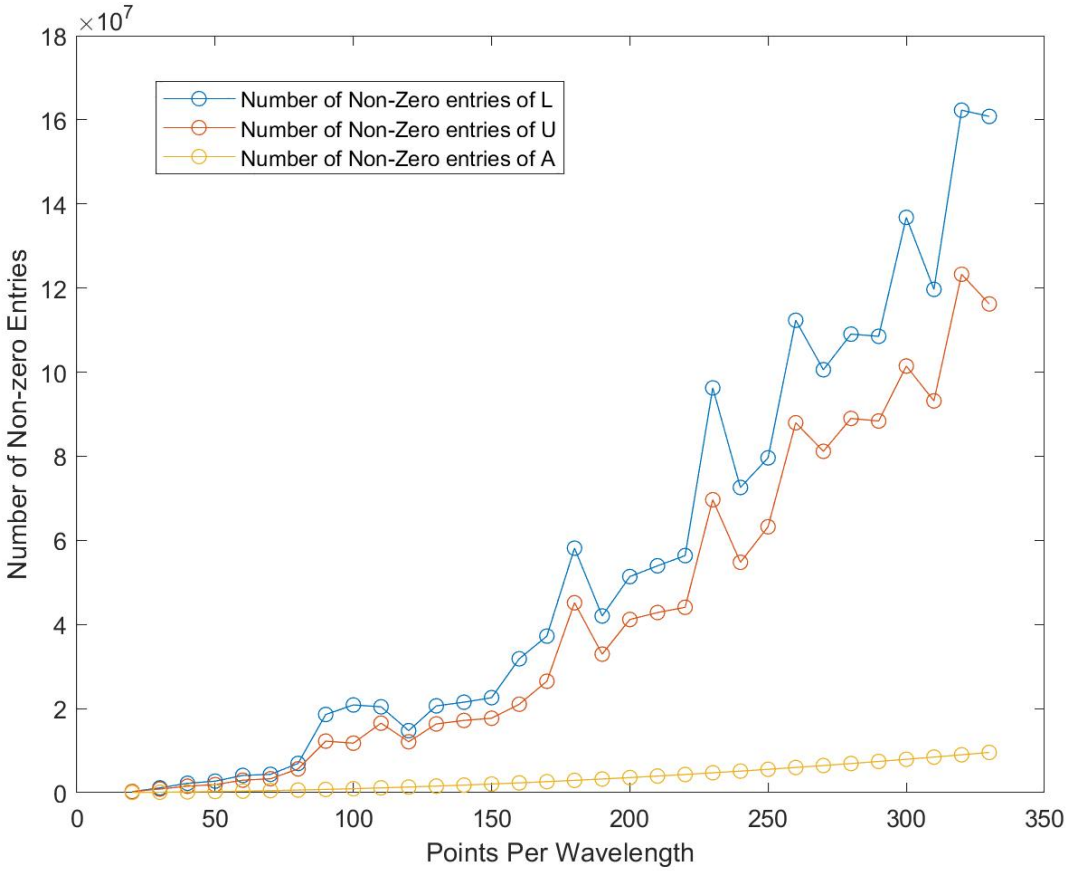
(a) The matrix $\mathbf{A}$. (b) The $\mathbf{L}$ factor of $\mathbf{A}$. (c) The $\mathbf{U}$ factor of $\mathbf{A}$.

Chapter 4. Numerical Results

We have chosen a few configurations to run experiments on to show the convergence of our algorithm. The first configuration is 3 cylinders of radius 1 lined up on the x-axis. The second is 6 cylinders of various sizes aligned in a "Y" shape. The last is 3 non-cylindrical obstacles arranged off-axis.

