AN INVESTIGATION INTO THE CREATION OF DIRECTIONAL RECEIVERS IN VIRTUAL ACOUSTIC MODELS IMPLEMENTED THROUGH USE OF FINITE DIFFERENCE TIME DOMAIN TECHNIQUES

Stephen G. Oxnard

s1152327

Final Research Project: Dissertation

MSc in Acoustics and Music Technology

Edinburgh College of Art

University of Edinburgh

2012

Abstract

This MSc Final Research Project report documents the work undertaken to develop receivers, that possess direction-dependent response characteristics, for use within virtual models rendered using finite difference time domain techniques. In total, three MATLAB schemes were devised and implemented to produce receivers with polar response characteristics of bi-directional and cardioid forms. Observations drawn from results and initial testing procedures provide insight into the difficulties faced when attempting to attain the aims and objectives of the project as initially proposed. Receivers with directional qualities were implemented successfully, however, further research into ensuring these qualities remain constant for all audible frequencies is required. Conclusions, based on the progress of the project to date, offer a basis for the proposition of further work.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Stephen G. Oxnard*
s1152327)

Table of Contents

Chapter 1

Introduction

1.1 Project Overview

In recent years, virtual acoustic modelling has been shown to be a highly useful and important practise across a number of applications. These applications include the sonic revival of soundscapes [1], that are no longer in existence or accessible, the physical modelling of musical instruments [2] and the recording of architectural acoustics for posterity [3], amongst others. A number of approaches to realising a virtual acoustic model exist currently. For example, ray-based approaches, 'Finite-Element Method' (FEM) and 'Finite Difference Time Domain' (FDTD) methods, may all be utilised to produce a virtual environment suitable for calculating predictions of acoustical parameters. Each method possesses inherent strengths and weaknesses. Ray-based methods, such as ray-tracing and image source methods, seek to approximate the propagation of sound throughout a space by employing a large number of 'infinitely thin' rays to investigate the given modelled space [4]. This approach, although computationally more efficient than FDTD schemes, has the disadvantage that sound wave characteristics are poorly represented, therefore low frequency phenomena such as diffraction and room modes are neglected during simulation. Finite Difference approaches allow the continuous wave equation to be descritized and directly implemented algorithmically ensuring that wave-like behaviour is preserved and well modelled. Virtual acoustic models developed through use of FDTD schemes naturally possess a high level of accuracy, but require greater computational resources than that of ray-based methods.
The studies documented in this report are concerned with the creation of new meth-

ods for the extraction of acoustical data from virtual models realised through use of FDTD schemes. Optimization techniques were employed and amalgamated with FDTD schemes in an attempt to generate virtual receivers that behaved in a manner similar to directional microphones. Primarily, the focus of this research was on modelling two common microphone response types: bi-directional (figure of eight) and cardioid (see section 3.2.1). In order to approach this outcome, the outputs taken from a square (2D case) or cube (3D case) of points surrounding a defined receiver position within a scheme may be weighted by a set of pre-calculated coefficients. The crux of this problem lies in determining correct coefficients for the desired response or polar pattern (see Chapter 3).

The subsequent chapters of this document detail the development and solution of the underlying mathematics that define the virtual receivers (Chapter 3). Computation of the devised schemes is discussed (Chapter 4) and consequent results are given (Chapter 5). Conclusions drawn from results and routes of further work are given in the closing chapter.

1.2 Motivation

Currently, it is common practise to output data from a FDTD scheme by recording the nature of the acoustic field at one or more defined receiver points. This method naturally gives rise to an omni-directional response sensitivity at the receiver position/s, which is suitable for applications such as modal analysis and simple auralizations. By employing a specified array of direction-dependent receivers (cardioid) within FDTD schemes, it would be possible to capture surround sound impulses that could then be harnessed for auralizations played across a surround sound array. The advantage of utilising surround sound loudspeaker arrays lies in the natural directivity of the sound experienced by a listener that is lost when playback is carried out over headphones. Ultimately, success in this area would lead to the ability to better experience the sonification of a space modelled through use of a technique which may be deemed far superior to traditional methods (such as ray-tracing) for reasons given previously.

It may also be assumed that the methods employed to produce directional receivers may also be used in the creation of directional sound sources. As such, this work could potentially impact on the way in which musical instruments, loudspeakers and other sources of sound are implemented using the FDTD approach.

Furthermore, through the application of both directional sources and receivers, the testing procedure that compares results taken from real spaces and those from their virtual representations may be further refined. In most cases, when acoustical data is recorded from a space, great effort is taken to approach ideal conditions. In practical applications however, fully omni-directional point sources and receivers do not currently exist. Instead, the methods approached in this project could be utilised to model the non-ideal aspects of the equipment used in practise when gathering data from the virtual model of the space. This could potentially minimise the descrepancies between modelled and actual results when attempting to ensure the correct operation of a virtual acoustic model by comparison.

1.3 Project Specifications

The specifications for this project, as intially proposed, are given below in the form of a concise set of aims and objectives:

• Develop a means of producing virtual receivers in FDTD schemes that possess frequency-dependent directionality. In this way, the response characteristics of the receivers produced will mimic that of mircophones with particular polar pick-up patterns (i.e. cardioid, bi-directional, hyper-cardioid).

• Embed the receivers in both 2-Dimensional and 3-Dimensional FDTD schemes in order to carry out initial testing.

• Utilise an array of cardioid receivers to collect impulse responses from FDTD acoustic model simulations that can then be used to produce auralizations across a surround sound array (i.e. Stereo and ITU 5.1 Surround Sound).

1.4 Software Overview

Mathworks MATLAB (R2011a) was utilised for the implemention of algorithms developed during the course of this project. MATLAB provides a highly sophisticated

algorithm development environment that is ideal for implementing and evaluating the FDTD schemes derived. Code structures, such as for loops and if statements, may be used in MATLAB alongside complex mathematical functions and plotting facilities that are easily utilised. Examples of the MATLAB code constructed are available for reference in the Appendices.

Chapter 2

Literature Review

Key references and texts that were central to progress and development of ideas over the course of the project are discussed here. Conclusions and outcomes gained from these relevent materials, and applied to the project, are noted as appropriate.

2.1 Virtual Acoustics and Auralization

2.1.1 Simulation of Room Acoustics with a 3-D Finite Difference Mesh - Savioja, Rinne and Takala

"Simulation of Room Acoustics with a 3-D Finite Difference Mesh" [5] serves as a concise introduction to the various, more commonly used, approaches to virtual acoustic modelling. The limitations of geometrical methods, such as ray-tracing, are highlighted and it is proposed that the finite difference scheme detailed may be used in conjuction with ray-based approaches with each approach operating across appropriate frequency bands. Mathematical formulation of the FDTD scheme is provided alongside details of the different conditions which may be applied to the terminating edges of the virtual acoustic field. The implementation of the derived scheme is irrelevant to the purposes of this project due to the fact that the algorithms developed were created using a different programming environment. However, the results presented are of interest as they show evidence of the preservation of wave-like characteristics within the schemes. This demonstrates the superiority of FDTD techniques over geometrical approaches in terms of accurate modelling and justifies further research into the uses

of this approach.

2.1.2 Recording Concert Hall Acoustics for Posterity - Farina and Ayalon

Angelo Farina and Regev Ayalon composed "Recording Concert Hall Acoustics for Posterity" [6] in order to document and expose a new, practical approach to acoustics measurement. This approach involved creating a sound receiver consisting of an array of microphones which could be rotated around the azimuth plane. As such, impulse responses could be recorded within a given space as the rotation of the 'listening' postion was varied. From the data collected, it was not only possible to discern values for acoustical parameters, but also to produce auralizations in a number of surround sound formats. Upon reading this text, it became apparent that a plug-in, one that reproduces this procedure in virtual acoustic models realised through application of geometric modelling techniques, had been developed. It was then decided that an attempt to extend this idea to FDTD schemes should be undertaken.

2.1.3 Auralization - An Overview - Kleiner, Dalenbäck and Svensson

In "Auralization - An Overview" [7], by Mendal Kleiner, Bengt-Inge Dalenbäck and Peter Svensson, the subject process is defined as follows:

"Auralization is the process of rendering audible, by physical or mathematical modelling, the sound field of a source in space, in such a way as to simulate the binaural listening experience at a position in the modelled space." [7]

As initially proposed, auralizations were to be rendered using a collection of directional impulse responses captured through use of the FDTD schemes devised. Although this paper is relatively dated, it served as a useful reference as it defines and describes two relevant techniques: "Fully Computed Auralization" and "Computed Multiple-Loudspeaker Auralization" [7]. The latter of the two techniques was deemed most approriate for the purposes of this project. From the section dedicated to this

method, the following notable facts were gathered:

• An advantage of using computed multiple-loudspeaker auralization is that a natural directivity arises from the use of a loudspeaker array.

• A disadvantage of this technique stems from the fact that, ideally, auralizations should take place in an anechoic chamber in order that the sound emitted by the loudspeakers may not be influenced by the reverberation of the room.

• Comment on the specifics of the loudspeakers to be used is given - they "must be small and full range. Large loudspeakers will give rise to scattering, which will contaminate the calculated RIR and create a false impression of diffuseness." [7]

Obtaining a rudimentary understanding of the auralization process was of importance when considering the potential for subjective testing of the virtual models produced. In addition, this research generated the idea of exploiting the advantages of multiple loud speaker arrangements to produce auralizations which possess an inherent realism.

2.2 Finite Difference Time Domain Schemes

2.2.1 Math 257: Finite Difference Methods - I J Hewitt

"Math 257: Finite Difference Methods", by I J Hewitt [8] is an online article that discusses the formulation of finite difference approximations from first principles. The information provided in this resource provided the basis for deriving the recursive update schemes, detailed in section 3.1, which seek to discretize and approximate the 2D and 3D wave equations. Graphical interpretations of the 2-Dimensional systems are provided as an aid to visualization and understanding of the schemes in operation. Boundary conditions, which must be applied to the edges of every virtual domain topology to ensure correct operation, are also given. For the purposes of this project, the simplest 'fixed' condition was deemed appropriate for use (as defined in section 3.1.4).

2.2.2 Numerical Sound Synthesis - Stefan Bilbao

"Numerical Sound Synthesis", by Stefan Bilbao [2], dedicates numerous chapters to materials relevant to the studies documented in this report. In particular, materials relating to frequency analysis and Von Neumann stability analysis were highly useful when developing a means of implementing both the wave equation and the virtual receiver patterns in MATLAB. Details of these analytical processes and their use within this project are given in Chapter 3 (Scheme Development).

2.3 Optimization

2.3.1 Numerical Computing with MATLAB - Cleve Moler

"Numerical Computing with MATLAB" [9], by Clevel Moler, is available as an online resource. This text explains the theory behind a host of numerical methods and the means by which they can be applied to a problem computationally within MATLAB. Chapter 5, concerned with the least squares method, was of particular importance for aiding the studies undertaken in this project. This optimization method was utilised in order to determine the unknown coefficients within the derived schemes which, when applied to weight a summation of outputs from a FDTD scheme, would produce a directional polar response curve. Details on the approaches MATLAB employs to solve a least squares problem (such as construction of the pseudo-inverse, QR factorisation etc) are provided, offering the reader an insight into the internal mechanisms behind functions such as the '\' command (see section 4.1.2).

2.3.2 Foundations of Optimization - Wilde and Beightler

"Foundations of Optimization" [10], was sourced to provide further background theory on optimization approaches such as least squares. In addition, alternative means of creating approximate polar curves were sourced from this resource and highlighted for potential use. These approaches included gradient, or steepest, descent method, simplex method and the bi-conjugate gradient method. Unfortunately, extensive study into the application of these alternatives was abandoned due to the current implementation proving problematic (see Chapters 5 and 6).