When the algorithm is run for a certain PPW, we get as output the scattered wave approximation U^m on each of the points in $\bar{\Omega}_m^-$ for each obstacle m. For our analysis in this chapter, we will use the exact solution given by (1.5) to obtain the true values of u on the discretized points on each obstacle's artificial boundaries, and we will store these true values in a vector u_b^m, where $(u_b^m)_i = u(R^m, \theta_i^m)$ for $i \in \{1, \ldots, N_1^m - 1\}$. We can then take the L2-norm error for each artificial boundary, $E_b^m = ||U^m[:, N_2^m] - u_b^m||_2$. We then define the L2

Figure 3.7: Plot of number of non-zero elements for the matrix $\mathbf{A}$ and it's LU factors, with $L = 9, R = 2, r_{min} = 1, k = 2\pi$.



boundary error to be

$$E_b = \max_{m \in \{1,\ldots,M\}} E_b^m. \tag{4.1}$$

We also get as output the approximations of the farfield family of functions $F_{m,l}$ and $G_{m,l}$ at the discretized θ_i^m values. We can numerically approximate the farfield pattern of the system at a global angle θ as such:

$$\hat{u}_\infty(\theta) = \frac{1-i}{\sqrt{\pi}} \sum_{m=1}^{M} (\tilde{F}_0^m(\theta) - i\tilde{G}_0^m(\theta)) e^{ik(b_x^m \cos\theta + b_y^m \sin\theta)} \tag{4.2}$$

where $\tilde{F}_0^m$ and $\tilde{G}_0^m$ are the interpolating functions based off of the discretized value approximations of $F_{m,0}$ and $G_{m,0}$. We then choose a number of points $Nfar$ to sample $\hat{u}_\infty$ at $Nfar$

equally spaced points between $[0, 2\pi]$. Call the resulting vector of values U_∞. We also get the exact solution of the farfield pattern using (1.6) at those same equally spaced points, and we will call this vector $u_{\infty,true}$. We can then take the L2-norm error between the two.

$$E_{ffp} = ||U_\infty - u_{\infty,true}||_2. \tag{4.3}$$

The two measures of error we will then use in our analysis of convergence is the artificial boundary error E_b and the farfield pattern error E_{ffp}.

4.1 3 Cylinders on the x-axis

An example computational grid of $PPW = 6$ is given for this configuration in Figure 4.1a.

For this configuration, we will set $\phi = -\pi/2$, $k = 2\pi$, $L = 10$ and $Tol = 10^{-10}$. The cylinders are all of radius 1 and artificial boundary of radius 2, and they are centered at $[-4.25, 0]^T$, $[0, 0]$ and $[4.25, 0]^T$. We will show that the algorithm converges with order 2 for $Z \in \{0, 0.5, 1\}$, which correspond to a Dirichlet, Robin, and Neumann problem. The PPW chosen for this problem were $PPW \in \{100, 200, 300, 400, 500\}$. The convergence analysis for the errors E_b and E_{ffp} are found in tables 4.1 - 4.3.

Table 4.1: Relative L2 Error Table for 3 Cylinders Configuration, $Z = 0$

PPW	h	Farfield Pattern Error	Order	Boundary Error	Order
100	0.01000	2.68E-4	-	2.87E-4	-
200	0.00500	6.65E-5	2.01001	7.12E-5	2.01188
300	0.00333	2.95E-5	2.00699	3.15E-5	2.00848
400	0.00250	1.66E-5	2.00537	1.77E-5	2.00596
500	0.00200	1.06E-5	2.00633	1.13E-5	2.00510

Looking at the results, it is evident that the solutions are converging with order 2 for all values of $Z \in \{0, 0.5, 1\}$. The accuracy, however, is different for the 3 Z values. The Dirichlet problem has the best accuracy, with both the farfield pattern and boundary L2 errors getting 5 decimals of accuracy past $PPW = 200$. The next best accuracy is given by the Neumann problem, with 5 decimals of accuracy past $PPW = 300$. For the Robin

Table 4.2: Relative L2 Error Table for 3 Cylinders Configuration, $Z = 0.5$

PPW	h	Farfield Pattern Error	Order	Boundary Error	Order
100	0.01000	7.36E-4	-	8.74E-4	-
200	0.00500	1.85E-4	1.99300	2.19E-4	1.99413
300	0.00333	8.23E-5	1.99435	9.76E-5	1.99687
400	0.00250	4.64E-5	1.99336	5.50E-5	1.99750
500	0.00200	2.97E-5	1.99219	3.52E-5	1.99862

Table 4.3: Relative L2 Error Table for 3 Cylinders Configuration, $Z = 1$

PPW	h	Farfield Pattern Error	Order	Boundary Error	Order
100	0.01000	4.96E-4	-	6.61E-4	-
200	0.00500	1.23E-4	2.00827	1.65E-4	2.00299
300	0.00333	5.48E-5	2.00236	7.33E-5	2.00067
400	0.00250	3.08E-5	2.00032	4.12E-5	1.99974
500	0.00200	1.97E-5	1.99865	2.64E-5	1.99903

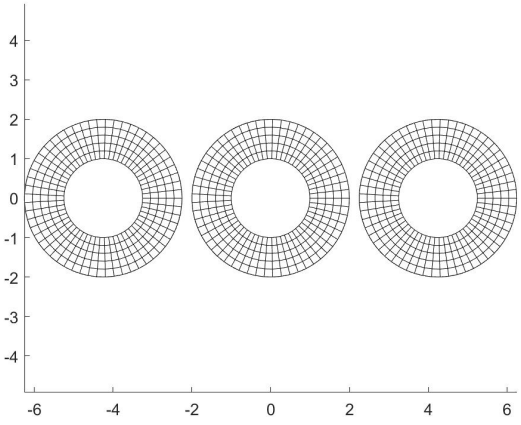
condition, we also get 5 decimals of accuracy after $PPW = 300$. Even though the Robin and Neumann problems don't achieve as much accuracy as quickly as the Dirichlet problem, the fact that we can still get 5 decimals of accuracy for any of these problems is quite amazing. It is extremely difficult in general for a second order finite difference scheme to get error below 1E-3. The finite difference algorithm presented in this thesis achieves error around the order of 1E-5. It is significant to note that running $PPW = 500$ for this configuration took less than 10 minutes on my home computer. I would have tried larger values of PPW, but the memory requirements for the LU factorization got too large.

The absolute value of the scattered wave and incident wave are plotted in Figure 4.1. The artificial boundaries are plotted in black, and the scattered wave is reconstructed in the region Ω^+ using (2.31).

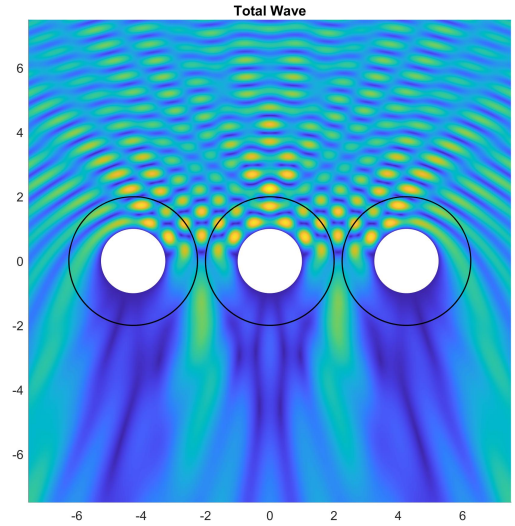
4.2 "Y"

We have shown in the previous experiment that the accuracy of our algorithm is incredibly good for a second order finite difference scheme. The purpose of this "Y" experiment is to show that we can vary the shape of the obstacles and the number of obstacles and still

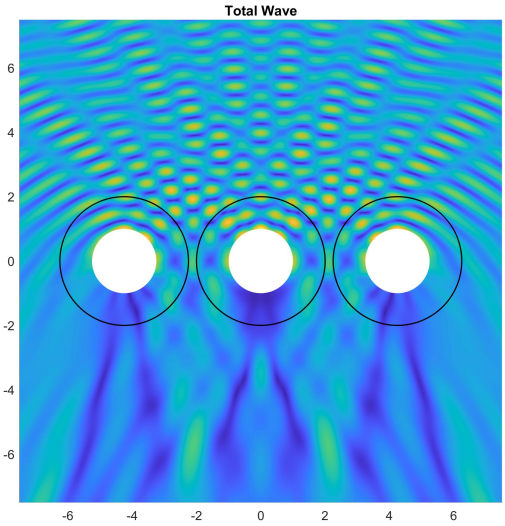
Figure 4.1: 3 Cylinders Mesh and Total Waves.