Chapter 3

Scheme Development

Formulation of the mathematical expressions and processes that underpin the directional receiver schemes is provided in this chapter. In additon, the derivation of finite difference approximations to the 2D and 3D wave equations are described as a precursor to the explanation of the algorithms used to produce response patterns. Details of these FDTD schemes are given here for two reasons. Firstly, the schemes derived in the following section are those in which the results gained from the response pattern algorithms were to be used. Secondly, stability analysis follows on naturally from the derivations leading to a result that is essential for correct implementation of the algorithms produced.

3.1 Finite Difference Approximations to the Wave Equation

The following section imparts the development of the finite difference schemes employed to model wave propagation in 2D and 3D space. Although these schemes are documented fully in various texts, it was decided that derivations from first principles would be constructed in order to ensure a clear understanding of their origin and meaning. Formulation of fixed boundary conditions was also carried out and is described here alongside the related stability criterion.

3.1.1 Finite Difference Approximations to 1st and 2nd Order Differentials

The definition of the first derivative of some function $f(x)$ is as follows:

$$f'(x) = \lim_{\triangle x \to 0} \frac{f(x+\triangle x) - f(x)}{\triangle x} \tag{3.1}$$

Using Taylor's Series Expansion, the following expressions for the values of $f(x)$ at positions 'ahead' and 'behind' by a finite amount may be obtained:

$$f(x+\triangle x) = f(x) + \triangle x f'(x) + \frac{\triangle x^2}{2!} f''(x) + \frac{\triangle x^3}{3!} f'''(x) + ... \tag{3.2}$$

$$f(x-\triangle x) = f(x) - \triangle x f'(x) + \frac{\triangle x^2}{2!} f''(x) - \frac{\triangle x^3}{3!} f'''(x) + ... \tag{3.3}$$

Having defined the above expressions it is then possible to form the relevant finite difference approximations.

3.1.1.1 1st Order Approximations

Rearranging equation (3.2) by taking $f(x)$ to the left hand side and dividing through by $\triangle x$ yields:

$$\frac{f(x+\triangle x) - f(x)}{\triangle x} = f'(x) + O(\triangle x) \tag{3.4}$$

Where $O(\triangle x)$ represents the remaining terms of order $\triangle x$ on the right hand side of the equation [8]. By similar means, equation (3.3) may be rearranged into the following form:

$$\frac{f(x) - f(x-\triangle x)}{\triangle x} = f'(x) + O(\triangle x) \tag{3.5}$$

Equations (3.4) and (3.5) are the forward difference and backward difference approximations respectfully. These definitions are known to be 'first order accurate' as the error introduced is of order $\triangle x$ [8]. By subtracting equation (3.3) from equation (3.2) one arrives at the centered difference approximation:

$$\frac{f(x+\triangle x)-f(x-\triangle x)}{2\triangle x} = f'(x)+O(\triangle x^2) \tag{3.6}$$

The above approximation is'second order accurate' due to the error introduced being of order $\triangle x^2$ [8]. In the following sections, the error term present in each of these equations is neglected as it is assumed that $\triangle x$ is sufficiently small.

3.1.1.2 2nd Order Approximation

The centered difference approximation to the second derivative of some function $f(x)$ is obtained through summation of equations (3.2) and (3.3):

$$\frac{f(x+\triangle x)-2f(x)+f(x-\triangle x)}{\triangle x^2} = f''(x)+O(\triangle x^2) \tag{3.7}$$

As before, $O(\triangle x^2)$ represents terms of order $\triangle x^2$ and the approximation is therefore second order accurate.

3.1.2 Formulating a Finite Difference Scheme for the 2D Wave Equation

The continuous-time 2D wave equation is defined as follows:

$$\frac{\partial^2 u}{\partial t^2} = c^2 \bigtriangledown^2 u \tag{3.8}$$

where 'c' is the wave speed in metres per second, 't' is time in seconds, $\bigtriangledown$ is the 2D Laplacian operator and u(x,y,t), referred to as the 'target acoustic field' [11], is dependent on time and two spatial dimensions 'x' and 'y' (measured in metres). Rewriting equation (3.8) in its less compact form demonstrates the potential for using the finite difference approximations previously defined.

$$\frac{\partial^2 u}{\partial t^2} = c^2(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}) \tag{3.9}$$

Obviously it is impossible to digitally compute such an equation and so it must first be discretized. The variables t, x and y are defined as having finite length from zero to

any given positive amount. The time variable increments by equal discrete amounts or 'time steps (s)' up to a given time. Similarly, the spatial variables increment by equal discrete amounts across a 'grid' defined over a 2D plane of a given area (in metres squared). For the remainder of this report the discrete change in space will be referred to as the 'grid spacing' and it will be assumed that the spacing is the same in both the x and y directions giving a rectilinear grid topology.
The left hand side of equation (3.8) may be discretized as follows:

$$\frac{f(u+\triangle t)-2f(u)+f(u-\triangle t)}{\triangle t^2} \Rightarrow \frac{u_{l,m}^{n+1}-2u_{l,m}^{n}+u_{l,m}^{n-1}}{k^2} \tag{3.10}$$

when one wishes to approximate the second derivative of $u(x,y,t)$ with respect to time. In this case, and all to follow, 'k' is the discrete time step of which there is an integer number ('n'). 'l' and 'm' are the number of discrete grid spacings in the x and y direction respectfully. The length of each grid spacing is of length 'h' (m) which will be introduced shortly.
By employing the discretization shown in (3.10), the 2D wave equation may be re-expressed in its entirety as follows:

$$\frac{u_{l,m}^{n+1}-2u_{l,m}^{n}+u_{l,m}^{n-1}}{k^2} = c^2\left(\frac{u_{l+1,m}^{n}-2u_{l,m}^{n}+u_{l-1,m}^{n}}{h^2}+\frac{u_{l,m+1}^{n}-2u_{l,m}^{n}+u_{l,m-1}^{n}}{h^2}\right) \tag{3.11}$$

where all variables are defined as they have been previously. It is worth reiterating here that the discrete change in the x direction is always equal to that in the y direction and, hence, both of the second order approximations on the right hand side of the equation appear over a common term h^2. In order to formulate a recursive update scheme, equation (3.11) must be rearranged for the unknown term, $u_{l,m}^{n+1}$, assuming the variable u has appropriate initial conditions.

$$u_{l,m}^{n+1} = \lambda^2(u_{l+1,m}^{n}+u_{l-1,m}^{n}+u_{l,m+1}^{n}+u_{l,m-1}^{n})+2(1-2\lambda^2)u_{l,m}^{n}-u_{l,m}^{n-1} \tag{3.12}$$

where,

$$\lambda = \frac{ck}{h} \tag{3.13}$$

Equation (3.12) describes the finite difference scheme used during the implementation stage of this project and is further documented in Chapter 4.

3.1.3 Formulating a Finite Difference Scheme for the 3D Wave Equation

The continuous-time form of the 3D wave equation takes exactly the same form as the 2D case except the Laplacian operator is now 3-Dimensional. In its non-compact form the 3D case is written as follows:

$$\frac{\partial^2 u}{\partial t^2} = c^2\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2}\right) \tag{3.14}$$

where 'z' represents the additional dimension. It is trivial to follow the steps in the previous section and arrive at a discretized approximation to the equation above:

$$\frac{u^{n+1}_{l,m,p} - 2u^{n}_{l,m,p} + u^{n-1}_{l,m,p}}{k^2} = c^2\left(\frac{u^{n}_{l+1,m,p} - 2u^{n}_{l,m,p} + u^{n}_{l-1,m,p}}{h^2} + \frac{u^{n}_{l,m+1,p} - 2u^{n}_{l,m,p} + u^{n}_{l,m-1,p}}{h^2} + \frac{u^{n}_{l,m,p+1} - 2u^{n}_{l,m,p} + u^{n}_{l,m,p-1}}{h^2}\right) \tag{3.15}$$

where 'p' is the integer number of grid spacings present in the z direction. This equation may also be rearranged for the unknown value $u^{n+1}_{l,m,p}$ producing a recursive finite difference update scheme which can be implemented in MATLAB.

$$u^{n+1}_{l,m,p} = \lambda^2 S^{n}_{l,m,p} + 2(1-3\lambda^2)u^{n}_{l,m,p} - u^{n-1}_{l,m,p} \tag{3.16}$$

where

$$S^{n}_{l,m,p} = (u^{n}_{l+1,m,p} + u^{n}_{l-1,m,p} + u^{n}_{l,m+1,p} + u^{n}_{l,m-1,p} + u^{n}_{l,m,p+1} + u^{n}_{l,m,p-1}) \tag{3.17}$$

and 'λ' is defined as it is in the 2D case.

3.1.4 Boundary Conditions

A simple mathematical condition was applied to the boundaries of the FDTD model detailed in Chapter 4. This simple condition is known as the 'fixed' or 'Dirichlet' boundary condition. For the 2D wave equation, this condition is defined as follows:

$$u(0,y,t) = 0, u(L,y,t) = 0, u(x,0,t) = 0, u(x,M,t) = 0 \tag{3.18}$$

where $0 < x < L$ and $0 < y < M$. If one takes $u(x,y,t)$ as describing the displacement of a 2D surface (or membrane), then the terminating edges will not move and can be thought of as 'physically clamped'.

3.1.5 Stability Conditions (Von Neumann Analysis)

3.1.5.1 Overview of Stability Analysis

Von Neumann stability analysis is a means of establishing whether a numerical scheme is stable without computation. By inserting a suitable ansatz, of the form $e^{jkn\omega+jalh+j\beta mh}$, into the 2D recursive schemes derived in the previous section, it is possible to find a condition on λ (and therefore the grid spacing h) such that ω is real $\forall\alpha,\beta$. If this condition is upheld, one may assume that the solution of the scheme in question is oscillatory for any given wave number. This technique may be applied to 3D models also.

3.1.5.2 2 Dimensional Case

In order to analyse the stability of the discretized 2D wave equation (3.11), one must firstly define the input ansatz. This will take the form of a discrete, single frequency wave which possesses the appropriate number of dimensions, thus:

$$[u] = e^{j\omega kn+jalh+j\beta mh} = e^{jkn\omega}e^{jalh}e^{j\beta mh} \tag{3.19}$$

Inserting the above value of $[u]$ into equation (3.11) yields a rather extensive expression that is omitted here for brevity. However, simplifying the resulting expression results in a more compact version:

$$e^{j\omega k} - 2 + e^{-j\omega k} = \lambda^2(e^{jah} - 2 + e^{-jah} + e^{j\beta h} - 2 + e^{-j\beta h})$$

Applying the inverse Euler identity, $\cos(\theta) = \frac{1}{2}(e^{j\theta} + e^{-j\theta})$, gives:

$$2\cos(\omega k) - 2 = \lambda^2(2\cos(ah) - 2 + 2\cos(\beta h) - 2)$$

and so,

$$\cos(\omega k) - 1 = \lambda^2(\cos(\alpha h) - 1 + \cos(\beta h) - 1)$$

Using the standard trigonometric identity, $\cos(\theta) = 1 - 2\sin^2(\frac{\theta}{2})$, and dividing through by -2 yields:

$$\sin^2(\frac{\omega k}{2}) = \lambda^2(\sin^2(\frac{\alpha h}{2}) + \sin^2(\frac{\beta h}{2}))$$

and by taking the square root,

$$\sin(\frac{\omega k}{2}) = \lambda((\sin^2(\frac{\alpha h}{2}) + \sin^2(\frac{\beta h}{2}))^{\frac{1}{2}}$$

The stability condition for the 2D wave equation can be found by examining the maximum values of all terms in the above equation when $\omega k = \alpha h = \beta h = \pi$:

$$\begin{aligned} 1 &\geq \lambda\sqrt{2} \\ 1 &\geq \frac{ck}{h}\sqrt{2} \\ h &\geq ck\sqrt{2} \end{aligned} \tag{3.20}$$

Therefore, it is apparent that the grid spacing 'h' must be equal to a value greater than or the same as $ck\sqrt{2}$, where 'c' is the wave speed and 'k' is the time step.

3.1.5.3 3 Dimensional Case

A suitable input ansatz for the descritized 3D wave equation is,

$$[u] = e^{j\omega kn + j\alpha lh + j\beta mh + j\zeta ph} = e^{j\omega kn} e^{j\alpha lh} e^{j\beta mh} e^{j\zeta ph} \tag{3.21}$$

Now, by inserting $[u]$ into equation (3.15), one arrives at:

$$e^{j\omega k} - 2 + e^{-j\omega k} = \lambda^2(e^{j\alpha h} - 2 + e^{-j\alpha h} + e^{j\beta h} - 2 + e^{-j\beta h} + e^{j\zeta h} - 2 + e^{-j\zeta h})$$

Employing exactly the same steps as those taken for the 2D case, the above equation may be simplified to,

$$\sin(\frac{\omega k}{2}) = \lambda((\sin^2(\frac{\alpha h}{2}) + \sin^2(\frac{\beta h}{2}) + \sin^2(\frac{\zeta h}{2}))^{\frac{1}{2}}$$

Finally, by letting $\omega k = \alpha h = \beta h = \zeta h = \pi$ in order to examine maximum values:

$$\begin{aligned} 1 &\geq \lambda\sqrt{3} \\ 1 &\geq \frac{ck}{h}\sqrt{3} \\ h &\geq ck\sqrt{3} \end{aligned} \tag{3.22}$$

Therefore the grid spacing 'h' must be greater than or equal to $ck\sqrt{3}$ to ensure numerical stability within the system.

3.2 Modelling 2D Response Patterns

3.2.1 Polar Patterns

Polar plots are commonly utilised to display the sensitivity of a microphone at a given frequency, with respect to the direction of incoming sound. It was intended that this form of plot would serve as the input data for the directional receiver schemes. As such, the nature of two examples of polar patterns are briefly described here. These curves can be defined mathematically as a function of magnitude and angle, for example:

$$r = acos^2(\theta) \tag{3.23}$$

defines a bi-directional polar pattern, where θ is the angle of incoming sound in the range $0^o - 360^o$ and a corresponds to the magnitude of the response at $\theta = 0^o, 180^o$. A cardioid response pattern may also be defined as:

$$r = a(cos(\theta) + 1) \tag{3.24}$$

where the values $2a$ and a relate to the magnitude of the response at $\theta = 0^o$ and $\theta = 90^o, 270^o$ respectively. An example of each defined response is provided in figure (1).

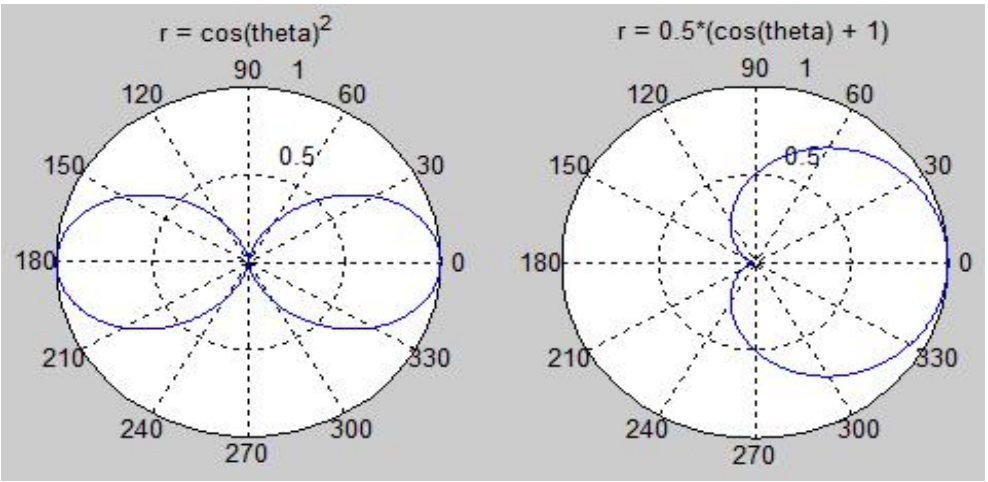

Figure (1): *Polar plots of bi-directional (left) and cardioid (right) patterns as defined in equations (3.23) and (3.24) respectively.*

As shown in the diagram above, the bi-directional pick up pattern describes a receiver that is equally receptive to sound arriving from in front or behind, whereas the cardioid response seeks to reject incoming sound from behind.
Having mathematically defined the response patterns related to directional receivers, it is now possible to develop a means of modelling them.

3.2.2 Problem Statement

As previously noted, it is known that defining a receiver within an FDTD scheme at a single grid point gives rise to an omni-directional response. In order to calibrate the sensitivity of the receiver such that it is directional, an array of grid points must contribute to a total output, i.e:

$$Output = \sum_{l,m} a_{l,m} u_{l,m} \tag{3.25}$$

Equation (3.25) relates to a 2-Dimensional system and states that the required output from a scheme is the weighted sum of data taken from points $u_{l,m}$ where $a_{l,m}$ are the weighting coefficients. The subscripts l,m are integer values defining the number of grid spacings (in the x,y directions respectively) between an output point and the receiver position (defined at $u_{0,0}$ for simplicity). Figure (2) portrays a graphical description of such a receiver system using a 3 x 3 array of grid points.

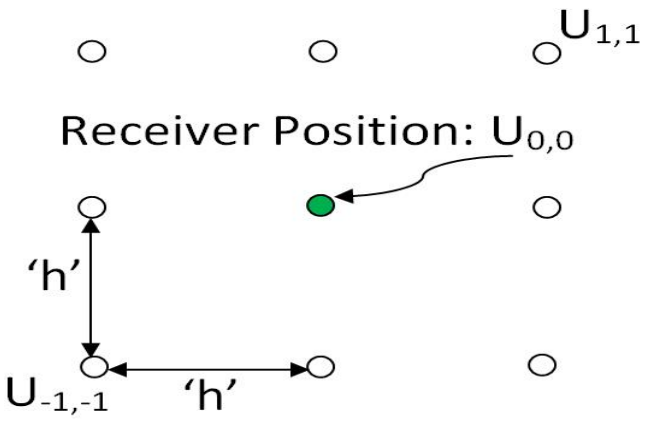

Figure (2): *Diagram of a 3x3 grid of points: The receiver position, signified by the green circle at location $U_{0,0}$, is surrounded by a number of contributing grid points at equal spacings of $'h'$ (defined using stability analysis).*

It is necessary to apply frequency analysis to the defined output in order to ascertain the nature of the receiver sensitivity and to calibrate it accordingly. In a manner similar to Von Neumann analysis, this process is initiated by first defining a test solution, or ansatz. An appropriate ansatz takes the form of a single frequency planar wave. For the 2-Dimensional case this is defined as follows:

$$u_{l,m} = e^{j(\beta_x lh + \beta_y mh)} \tag{3.26}$$

where h is the discrete grid spacing (defined as per section 3.1.5) and β_x and β_y are contributions to wave number in polar coordinates, more explicity:

$$\beta_x = (\frac{w}{c})cos(\theta)$$
$$\beta_y = (\frac{w}{c})sin(\theta)$$

where w represents frequency in radians per second, c is wave speed in metres per second and θ is the angle of incoming sound over the 2D azimuth. Note that the ansatz given in equation (3.26) does not contain a time component. This is due to the fact that the test solution aims to yield the nature of the frequency response at any given present time step and does not rely on data captured at historic or future time steps within a recursive scheme.

The weighted summation of responses from multiple grid points, that aims to mimic the response of a directional receiver, may now be defined as:

$$Frequency\,Response\,Pattern = \sum_{l,m} a_{l,m} e^{j(\beta_x lh + \beta_y mh)} \tag{3.27}$$

The unknown coefficients $a_{l,m}$ must now be calibrated such that the output polar curve matches that of a pre-defined function $r(\theta)$. At this point, it is evident that the problem can be solved through curve fitting, for which the least squares optimization method may be utilised. It can be stated that the cofficients $a_{l,m}$ must be set to values which minimise the error between the function r and the output response. The least squares approach is initialised by defining this error as the magnitude of the difference between the two responses squared.

$$E = \int_\theta (|\sum_{l,m} a_{l,m} e^{j(\beta_x lh + \beta_y mh)} - r|)^2 \delta\theta \tag{3.28}$$

The squared difference in the above equation is integrated over θ in the range $0^o - 360^o$ such that, when the least squares procedure is followed, $a_{l,m}$ can be calculated by matching areas under functions of θ. A concise description of the means by which this problem was solved is given in the following section.

3.2.3 Solution Using a Single Time Step

Applying the standard result, $|x| = xx^*$, to equation (3.28) yields an expression for the error terms that is differentiable with respect to a particular value of $a_{l,m}$,

$$E = \int_\theta [\sum_{l,m} a_{l,m} e^{j(\beta_x lh + \beta_y mh)} - r][\sum_{l,m} a_{l,m} e^{-j(\beta_x lh + \beta_y mh)} - r]\delta\theta \tag{3.29}$$

To proceed, the product rule may be applied in order to find the derivative of the above expression with respect to a_{l^*,m^*}, a coefficient related to one possible combination of l and m,

$$\frac{\partial E}{\partial a_{l^*,m^*}} = \int_\theta e^{-j(\beta_x l^* h + \beta_y m^* h)} [\sum_{l,m} a_{l,m} e^{j(\beta_x lh + \beta_y mh)} - r] +$$

$$e^{j(\beta_x l^* h+\beta_y m^* h)}[\sum_{l,m} a_{l,m} e^{-j(\beta_x lh+\beta_y mh)} - r]\delta\theta \tag{3.30}$$

Multiplying out and reversing the order of summation and integration gives,

$$\frac{\partial E}{\partial a_{l^*,m^*}} = \sum_{l,m} a_{l,m} \int_\theta e^{j(\beta_x(l-l^*)h+\beta_y(m-m^*)h)} + e^{-j(\beta_x(l-l^*)h+\beta_y(m-m^*)h)} - r(e^{j(\beta_x l^* h+\beta_y m^* h)} + e^{-j(\beta_x l^* h+\beta_y m^* h)})\delta\theta \tag{3.31}$$

Using Euler's identity, the exponential terms reduce to cosines as the imaginary parts of each term cancel.

$$\frac{\partial E}{\partial a_{l^*,m^*}} = \sum_{l,m} a_{l,m} \int_\theta 2cos(\beta_x(l-l^*)h+\beta_y(m-m^*)h) - 2rcos(\beta_x l^* h+\beta_y m^* h)\delta\theta \tag{3.32}$$

In order to minimise the error terms, in accordance with the least squares method, the derivative is set to zero leaving the two terms on the right hand side to be equated as follows,

$$\sum_{l,m} a_{l,m} \int_\theta cos(\beta_x(l-l^*)h+\beta_y(m-m^*)h)\partial\theta = \int_\theta rcos(\beta_x l^* h+\beta_y m^* h)\delta\theta \tag{3.33}$$

Equation (3.33) is now a system of equations of the form,

$$\sum_{l,m} a_{l,m} K_{l,m,l^*,m^*} = b_{l^*,m^*} \tag{3.34}$$

when considering all possible combinations of particular values of l and m (denoted by l^* and m^*). This system of equations can be better visualised when displayed in matrix form as shown in figure (3).