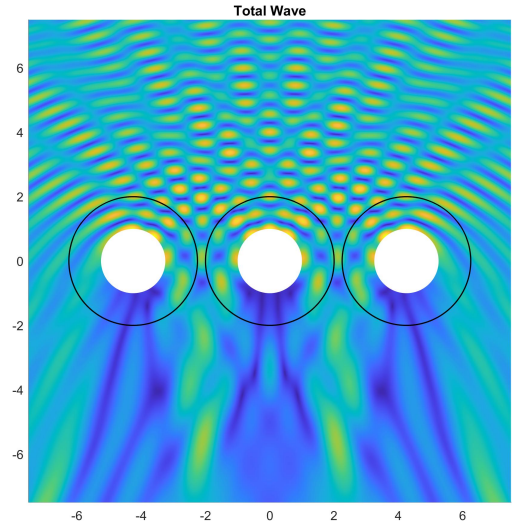
(a) 3 Cylinders Mesh, $PPW = 6$.



(b) $Z = 0$.



(c) $Z = 0.5$.



(d) $Z = 1$.

achieve great precision.

For this configuration, we have $\phi = -\pi/2$, $L = 10$, $k = 2\pi$ and $Tol =$ 1E-10. There are 2 cylinders of radius 1.25 and artificial boundary of radius 2.5 located at $[-4, 8]^T$ and $[4, 8]^T$. Another 2 cylinders are of radius 0.7 and artificial boundary of radius 1.7 located at $[-2, 4]^T$ and $[2, 4]^T$. The last 2 cylinders are of radius 0.5 and artificial boundary radius of 1.7, and are located at $[0, 0]^T$ and $[0, -4]^T$. An example computational grid of $PPW = 6$ is given for this configuration in Figure 4.2a.

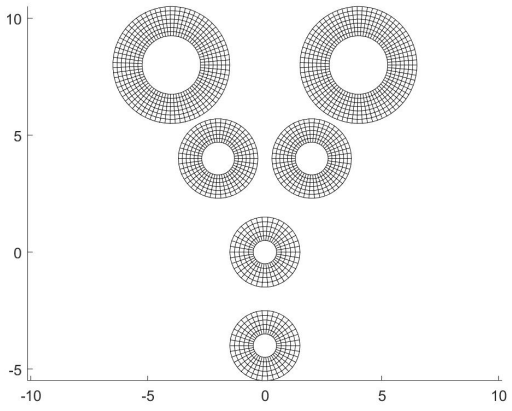
This configuration has 6 obstacles, which represents a very difficult problem for most

algorithms. For $PPW = 300$, the coupled solver given in [13] doesn't complete within 3 days because the system built is so large. The iterative method given in this thesis completed the $PPW = 300$ configuration in around 12 minutes, and gives accuracy on the order of 1E-5. We can also see that the errors for both the farfield pattern and the boundary converge with order 2, as shown in table 4.4.

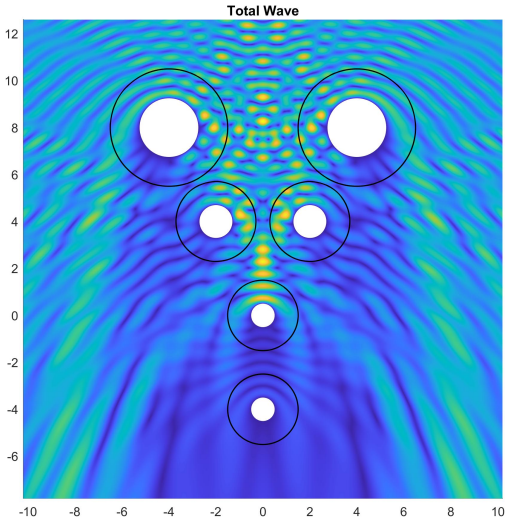
Table 4.4: Relative L2 Error Table for the "Y" Configuration

PPW	h	Farfield Pattern Error	Order	Boundary Error	Order
50	0.02000	2.30E-3	-	2.51E-3	-
100	0.01000	5.73E-4	2.00628	6.24E-4	2.00616
150	0.00667	2.54E-4	2.01069	2.76E-4	2.01440
200	0.00500	1.43E-4	1.99489	1.56E-4	1.99294
250	0.00400	9.12E-5	2.01095	9.93E-5	2.01145
300	0.00333	6.35E-5	1.99002	6.91E-5	1.98759

Figure 4.2: "Y" configuration



(a) "Y" mesh for $PPW = 6$.