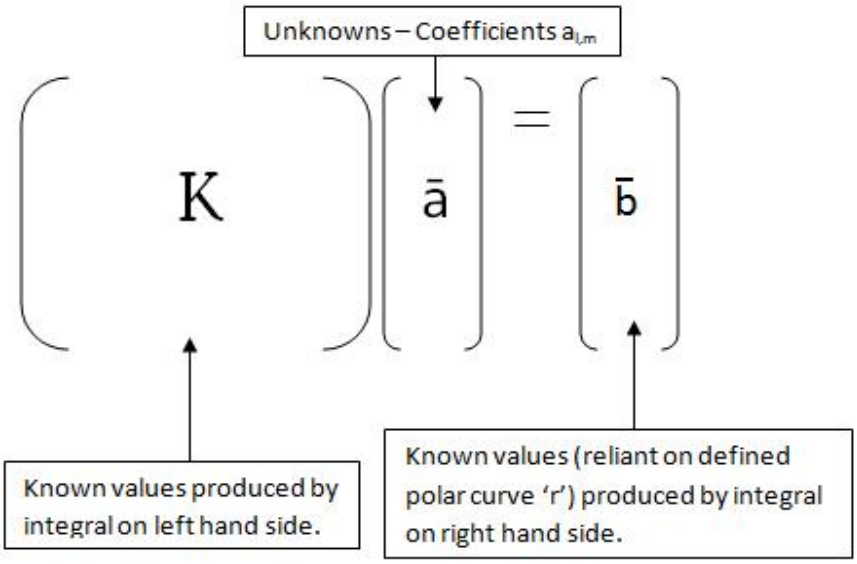

Figure (3): *System of equations for the solution using a single time step depicted in matrix form.*

Details of the implementation and computation of this system are provided in section 4.2.

3.2.4 Solution Using Multiple Time Steps

While gathering results from the implementation of the algorithm presented in the previous section, it became apparent that the system can not converge on polar curves that are symmetric about one axis only (i.e. cardioid). In order to overcome this problem, an algorithm that looks across three time steps was devised. The initial problem given in equation (3.28) was extended to include the weighted sum of read-outs taken from each grid point at previous and next time steps. Recalling that a shift of a single discrete time step, k, is represented by a multiplication by e^{jwk} in the frequency domain, the additional ansatzes required are:

$$u_{l,m}^{+1} = e^{j(\beta_x lh + \beta_y mh + wk)}$$
$$u_{l,m}^{-1} = e^{j(\beta_x lh + \beta_y mh - wk)}$$

and so the expression for error, E, may now be written as:

$$E = \int_{\theta} |\sum_{l,m} a_{l,m} e^{j(\beta_x lh + \beta_y mh)} + \sum_{l,m} b_{l,m} e^{j(\beta_x lh + \beta_y mh + wk)} + \sum_{l,m} c_{l,m} e^{j(\beta_x lh + \beta_y mh - wk)} - r|^2 \delta\theta \tag{3.35}$$

where $b_{l,m}$ and $c_{l,m}$ are the coefficients of the grid points to be applied during the next and previous time steps respectively.

To proceed, the derivative of E is taken with respect to particular values of all sets of coefficients resulting in a set of three expressions. Again, the magnitude is re-expressed such that the product rule may be applied to find the derivatives. The case of $\frac{\partial E}{\partial a_{l^*,m^*}}$ is developed below in order to show the method through which the systems of equations were constructed. Development of the remaining two derivatives is omitted for brevity.

$$\begin{aligned}\frac{\partial E}{\partial a_{l^*,m^*}} = \int_{\theta} e^{-j(\beta_x l^* + \beta_y m^*)h} [\sum_{l,m} a_{l,m} e^{j(\beta_x l + \beta_y m)h} + \sum_{l,m} b_{l,m} e^{j((\beta_x l + \beta_y m)h) + wk)} + \\ \sum_{l,m} c_{l,m} e^{j((\beta_x l + \beta_y m)h) - wk)} - r] + e^{j(\beta_x l^* + \beta_y m^*)h} [\sum_{l,m} a_{l,m} e^{-j(\beta_x l + \beta_y m)h} + \\ \sum_{l,m} b_{l,m} e^{-j((\beta_x l + \beta_y m)h) + wk)} + \sum_{l,m} c_{l,m} e^{-j((\beta_x l + \beta_y m)h) - wk)} - r] \delta\theta\end{aligned} \tag{3.36}$$

Multiplying out the expressions in equation (3.36) and collecting the terms of each set of coefficients yields:

$$\begin{aligned}\frac{\partial E}{\partial a_{l^*,m^*}} = \int_{\theta} \sum_{l,m} a_{l,m} (e^{j(\beta_x (l-l^*) + \beta_y (m-m^*))h} + e^{-j(\beta_x (l-l^*) + \beta_y (m-m^*))h}) \\ + \sum_{l,m} b_{l,m} (e^{j((\beta_x (l-l^*) + \beta_y (m-m^*))h + wk)} + e^{-j((\beta_x (l-l^*) + \beta_y (m-m^*))h + wk)}) \\ + \sum_{l,m} c_{l,m} (e^{j((\beta_x (l-l^*) + \beta_y (m-m^*))h - wk)} + e^{-j((\beta_x (l-l^*) + \beta_y (m-m^*))h - wk)}) \\ - r(e^{-j(\beta_x l^* + \beta_y m^*)h} + e^{j(\beta_x l^* + \beta_y m^*)h}) \delta\theta\end{aligned} \tag{3.37}$$

As in the single time step case, the exponential terms reduce to cosines and the derivative is set equal to zero in order to minimise the error. The resulting expression describes a set of equations that contributes to the overall system initialised in equation (3.35):

$$\int_{\theta}\sum_{l,m} a_{l,m}cos(A)+\sum_{l,m} b_{l,m}cos(B)+\sum_{l,m} c_{l,m}cos(C)\delta\theta = \int_{\theta} rcos((\beta_x l^* + \beta_y m^*)h)\delta\theta \tag{3.38}$$

where,

$$\begin{aligned} A &= ((\beta_x(l-l^*)+\beta_y(m-m^*))h \\ B &= ((\beta_x(l-l^*)+\beta_y(m-m^*))h+wk \\ C &= ((\beta_x(l-l^*)+\beta_y(m-m^*))h-wk \end{aligned}$$

Taking the derivative of equation (3.36) with respect to b_{l^*,m^*} gives:

$$\int_{\theta}\sum_{l,m} a_{l,m}cos(A)+\sum_{l,m} b_{l,m}cos(B)+\sum_{l,m} c_{l,m}cos(C)\delta\theta = \int_{\theta} rcos((\beta_x l^* + \beta_y m^*)h+wk)\delta\theta \tag{3.39}$$

and now,

$$\begin{aligned} A &= ((\beta_x(l-l^*)+\beta_y(m-m^*))h-wk \\ B &= ((\beta_x(l-l^*)+\beta_y(m-m^*))h \\ C &= ((\beta_x(l-l^*)+\beta_y(m-m^*))h-2wk \end{aligned}$$

Lastly, taking the derivative with respect to c_{l^*,m^*} gives,

$$\int_{\theta}\sum_{l,m} a_{l,m}cos(A)+\sum_{l,m} b_{l,m}cos(B)+\sum_{l,m} c_{l,m}cos(C)\delta\theta = \int_{\theta} rcos((\beta_x l^* + \beta_y m^*)h-wk)\delta\theta \tag{3.40}$$

this time,

$$\begin{aligned} A &= ((\beta_x(l-l^*)+\beta_y(m-m^*))h+wk \\ B &= ((\beta_x(l-l^*)+\beta_y(m-m^*))h+2wk \\ C &= ((\beta_x(l-l^*)+\beta_y(m-m^*))h \end{aligned}$$

Equations (3.38 - 3.40) together form the system of equations which were computed in order to produce coefficients for a cardioid polar response pattern (see section 5.1.2). The full system is of a similar form to that of the previous case and is shown graphically in figure (4).

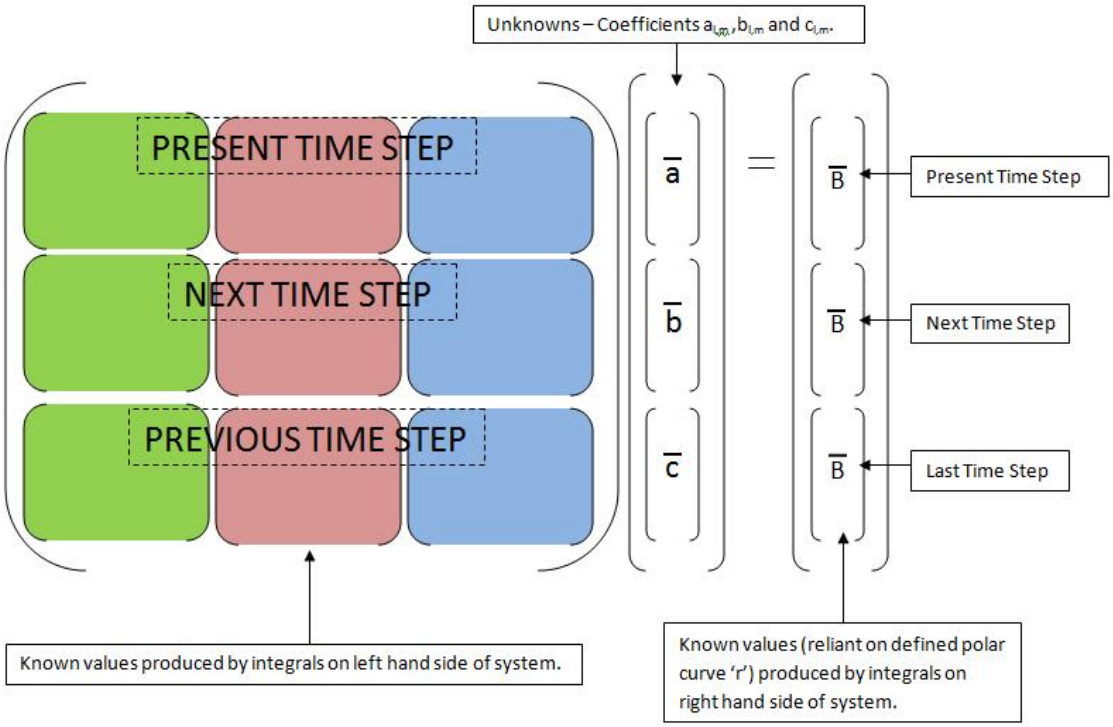

Figure (4): *System of equations for the solution using multiple time steps depicted in matrix form where: the green matrices signify the integrals formed by taking the derivative with respect to* a_{l^*,m^*}*; the red with respect to* b_{l^*,m^*}*; the blue with respect to* c_{l^*,m^*}.

The MATLAB code constructed to calculate the unknowns in this system is described in section 4.3.

3.3 Fixing Response Patterns at Multiple Frequencies

In practice, the sensitivity pattern for a directional microphone will vary across different frequency octaves. For example, a cardioid microphone will become increasingly directional (biased towards reception to sounds incoming from directly in front) as fre-

quency increases. It was decided that this characteristic would be built into the virtual receivers. To do so, the intial problem statement was altered accordingly:

$$E = \int_{\theta} |\sum_{w}\sum_{l,m} a_{l,m} e^{j(\beta_x(\theta,w)lh+\beta_y(\theta,w)mh)} - \sum_{w} r(\theta,w)|^2 \delta\theta \tag{3.41}$$

where, $w = w_1, w_2, ..., w_n$. Equation (3.41) states that a single set of coefficients, $a_{l,m}$, must be calibrated such that the error between the sum of defined and approximate curves, across all frequency bands (w), reaches a minimum. By following the same mathematical procedures as in section 3.2, the system of equations may be derived:

$$\sum_{w}\sum_{l,m} a_{l,m} \int_{\theta} cos(\beta_x(\theta,w).(l-l^*)h + \beta_y(\theta,w).(m-m^*)h)\delta\theta = \sum_{w} \int_{\theta} r(\theta,w).cos(\beta_x(\theta,w)l^*h + \beta_y(\theta,w)m^*h)\delta\theta \tag{3.42}$$

which is of the form:

$$\sum_{l,m} a_{l,m} K_{l^*,m^*,l,m,w_1} + \sum_{l,m} a_{l,m} K_{l^*,m^*,l,m,w_2} + ... + \sum_{l,m} a_{l,m} K_{l^*,m^*,l,m,w_n} = b_{l^*,m^*,w_1} + b_{l^*,m^*,w_2} + ... + b_{l^*,m^*,w_n} \tag{3.43}$$

Figure (5) shows the system of equations in matrix form. As with the intial problem, which is defined at a single frequency only, this case may be extended to include multiple time steps.

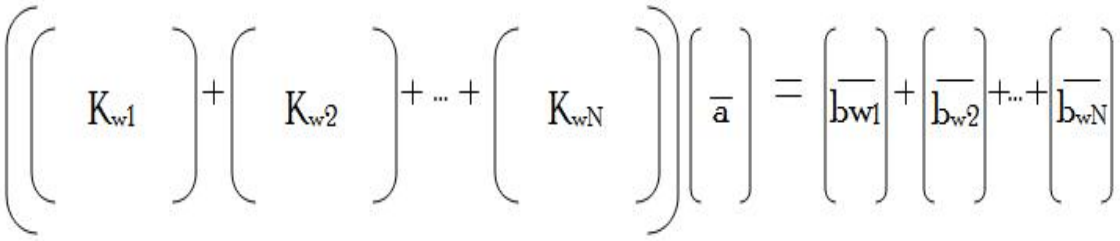

Figure (5): *The system of equations utilised to find unknowns, $a_{l,m}$, which in this case are set to provide differing polar patterns as frequency increases.*

3.4 Modelling 3D Response Patterns

The 2-Dimensional case may be extended to 3 dimensions by including a further spatial component. To begin, the wave number contributions must be re-expressed such that they suit the spherical coordinate system, i.e:

$$\beta_z = (\frac{w}{c})cos(\phi)$$
$$\beta_y = (\frac{w}{c})sin(\theta)sin(\phi)$$
$$\beta_x = (\frac{w}{c})cos(\theta)cos(\phi)$$

where, β_z, β_y and β_x are contributions from the z,y and x directions respectively. Angle θ remains the angle of orientation around the azimuth plane and the additional angle, ϕ, is the angle of elevation around the zenith (defined in the range $0^o - 180^o$). With these values defined, the 3-Dimensional least squares expression can be constructed.

$$E = \int_R \int_\phi \int_\theta [(|\sum_{l,m,n} a_{l,m,n} e^{j(\beta_x lh + \beta_y mh + \beta_z nh)} - r|)^2 R^2 sin(\phi)] \delta\theta\delta\phi\delta R \tag{3.44}$$

Equation (3.44) is of exactly the same form as that given for the 2D case, with the exception that the process of integration determines volume, not area, and must be weighted appropriately. The weighting terms R and $sin(\phi)$ (where R is the magnitude from the origin to a given point) are standard for ensuring that the correct volume is found when integrating over a spherical domain.

It is trivial to re-express the above equation in a manner similar to all previously noted cases, such that the coefficients, $a_{l,m,n}$, may be deduced by applying a linear system solve.

Chapter 4

Implementation

This chapter is primarily concerned with the description of MATLAB script files that were constructed in order to calculate the unknown coefficients in the derived receiver schemes. Each case is examined individually and excerpts of code are provided to clarify the different methods of implementation. These implementations were subject to several revisions over the course of the project. The examples given here are the most recent, condensed versions produced. Full MATLAB scripts are available for reference both in the appendices and on the accompanying CD.

4.1 Numerical Methods

Prior to providing details of the code produced, two particular numerical methods, employed in each MATLAB scipt, warrant brief explanation.

4.1.1 Integration

The basic trapezium rule was utilised to calculate the integrals in each system of equations. This process of numerical integration, in its simplest form, involves dividing equally the area confined by a curve in an approximate manner using rectangles. The area of these rectangles is then summated giving the approximate area. Figure (6) displays this approach.

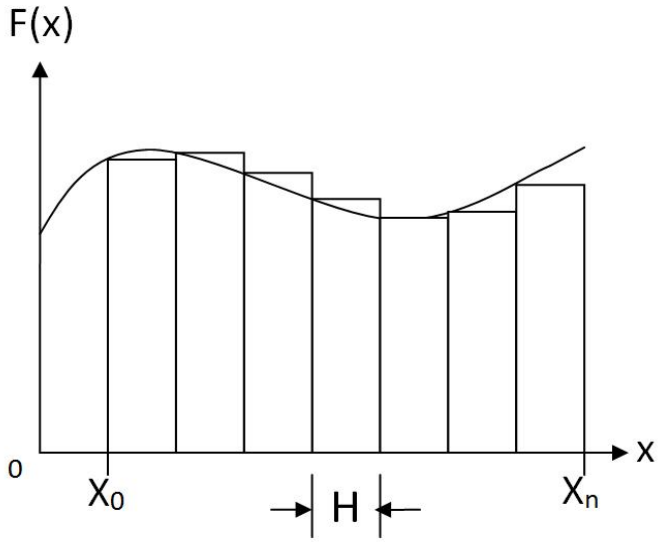

Figure (6): *The trapezium rule applied to an example function* $f(x)$.

When considering the area bounded by the function $f(x)$ and the x-axis, in the interval $x_0 - x_n$, a mathematical expression for such a value may be written as:

$$Area = \int_{x_0}^{x_n} f(x)\delta x \tag{4.1}$$

As shown in figure (6), this expression can be approximated using the summation of n trapezia. Assuming that, for every value of x, there is a known corresponding value of $f(x)$, this approximate area may be written as:

$$Area = \frac{H}{2}(f(x_0)+f(x_1)) + \frac{H}{2}(f(x_1)+f(x_2)) + ... + \frac{H}{2}(f(x_{n-1})+f(x_n)) \tag{4.2}$$

and therefore,

$$Area = \frac{H}{2}(f(x_0)+2f(x_1)+2f(x_2)+2f(x_3)+...+2f(x_{n-1})+f(x_n)) \tag{4.3}$$

where,

$$H = \frac{x_n - x_0}{n} \tag{4.4}$$

In order to compute the value of an integral in MATLAB, a simple vector implementation of this rule was utilised. Each function to be integrated was defined in the range

$\theta = [0 - 2\pi]$ and stored as a vector of length 'res'. The spacing between each value was, therefore, $H = \frac{2\pi}{res}$. A second vector, 'trapvec' (also length 'res'), was set such that all elements were equal to 2 with the exception of the first and final element. The trapezium rule could then be implemented by multiplying 'trapvec' by H and then carrying out vector multiplication of the result and the function whose enclosed area was to be found.

4.1.2 Least Squares Method

A number of approaches to solving a system of equations of the form $Ka = b$ in a least squares sense are available. As a starting point, the Moore-Penrose psuedo-inverse approach, detailed in [9], was studied. The psuedo-inverse of the matix K may be found using the formula:

$$K_{pinv} = (K^T K)^{-1} K^T \tag{4.5}$$

or simply computed using the MATLAB command 'pinv(K)'. Upon finding the psuedo-inverse, the unknown values a may be determined:

$$a = (K^T K)^{-1} K^T b \tag{4.6}$$

Applying this technique to a simple polynomial curve fitting problem exposed the fact that this approach is computationally inefficient. More importantly, if K is not full rank (i.e. not all rows are linearly independent), use of the psuedo-inverse yields unreliable results. During the scheme development stage, it became obvious that the design matrix K would never be full rank due to the expressions used to construct it. For the purposes of accuracy and efficiency, the MATLAB backslash command was employed to find the unknown terms. Upon execution of this command, MATLAB analyses the design matrix and discerns the best approach for solution.

4.2 2D Bi-Directional Receiver

The MATLAB script developed to solve the system described in section 3.2 is detailed concisely here.

4.2.1 Set Parameters

A list of the set parameters required within this script is given below.

• The sample rate, 'SR' - set to audio rate: 44100 Hz.
• The wave speed, 'c' - set to an approximate value for the speed of sound: 340 m/s.
• The test solution, or ansatz, relies on the definition of a test frequency, 'f', that can be altered as appropriate. This parameter is then converted into angular frequency, w, in rad/s.
• It was decided that the code would be generalised such that a variable, 'p', could be used to alter the number of grid points that contribute to the approximated response. The receiver was always implemented using a square grid of points of side length p.
• A variable 'res' sets the resolution, or number of points, used to define the target polar curve.

4.2.2 Derived Parameters

Several parameters and variables used within the MATLAB script are reliant on the set parameters. These include the time step, k, which is defined as the reciprocal of the sampling frequency. Although this particular scheme operates over a single time step, k must still be calculated in order to define the discrete spatial step size, or grid spacing, 'h'. This parameter is calculated as per the stability condition given in section 3.1.5.
Two vectors are initialised: one to enable integration through use of the trapezium rule; the other to define angles 'θ' of which there are 'res' equally spaced values ranging between $0^o - 360^o$. Using the vector of angles (named 'thetavec'), the wave speed and the analysis frequency, the wave number contributions (β_x and β_y) may also be calculated and stored as vectors. The bi-directional target polar curve, denoted by 'r', is defined by inserting the angle vector into the function given in equation (3.23).

4.2.3 Calculation of Integrals

Recall the form of the system of equations for the case over a single time step:

$$\sum_{l,m} a_{l,m} K_{l^*,m^*,l,m} = b_{l^*,m^*}$$

The term K takes the form of a matrix whose elements are the areas under the functions of θ generated using every possible permutation of l, m, l^* and m^* (as per equation (3.33)). On the left hand side, the term b is a vector containing the areas of each curve produced using every permutation of l^* and m^*. Having first defined the appropriate parameters, nested loops are employed to fill the initialised vector 'b'. This is implemented using the following code:

```
for lstar = low:high
for mstar = low:high
counter = counter + 1
b(counter) = (r.*cos(((betax*lstar)+(betay*mstar))*h))*trapvec;
end
end
```

The loop counters 'low' and 'high' are set to $-((\frac{p+1}{2})-1)$ and $((\frac{p+1}{2})-1)$ respectively. This means that, provided the parameter 'p' is odd, the function inside the loops will take in all combinations of l^* (lstar) and m^* (mstar), that, together, define the positioning of a grid point with respect to the receiver position. The function calculates each area by firstly multiplying, element by element, the defined curve, 'r', with the cosine function. The trapezium rule is then applied through vector multiplication with a pre-defined vector, 'trapvec', that is of the same form as that described in section 4.1.1. The location of each area within vector 'b' is set by a counter, initialised to zero, which increments by one on each iteration.
The matrix 'K' is constructed through similar means using the following code:

```
lstar = low:high
mstar = low:high
rcounter = rcounter+1;
ccounter = 0;
l = low:high
m = low:high
```

```
ccounter = ccounter+1;
K(rcounter,ccounter) = cos((betax*(l-lstar) + betay*(m-mstar))*h)*trapvec;
end
end
end
end
```

In this case, four nested loops are used to ensure all combinations of lstar, mstar, l and m are considered by the main function. Two counters,'rcounter' (rows) and 'ccounter' (columns), are first initialised to zero and then both increment appropriately to set the location of each area in matrix K. Note that the column counter resets upon completion of the two interior loops. Again, a vector multiplication of the cosine function and 'trapvec' ensures that the integration is performed.