(b) Total Wave, $PPW = 300$.

4.3 3 Non-Cylindrical Obstacles

An example computational grid of $PPW = 6$ is given for this configuration in Figure 4.3a.

The algorithm is intended to work for arbitrary geometry. In fact, arbitrary geometry is the whole purpose for creating a numerical method! We have an analytical solution for the

scattered waves off of cylinders, so the real challenge is solving the BVP where the obstacles are non-cylindrical.

For our non-cylindrical configuration, we have

$$\mathcal{B}^1_{in} = \begin{bmatrix} (1+\frac{1}{3}\cos(3\cos(1.75\cos(1.5\theta^1))))\cos(\theta^1) \\ (1+\frac{1}{3}\cos(3\cos(1.75\cos(1.5\theta^1))))\sin(\theta^1) \end{bmatrix}$$

$$\mathcal{B}^2_{in} = \begin{bmatrix} (2-\cos(\sin(3\theta^2)))\cos(\theta^2) \\ (2-\cos(\sin(3\theta^2)))\sin(\theta^2) \end{bmatrix}$$

$$\mathcal{B}^3_{in} = \begin{bmatrix} (1+\frac{1}{3}\cos(3\cos(1.75\cos(1.5\cos(2\theta^3)))))\cos(\theta^3) \\ (1+\frac{1}{3}\cos(3\cos(1.75\cos(1.5\cos(2\theta^3)))))\sin(\theta^3) \end{bmatrix}.$$

These three obstacles will all have an artificial boundary of radius 2.5, and are located at $[0,-2]^T$,$[5,0]^T$ and $[0.5,3.5]^T$, respectfully.

Since there is no analytical solution to this configuration, we can't compare the numerical result to an analytical solution. We can still, however, get a gauge on it's convergence. We can compare the numerical results we get for smaller PPW to a reference solution generated from an extremely refined grid. We can then gauge how well the numerical results converge to the reference solution. For this experiment, the reference solution used is the solution given from running the algorithm at $PPW = 500$.

We also have for this configuration that $\phi = -\pi/4$, $k = 2\pi$, $Z = 0$, and Tol=1E-10.

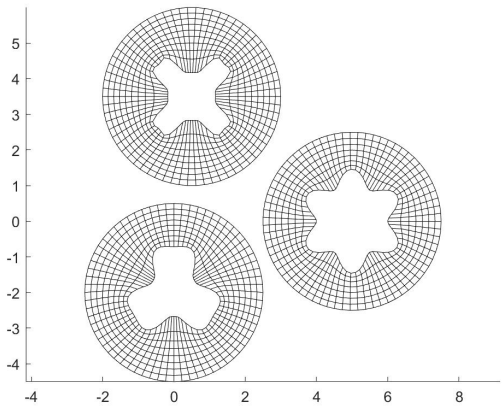
Table 4.5: Relative L2 Error Table for 3 Obstacles

PPW	h	Farfield Pattern Error	Order	Boundary Error	Order
20	0.02000	3.21E-2	-	3.50E-2	-
60	0.01000	4.12E-3	2.11377	3.54E-3	2.08490
100	0.00667	1.07E-3	2.12095	1.21E-3	2.09655
140	0.00500	5.15E-4	2.15454	5.83E-4	2.17642
180	0.00400	2.30E-4	1.89549	3.43E-4	2.11199