4.2.4 Calculating Scheme Coefficients

A vector of the unknown coefficients, 'a', can be easily calculated by simply applying the backslash operator to matrix K and vector b. In MATLAB, this is achieved using a single line of code,

```
a = K\b;
```

The backslash command performs a linear system solve to find coefficients suitable for creating a bi-directional receiver. Results gained from this MATLAB script are noted later in the report.

4.3 2D Cardioid Receiver

In order to construct and solve the system of equations, which are derived using multiple time steps, the MATLAB script for the previous case was extended. All setup and derived parameters remain the same. A notable difference between the two scripts is the size of matrix K and vector b and their contents. Due to the fact that, in this case past, present and next time steps contribute to the output response, both K and b are three times as large. As noted in section 3.2.4, the construction of vector b relies on three separate equations as shown below:

```
for lstar = low:high
for mstar = low:high
counter = counter +1;
b(counter) = (r.*cos(((betax*lstar)+(betay*mstar))*h))*trapvec;
b((p^ 2)+counter) = (r.*cos(((betax*lstar)+(betay*mstar))*h + w*k))...
*trapvec;
b(2*(p^ 2)+counter) = (r.*cos(((betax*lstar)+(betay*mstar))*h - w*k))...
*trapvec;
end
end
```

The value p^2 refers to the total number of grid points that contribute to the overall response at each time step. Using this term in conjunction with the incrementing variable, 'counter', ensures that three areas are calculated and placed in vector b at the correct positions during every iteration. Note that the first p^2 elements of b correspond to the present time step, the next block of p^2 elements relate to the next time step and the final elements relate to the last time step (in accordance with the matrix form depicted in figure (4)).
In this case, the matrix K is defined using nine equations and can be constructed through similar means as those employed in the single time step case. Again, the value p^2 is applied alongside both column and row counters such that the areas calculated on each iteration are appropriately positioned within the matrix. The code for this case is omitted here for brevity. Upon determining the known values in K and b, the backslash operator may again be employed to generate the desired vector of coefficients for a cardioid receiver.

4.4 2D Receivers Fixed at Multiple Frequencies

The method of extending the scripts previously described, such that they attempt to approximate a number of polar patterns at varying frequencies, is detailed in this section. Referring back to figure (5), it is apparent that the matrices K_w and vectors b_w may be calculated using exactly the same approaches as those previously described. The choice of using a single time step or multiple time step solution relies on the nature of the target polar pattern to be approximated. The scripts for both cases may be

repeated for each pair of analysis frequency and defined polar curves as described in the following pseudo-code:

```
Initialise set and derived parameters
Define vector of analysis frequencies and related target polar curves
Initialise arrays, K, Knext, Kprev, b, bnext, bprev to store values
during each loop (across analysis frequencies)
for (Loop over analysis frequencies and corresponding polar curves)
Calculate wave number contributions depending on current analysis
frequency
Calculate vector b using previously detailed method
Store b as bprev
Define bnext = b + bprev
Construct matrix K
Store K as Kprev
Define Knext = K + Kpev
Reset counters as appropriate
end Terminate once all analysis frequencies have been considered
Generate coeffs 'a' using a = Knext\bnext
End of File
```

The full MATLAB script is omitted here for brevity but is available for reference in the appendices.

4.5 2D FDTD Scheme with an Embedded Receiver

This section is dedicated to describing the computation of the 2D wave equation and highlights the proposed means of inserting a directional receiver into the simulation.

4.5.1 Set and Derived Parameters

The set parameters for this scheme included the sampling rate (SR), wave speed (c), time step (k), grid side length (l = 1) and the run-time (NF). Wave speed, sampling rate and time step were all assigned the same value as those given in the receiver schemes

to ensure continuity. The derived parameters such as the grid spacing and the courant number, λ, were calculated in accordance with equations (3.20) and (3.13) respectively. In order to aquire the number of grid points, 'dim', needed to cover a required side length the following equation was used:

```
dim = floor(l/h);
```

The above expression finds an integer amount of grid points by applying the MATLAB 'floor' function which rounds a value down to the nearest whole number. The grid spacing 'h' is subsequently reset as $\frac{l}{dim}$. Coefficients which appear within for loops were also pre-computed and defined as variables in order to lessen computation times.

4.5.2 Defining the Surface Grid

In MATLAB, a 2-Dimensional grid of points may be represented by a matrix where each element corresponds to a grid point. A simple square grid topology was chosen for this implementation and, as such, the required matrix was of size (M x M). Two matrices of this form were initialised. The first stores the values from all grid points at any position in time. The second stores the values present one time step previous as per the requirements of the recursive schemes. A third matrix was initialised to take on the values occuring one time step ahead and thus, through pointer swapping, it was possible to effectively pass data from one matrix to another as appropriate upon an increment in time.

4.5.3 Source and Receiver Positioning

Source and receiver positions were defined by using percentages of the distance along both the x and y axes. In order to implement this in the MATLAB script file, expressions of the following form were used:

```
isrcx = floor(srcx*dim)
```

where‘isrcx’ is the distance along the x axis at which the source is placed. ‘srcx’ is a pre-defined decimal value corresponding to a percentage of the length of the x axis. By multiplying srcx by dim and rounding the value to the nearest integer, the resulting value relates to a grid row. The coordinates of both the source and receiver positions were calculated in this manner. The receiver coordinates were used to localise the central point of the grid point array that was needed to produce the directional response. To excite the virtual acoustic field, the initial value of the grid point at the source position was set to 1. In doing so, an impulse was placed on the grid.

4.5.4 Implementing the Finite Difference Scheme for Interior Points

Equation (3.12) was placed within a for loop that executed upon each increment in time from zero to the run-time value, NF. Two further for loops were nested within the main time loop in order to update all grid values in both the x and y direction as per the recursive update scheme, i.e:

```
for y = 2:dim
for x = 2:dim
u(y,x) = coeff1*(u1(y-1,x)+u1(y+1,x)+ u1(y,x-1)+ u1(y,x+1))+...
coeff2*u1(y,x)-u2(y,x);
end
end
```

where coeff1 and coeff2 are λ^2 and $2(1-2\lambda^2)$ respectively. It is worth stating here that the loop counters over x and y increment from 2 to dim as these values correspond to the interior points of the grid. The elements in the first and last rows and columns of the matrix correspond to the grid points that lie on a boundary. As such, the side length used for each model can be quantified as $(dim+1)h$.

4.5.5 Implementing Fixed Boundaries

The condition for fixed boundaries, discussed in section 3.1.4, states that the terminating edges of a surface may not move. Under such constraints, the grid points that lie on a boundary within the model must remain at a value of zero. The initial values

of all grid points (matrix elements) are set to zero when the grid matrix is initialised. Therefore, in order to implement fixed boundaries, the interior points must be updated as described in the previous section, leaving the boundary points 'fixed' to a value of zero.

4.5.6 Acheiving Directional Receiver Sensitivity

During simulation of the 2D wave equation, it is possible to record the fluctuations of the acoustic field by storing the values of the matrix element $u1(irecx, irecy)$ in a vector of size NF. This allows the sound arriving at the receiver position to be captured. For a directional receiver, the fluctuations of a square array of grid points (of side length 'p') centered on the receiver position must be gained. In order to store these values, an array of size p^2 x NF is used. The rows of this array will then correspond to the output recorded at each of the grid points. Each output can then be multiplied by the corresponding pre-calculated weighting coefficient computed using the schemes described previously. Summating these weighted outputs produces an overall output that should possess the directional characteristics defined during the calculation of the coefficients. Note that this approach relates to the production of a bi-directional response pattern as only a single time step (stored in matrix u1) is considered. Extending this simple method to the case of multiple time steps is made trivial through use of matrices u and u2.

4.6 3D Receivers

The development of directional receivers for use in 3-Dimensional virtual acoustic models was abandoned for the following reasons:

• The integral weighting term R (see equation (3.44)) is very difficult to quantify such that the volume enclosed by each function is correctly calculated.
• In MATLAB, 3-Dimensional polar plots are not available and, therefore, the solids defined in spherical coordinates could not be visualised. Securing a graphical means of interpreting the behaviour of a function is highly advantageous when attempting to solve a problem.
• Unexpected errors were exposed during rudimentary testing of the 2D receiver schemes.

Ultimately, it was decided that the 2D cases required further analysis before attempting to extend implementation to 3D.

Chapter 5

Results

The results presented in this chapter demonstrate the successful production of directional response patterns at defined analysis frequencies. In order to verify the correctness of the coefficients produced by each scheme, the resulting approximate polar curves were visually compared with the defined curves using plotting functions available within MATLAB. A simple testing framework was devised in order to analyse the behaviour of the receiver response across a range of frequencies.

5.1 2D Receiver Schemes at a Single Frequency

As previously described, the weighting coefficients $a_{l,m}$ were to be multiplied with the output response of each point of the receiver grid array. This process was rendered possible in the MATLAB code for both the single and multiple time step cases. In doing so, the resultant approximate polar response pattern could be plotted. Examples of these plots are given here alongside plots of the defined polar curves as a means of comparison.

5.1.1 Bi-Directional

Figure (7) shows the single time step scheme developed in the MATLAB script "bidir.m" (see appendix or accompanying CD) converging on a bi-directional response pattern. A total of 9 grid points, arranged in a 3 x 3 array, were employed to produce the approximate polar pattern (shown right) at an analysis frequency of 250 Hz.

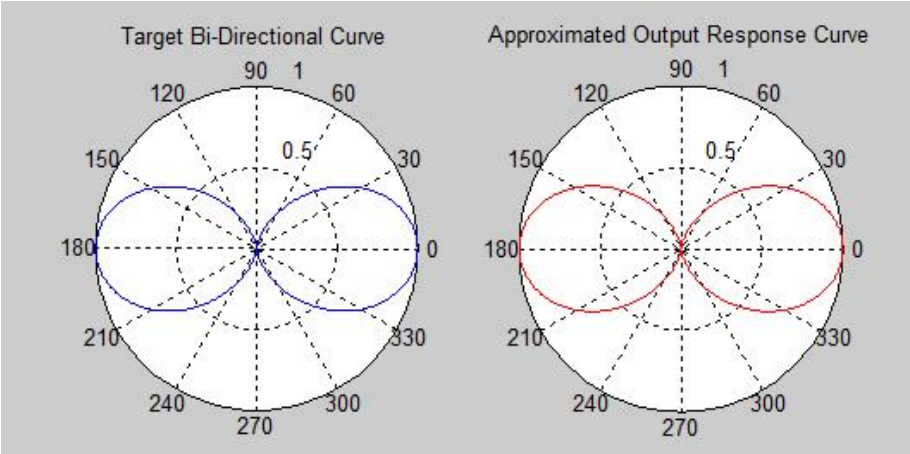

Figure (7): *Output of MATLAB script "bidir.m". Comparison between the defined polar plot 'r' (left) and the calculated approximation (right) is displayed.*

The diagram above shows that the desired response characteristics have been effectively modelled at the analysis frequency. When attempting to approximate a cardioid pattern using this scheme, the result shown in figure (8) is observed.