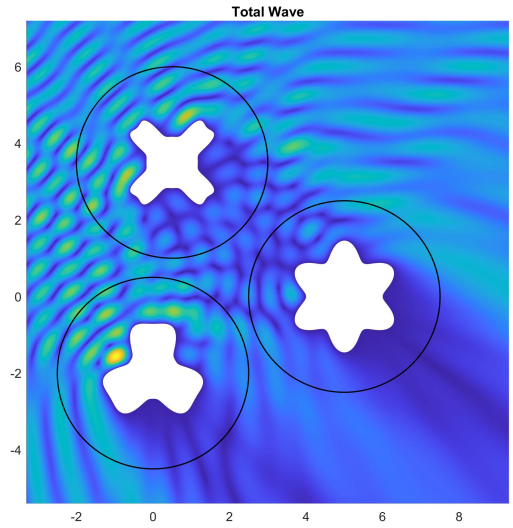
The table does show that the algorithm is converging as we refine our grid, and we even get down to the 1E-4 range in the errors starting at $PPW = 140$. The order of convergence for both the farfield pattern and the artificial boundary error are slightly more than 2, which

happens frequently when comparing to a reference solution.

Figure 4.3: 3 Obstacles Mesh and Total Wave



(a) 3 Obstacles Mesh, $PPW = 6$.



(b) 3 Obstacles Total Wave, $PPW = 180$.

Chapter 5. Conclusion

In this work, we have reduced the complex multiple scattering problem from M obstacles, which is given by the original BVP (1.2)-(1.4) for the total scattered wave u, to a system of M single scattering problems for the scattered waves u_m $(m = 1 \dots M)$, given by (2.1)-(2.3). This reduction is based on a decomposition of the total scattered field u as the sum of individual scattered fields u_m from each obstacle [14]. The novelty of our approach is that we complete the formulation of each of the scattered problems for u_m with a high order local absorbing boundary condition, which is based on Karp's farfield expansion representation of the outgoing wave from each obstacle. This novel ABC was introduced by Villamizar et al. [8] for a single scattering problem. The new system of BVPs is given by (2.16)-(2.22).

A numerical solution of this last formulation would require us to consider all the scattered field u_m as unknowns. As a result, a large system of equations coupling all of the u_m discrete values would need to be solved for each scatterer. To avoid this inconvenience, we adopt an

iterative formulation introduced by Geuzaine et al. [9] that allows us to uncouple the BVPs (2.16)-(2.22) into single scattering problems for each u_m given by (2.23)-(2.29). They are numerically solved in an iterative fashion. We have adopted both Jacobi and Gauss-Seidel iterative techniques in our formulations.

The iterative BVPs just described can be solved by any numerical method. In this work, we have adopted second order finite difference to approximate the scattered waves u_m inside their computational domains coupled with second order finite difference discretizations of the KFE at the artificial boundaries. As a result, we have derived an overall second order technique based on finite difference for the acoustic multiple scattering problems modeled by Helmholtz equations and KFE absorbing boundary conditions at the artificial boundaries of each scatterer.

The iterative formulation requires us to solve a BVP for each scatterer several times until the convergence is reached. This may be seen as a disadvantage compared with a coupled formulation approach, as the one recently introduced by Villamizar, Badger and Acosta [13]. The truth is that the iterative approach is computationally more efficient than the coupled one. First, the linear systems that result after discretization are much smaller than the coupled one, especially when we have many scatterers. Therefore, the condition number of these matrices is much smaller than the larger matrix of the coupled approach. As a consequence, the coupled method is limited to a few scatterers, while the iterative method, in principle, can handle any number of obstacles. Secondly, the solution of several linear systems until convergence is reached can be made relatively fast by using an LU decomposition, as shown in Section 3.4.1. In fact, the matrix defining the linear system at each iteration remains the same during the iteration process. If a certain configuration of M obstacles converges after N iterations, we can eliminate $(N-1)*M$ costly LU factorizations by doing M LU factorizations before the iteration process even begins. We can further save time by doing these M LU factorizations in parallel. Once we have the LU factorizations, solving the single scattering problem for each obstacle is as simple as a backward and forward

substitution, which can be done in less than a second for even high points per wavelength (see figure 3.5). In practice, the LU factorization applied to the 3 cylinders configuration in Figure 4.1a reduced the convergence time by over 90% (see table 3.1).