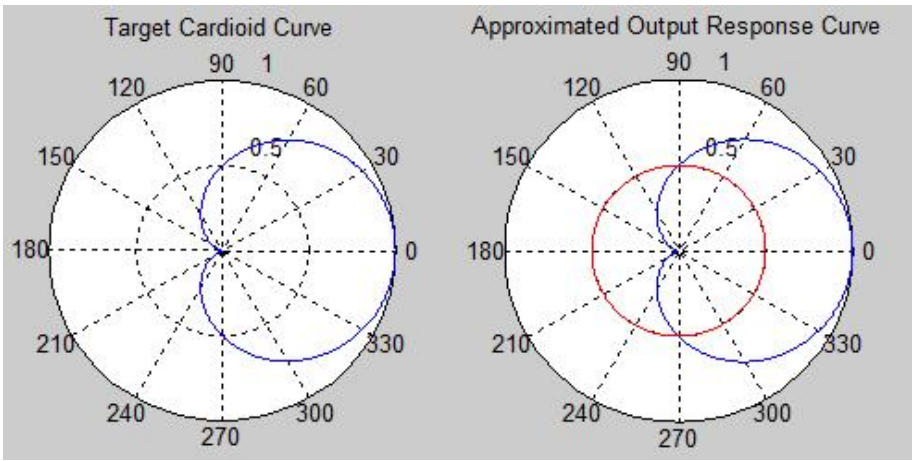

Figure (8): *Plot of the poorly approximated curve yielded when applying the single time step scheme to a target curve that is symmetrical about one axis only (right). The defined cardioid curve is shown left.*

The above plots show that the single step scheme was incapable of calculating suitable coefficients to render such a curve. It was thought that by increasing the number of grid points in the receiver array, the approximated curve would become more ideal. This, however, was not the case and hence the multiple time step scheme was developed to overcome this issue.

5.1.2 Cardioid

The script "cardir.m", also available in the appendix, was written to implement the multiple time step scheme derived in section 3.2.4. This scheme was shown to be capable of accurately modelling a 2D cardioid receiver response sensitivity pattern at any chosen analysis frequency. Figure (9) displays the resultant output when using an analysis frequency of 250 Hz.

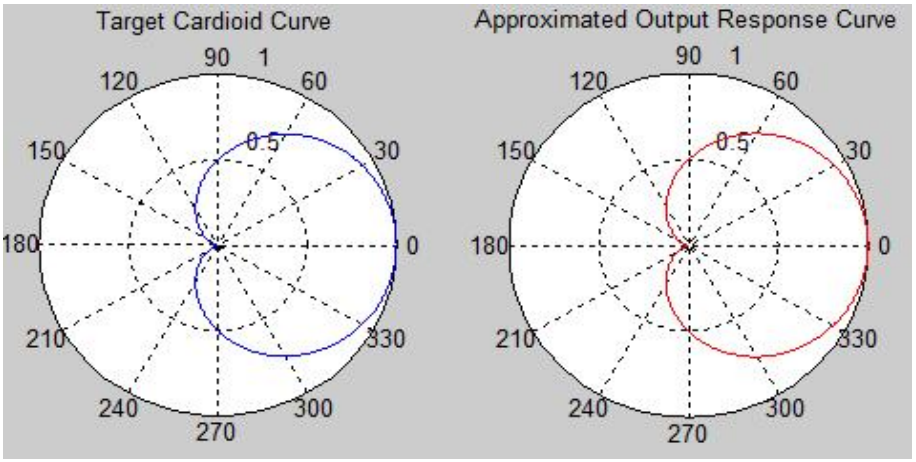

Figure (9): *Output of MATLAB script "cardir.m". Comparison between the defined polar plot 'r' (left) and the calculated approximation (right) is displayed.*

Again, a promising result is gained as the coefficients calculated within the scheme produce an exact replica of the defined cardioid pattern. In the previous bi-directional case and this case, the target polar curve was created using a function of θ. Figure (10) depicts the multiple time step scheme converging on a cardioid-like pattern with an increasing number of grid points used.

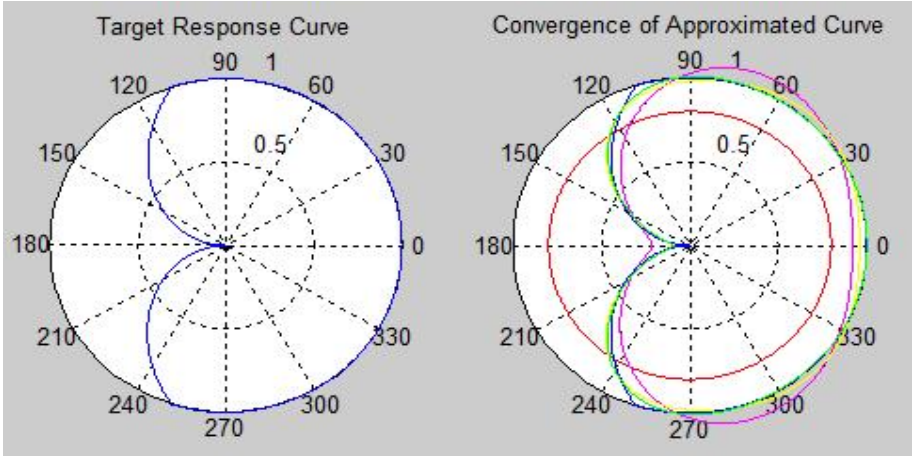

Figure (10): *Convergence of approximate polar curve as the number of grid points employed by the scheme is increased (shown left).*

In figure (10), the target curve displayed was create by developing a matrix whose elements roughly described a cardioid polar curve. This was carried out in order to display the effect of increasing the number of grid points and, therefore, the number of coefficients used to converge on a given curve. The coloured plots on the right hand side correspond to the number of coefficients applied as follows: red - 1 coeff.; magenta - 81 coeffs.; yellow - 121 coeffs.; green - 169 coeffs. The success of this process shows that schemes are also capable of converging to non-ideal target curves (i.e. those used to describe the response of microphones at higher frequencies).

5.2 2D Receiver Schemes at Multiple Frequencies

The code developed in accordance with the documentation provided in section 4.4 sought to produce coefficients such that the approximated polar curve would match target curves given at different frequencies. The MATLAB script entitled “freqvar.m” implements this scheme yielding the result given in figure (11).

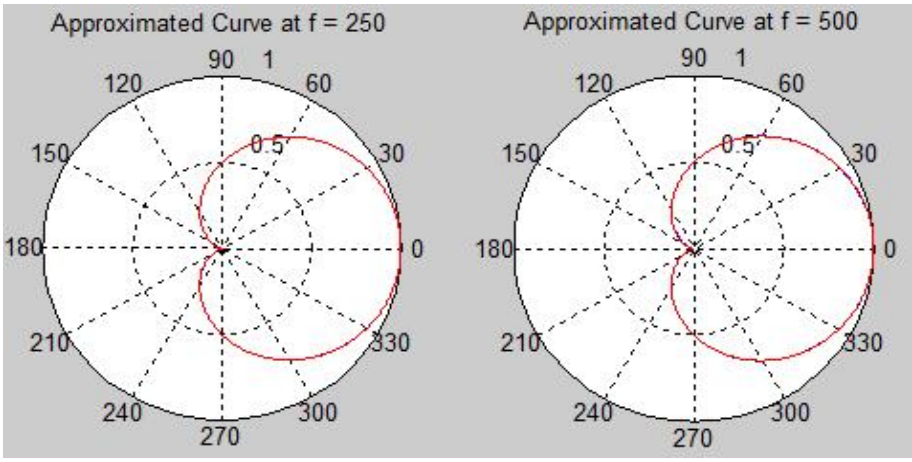

Figure (11): *An example of an output gained from "freqvar.m". As shown the scheme converges on a cardioid polar pattern defined at two frequencies.*

For the case portrayed in figure (11), 25 grid points/coefficients and two analysis frequencies (250 and 500 Hz) were used. The scheme successfully converges on the target cardioid curve at both frequencies. The reader is invited to access the script via the accompanying CD and experiment with varying coefficient and frequency parameters.

5.3 Initial Testing

The implementation of the 2D wave equation with an embedded receiver was utilised for a simple test procedure. This test involved exciting the acoustic field at source locations positioned at angles of $0^o, 90^o, 180^o$ and 270^o around the receiver array. The impulse response generated using each source and receiver combination was then examined to ascertain whether the amplitude of the first reading corresponded to the nature of the polar curve used. It can be assumed that, for a bi-directional receiver, the response to sound incoming from 0^o and 180^o would be equal and greater than that to sound incoming from 90^o and 270^o. However, the results gathered suggested that the receiver was equally receptive to sound incoming from all directions and, therefore, the directional sensitivity was not being modelled. Similar results were gained when attempting to apply a cardioid directionality.

In order to investigate this issue more thoroughly, the coefficients calculated by each scheme were employed to weight the summation of grid point responses across a range of frequencies. The MATLAB scripts "freqtest.m" and "freqtestts.m" provided

a means of testing the coefficients calculated using the single time step and multiple time step schemes respectively. Figure (12) shows coefficients generated using a 5x5 grid array in the single time step scheme during such a test.

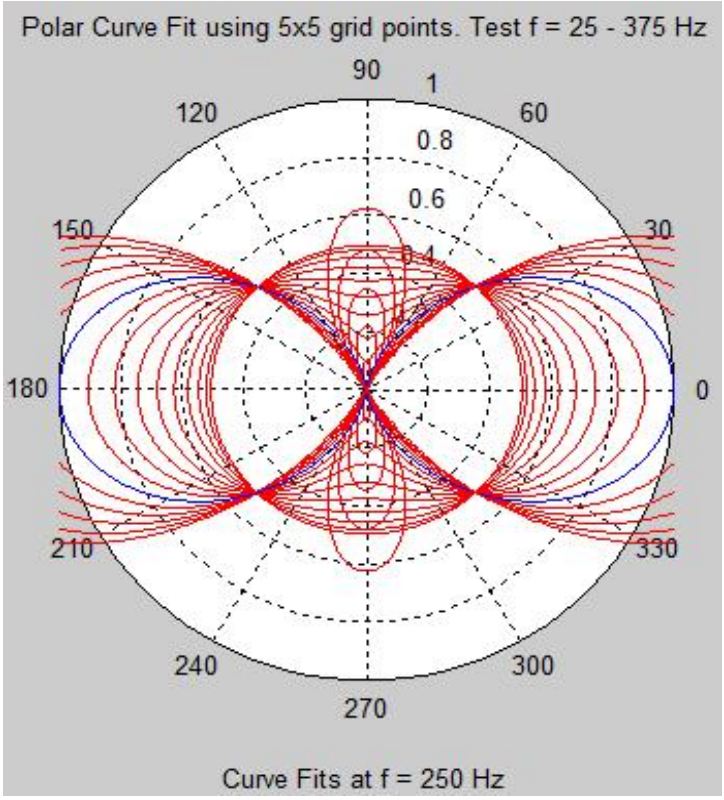

Figure (12): *The behaviour of an approximated response curve using set coefficients as frequency increases.*

The coefficients were calibrated using an analysis frequency of 250 Hz and the test applied a range of frequencies between 25 and 375 Hz (increasing in increments of 25). It was observed that when the test procedure reached analysis frequency, the approximate polar curve converged to the defined bi-directional pattern. However, at low frequencies an omni-directional response is generated. At frequencies higher than the analysis frequency, the approximated polar curve begins to develop lobes at angles 90^o and 270^o. If the frequency is allowed to increased further, these lobes become increasingly large until the response pattern has lost all definition. Similar results were gained through testing of the cardioid scheme.