In Section 3.2, we reformulated the BVP for each scatterer in generalized curvilinear coordinates. This allows us to deal with obstacles of arbitrary shape by using a sufficiently smooth grid generator for the various scatterers. In this work, we have considered scattering from circular cylindrical scatterers and also from scatterers of more complex geometries. For the circular cylindrical scatterers, we compared our numerical results against the available exact solutions. For complex geometries, we generate reference solutions using a very refined grid and gauged convergence with respect to that reference solution. For our complex geometry example in Section 4.3, we used a reference solution based off a $PPW = 500$ grid and $L = 14$ terms in the Karp's Farfield Expansion ABC. Comparing to this reference solution, we achieved second order convergence for $PPW \in \{20, 60, 100, 140, 180\}$ (see table 4.5).

As mentioned in the body of this thesis, the iterative nature of our numerical method allowed us to pair it with parallelization and LU factorization. As a result our method ended being relatively fast compared with previous formulations [13, 11]. Our numerical results show the robustness and accuracy of our numerical method for Dirichlet (soft obstacles), Neumann (hard obstacles) and intermediate hardness ($0 < Z < 1$) BVPs. As seen in Tables 4.1-4.3, we could use the amazing quantity up to 500 points per wavelength for the three circular cylinders without issues relating to matrix conditioning numbers. This allowed us to obtain an L^2-error on the order of 1E-5, which is a true accomplishment for a second order technique. The "Y" configuration of Section (4.2) represents a challenge for any numerical approach. We were able to obtain the results presented in Figure 4.2 with PPW=300 in 12 minutes. Even more amazing is the fact that we could still get an error on the order of 10^{-5} for this configuration in only 4 minutes using $PPW = 250$. These times were achieved with 32 Gigabytes of ram and an Intel(R) Core(TM) i7-1065G7 CPU @ 1.30GHz quad-core processor.

Bibliography

[1] D. Colton and R. Kress. *Inverse Acoustic and Electromagnetic Scattering Theory*. Springer, second edition, 1998.

[2] P. Martin. *Multiple Scattering*. Cambridge Univ. Press, 2006.

[3] D. Givoli. Non-reflecting boundary conditions. *J. Comp. Phys.*, 94:1–29, 1991.

[4] D. Givoli. High-order non-reflecting boundary conditions. *Wave Motion*, 39:319–326, 2004.

[5] Samuel N Karp. A convergent 'farfield'expansion for two-dimensional radiation functions. *Communications on Pure and Applied Mathematics*, 14(3):427–434, 1961.

[6] A. Bayliss, M. Gunzburguer, and E. Turkel. Boundary conditions for the numerical solution of elliptic equations in exterior regions. *SIAM J. Appl. Math.*, 42:430–451, 1982.

[7] J. Keller and D. Givoli. Exact non-reflecting boundary conditions. *J. Comput. Phys.*, 82:172–192, 1989.

[8] Vianey Villamizar, Sebastian Acosta, and Blake Dastrup. High order local absorbing boundary conditions for acoustic waves in terms of farfield expansions. *Journal of Computational Physics*, 333:331–351, 2017.

[9] C. Geuzaine, A. Vion, R. Gaignaire, P. Dular, and R. Sabariego. An amplitude finite element formulation for multiple-scattering by a collection of convex obstacles. *IEEE Trans. Magnetics*, 46(8):2963–2966, August 2010.

[10] M. Balabane. Boundary decomposition for Helmholtz and Maxwell equations I: Disjoint sub-scatterers. *Asymptotic Anal.,*, 38(1):1–10, 2004.

[11] M. Grote and C. Kirsch. Dirichlet-to-Neumann boundary conditions for multiple scattering problems. *J. Comput. Phys.*, 201:630–650, 2004.

[12] Sebastian Acosta and Vianey Villamizar. Coupling of dirichlet-to-neumann boundary condition and finite difference methods in curvilinear coordinates for multiple scattering. *Journal of Computational Physics*, 229(15):5498–5517, 2010.

[13] V. Villamizar, J. Badger, and S. Acosta. High order local farfield expansions absorbing boundary conditions for multiple scattering. *Journal of Computational Physics*, 460:44 pages, 2022.

[14] Xue Jiang and Weiying Zheng. Adaptive perfectly matched layer method for multiple scattering problems. *Computer Methods in Applied Mechanics and Engineering*, 201-204:42 – 52, 2012.