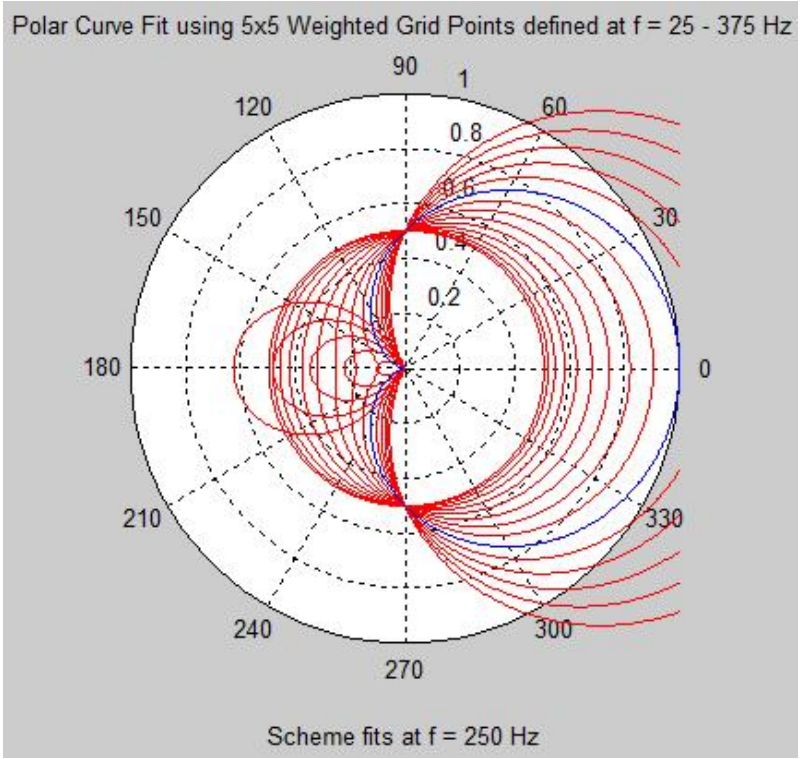

Figure (13): *The results of the testing procedure as applied to the coefficients generated within the multiple time step scheme to produce a cardioid polar curve.*

As in the case of the bi-directional pattern, the approximate curves shown in figure (13) possess omni-directional characteristics. As the test frequency increases towards the analysis frequency, the resulting polar curve approaches the defined cardioid pattern. At the analysis frequency of 250 Hz, the target and approximate responses match to a high level of accuracy. As the test frequency increases further, a lobe begins to develop at angle 180^o.

The observations gained from this testing procedure provide a reason for the embedded receiver to behave in such a way that directionality is not preserved. Both examples show that each set of coefficients produces a response pattern which varies dramatically as frequency increases. At high frequencies, the lobes that develop in both cases become larger and larger until the response pattern is effectively omni-directional.

In response to the exposition of this issue, the scheme that seeks to fix polar patterns at a range of frequencies was also tested in the same manner. Figure (14) shows the results of this test.

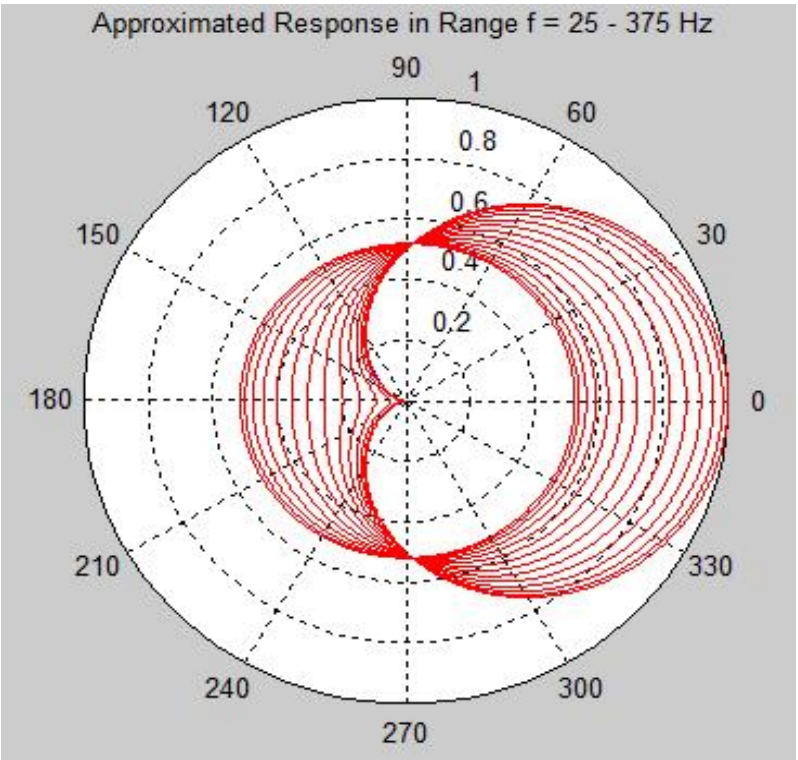

Figure (14): *Plot of the approximated polar curve (created through application of coefficients generated using "freqvar.m") converging on the target pattern as frequency increases.*

Figure (14) displays an improvement on the results gained through use of the single and multiple time step schemes. In this case, the weighting coefficients are initially tailored to attempt to fit a given target plot at a number of frequencies using the code outlined in section 4.4. As displayed, unwanted lobes are removed from the approximate polar curve such that directionality is preserved. However, at frequencies lower than that of the highest analysis frequency, the polar pattern approximation is constrained to approach the shape of the defined curve, not match it. Therefore, this method is slower to converge on the desired response characteristics as frequency increases.

Chapter 6

Conclusions

The work documented in this project report serves as an intial step towards attempting to define and solve a problem that, up until now, has remained untouched in the field of FDTD virtual acoustics research. It is unfortunate that many of the aims, as first proposed, have not been attained due to timing constraints and unforseen issues. However, these studies have exposed outcomes and results that will prove useful in further attempts to model FDTD directional receivers. This report closes with a summary of project outcomes and proposed routes of further work.

6.1 Conclusions Drawn from Results

Currently, the schemes devised have been shown to be capable of approximating two common directional polar patterns through the weighted sum of an array of output responses from a 2D FDTD model. Through various experiments conducted using the MATLAB scripts compiled, conclusive evidence has been gathered to suggest that directional response characteristics may be attained at all audible frequencies. The main concern with the schemes developed is that they produce output coefficients that are suited to weighting model outputs at a single frequency only. At frequencies above and below this defined analysis frequency the resulting polar plots have been shown to be far from ideal. This is extremely problematic when attempting to gain a directional impulse response from an FDTD model.

The scheme developed in section 4.4 generated slightly improved results. This is due to the fact that the weighting coefficients are averaged over a defined number of differ-

ent frequencies. The scheme will, therefore, attempt to fit the defined polar curve to the target curve across a range of frequencies and so the creation of unwanted (directional) lobes is avoided. However, this improvement comes at a cost. The polar patterns achieved at all frequencies lower than the highest defined frequencies approach the target curve, but do not match it. As displayed in the previous section, the target curve is matched exactly only at the highest frequency.
The reasons behind the behaviour of the schemes could stem from a number of potential factors. Firstly, the design matrix K, used in each case, is defined never to be full rank. Essentially, this results in the weighting coefficients being approximated with less than the optimal amount of information. Secondly, the intial problem statement may be ill-conditioned such that the round-off error introduced by machine precision has a profound effect on the devised schemes. Lastly, the least squares approach is but one of many techniques that could be employed to calculate the weighting coefficients: it is possible that a method of optimization, more suited to the problem, is available.

6.2 Level of Success in Attaining Project Aims

Developing a means of modelling receivers with directional sensitivity characteristics has been partially achieved. The schemes created render possible the production of receivers that possess directionality, but only at one or two predefined frequencies. Over a range of frequencies, directional characteristics may still be preserved, however, the directionality is less accurate than initially intended. Due to these outcomes, extension to 3D schemes and, therefore, the production of receivers to be used for auralization processes was unattainable. As a result, the surround sound sonification of FDTD virtual models is now an appropriate goal of proposed further work.

6.3 Suggested Further Work

Potential routes of further study are listed below:

• Obtain numerical values that describe the nature of the response of either a bi-direction or cardioid microphone at a number of frequencies. This will provide further insight into the characteristics of a real receiver type and may help to discern the fol-

lowing: whether the expectation of the schemes is realistic; whether the true response of a microphone may be roughly mimicked.

• Research the possibility of achieving better numerical conditioning within the schemes in their current format.

• Devise a new means of optimization to calculate the required weighting coefficients. Another approach may yield similar, or even, improved results.

• Continue testing the schemes to establish, precisely, the tolerance margins that must be allowed for when using the devised approaches to generate directional receivers.

• Continue work on the problem in 3-Dimensions in order to approach the main, initially proposed, aims of this project.

Bibliography

[1] D. Murphy and M. Beeson, "St Michael's Cathedral - 'Old' Coventry Cathedral," in *[Online Resource] http://www-users.york.ac.uk/ dtm3/virtualacoustics.html*, Last Accessed: Aug. 2012.

[2] S. Bilbao, *Numerical Sound Synthesis: Finite Difference Schemes and Simulation in Musical Instruments*, John Wiley and Sons, Chichester, UK, 2009.

[3] Z. Karabiber, "A New Approach to an Ancient Subject: CAHRISMA Project," in *[Online Resource] http://www.eng.um.edu.mt/ pjmica/research/cahrisma/cahrisma.htm*, Last Accessed: Aug. 2012.

[4] A. Farina, "Ramsete - a new pyramid tracer for medium and large scale acoustic problems," in *Proceedings of the EURO-NOISE 95 Conference*, Lyon, France, 1995.

[5] L. Savioja T. Rinne T. Takala, "Simulation of Room Acoustics with a 3-D Finite Difference Mesh," in *Proceedings of Int. Computer Music Conf.*, Aarhus, Denmark, 1994, pp. 463–466.

[6] A. Farina and R. Ayalon, "Recording Concert Hall Acoustics for Posterity," in *Proceedings of 24th AES Conference on Multichannel Audio*, Banff, Canada, 2003.

[7] M. Kleiner B. Dalenbäck P. Svensson, "Auralization - An Overview," *Audio Engineering Society*, vol. 41(11), pp. 861–875, 1993.

[8] I. J. Hewitt, "Math 257: Finite Difference Methods," in *[Online Resource] http://www.math.ubc.ca/ hewitt/math257/fd.pdf*, Last Accessed: Aug. 2012.

[9] C. Moler, "Numerical Computing With MATLAB," in *[Online Resource] http://www.mathworks.co.uk/moler/chapters.html*, Last Accessed: Aug. 2012.

[10] D. J. Wilde C. S. Beightler, *Foundations of Optimization*, Pretince - Hall, Inc., NJ, USA, 1993 edition, 1967.

[11] C. J. Webb and S. Bilbao, "Computing Room Acoustics with CUDA - 3D FDTD Schemes with Boundary Losses and Viscosity," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP*, Prague, Czech Republic, 2011.

Appendices