Theses and Dissertations

2005-05-23

Development of an Energy-Based Nearfield Acoustic Holography System

Michael C. Harris
Brigham Young University - Provo

Follow this and additional works at: https://scholarsarchive.byu.edu/etd

Part of the Mechanical Engineering Commons

BYU ScholarsArchive Citation
https://scholarsarchive.byu.edu/etd/325

This Thesis is brought to you for free and open access by BYU ScholarsArchive. It has been accepted for inclusion in Theses and Dissertations by an authorized administrator of BYU ScholarsArchive. For more information, please contact scholarsarchive@byu.edu, ellen_amatangelo@byu.edu.
DEVELOPMENT OF AN ENERGY-BASED NEARFIELD ACOUSTIC HOLOGRAPHY SYSTEM

by

Michael C. Harris

A thesis submitted to the faculty of

Brigham Young University

Master of Science

Department of Mechanical Engineering

Brigham Young University

August 2005
This thesis has been read by each member of the following graduate committee and by majority vote has been found to be satisfactory.

Date

Jonathan D. Blotter, Chair

Date

Craig C. Smith

Date

Scott D. Sommerfeldt
As chair of the candidates graduate committee, I have read the thesis of Michael C. Harris in its final form and have found that (1) its format, citations, and Bibliographical style are consistent and acceptable and fulfill university and department style requirements; (2) its illustrative materials including figures, tables, and charts are in place; and (3) the final manuscript is satisfactory to the graduate committee and is ready for submission to the university library.

Date

Jonathan D. Blotter
Chair, Graduate Committee

Accepted for the Department

Matthew R. Jones
Graduate Coordinator

Accepted for the College

Alan R. Parkinson
Dean, Ira A. Fulton College of Engineering and Technology
ABSTRACT

DEVELOPMENT OF AN ENERGY-BASED NEARFIELD ACOUSTIC HOLOGRAPHY SYSTEM

Michael C. Harris
Department of Mechanical Engineering
Master of Science

Acoustical-based imaging techniques have found merit in determining the behavior of vibrating structures. These techniques are commonly used in numerous applications to obtain detailed noise source information and energy distributions on source surfaces. This thesis focuses on the continued development of the nearfield acoustic holography (NAH) approach. Conventional NAH consists of first capturing pressure data on a two-dimensional conformal measurement contour in the nearfield of the radiating source. These data are then propagated back to the vibrating structure to obtain the normal velocity profile on the source surface. With the source surface velocity profile known, the acoustic pressure, particle velocity, and intensity generated by the source can be reconstructed anywhere in space. The precision of source reconstruction is reliant upon accurate measurement of the pressure field at the hologram surface. For complex acoustic fields this requires fine spatial resolution and therefore demands large
microphone arrays. In this thesis, a technique is developed for performing NAH using energy-based measurements. Recent advancements in the area of acoustic sensing technology have made particle velocity field information more readily available. Because energy-based measurements provide directional information about the field, a more accurate measurement of the pressure field is obtained. It is proposed that an energy-based system will significantly reduce the number of measurements required to perform NAH without sacrificing accuracy. Significantly reducing the number of measurements required to perform NAH will reduce the time, and therefore the expense, of using NAH as an analysis tool.

Many potential applications exist for an improved NAH measurement method in the automobile and aerospace industries. These industries provide numerous large-scale applications where employing time-consuming scanning methods is not cost-effective. This is especially the case for airplane in-flight passenger noise tests, where the expense of operating the airplane is extremely high. Therefore, even a small savings in data acquisition time would be very beneficial.
ACKNOWLEDGEMENTS

I would like to thank all those who have helped me in my efforts and contributed in any way to the completion of this thesis. I first would like to recognize my advising professor and committee chair, Jonathan Blotter. Some of the most difficult concepts relating to this research became clear only after long discussions in his office. His excitement about the project and encouragement helped me to carry on when difficult obstacles were encountered. I also wish to thank the other two members of my graduate committee, Craig Smith, and Scott Sommerfeldt. They proved to be invaluable resources in helping me understand many of the important intricacies of this research.

These acknowledgments would not be complete if I did not mention the support of my wife, Laura. She has spent many late evenings home alone without a single complaint as I toiled away. Her constant encouragement and confidence in me have sustained me throughout the journey of this project. Finally, I also must direct my thanks heavenward for inspiration received along the way.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>1.1</td>
<td>PROBLEM STATEMENT</td>
<td>1</td>
</tr>
<tr>
<td>1.2</td>
<td>OBJECTIVE AND GOALS</td>
<td>4</td>
</tr>
<tr>
<td>1.3</td>
<td>HYPOTHESIS</td>
<td>4</td>
</tr>
<tr>
<td>1.4</td>
<td>THESIS OUTLINE</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>BACKGROUND AND LITERATURE REVIEW</td>
<td>7</td>
</tr>
<tr>
<td>2.1</td>
<td>FUNDAMENTALS IN FOURIER ACOUSTICS</td>
<td>7</td>
</tr>
<tr>
<td>2.1.1</td>
<td>Fundamental Equations</td>
<td>7</td>
</tr>
<tr>
<td>2.1.2</td>
<td>Fourier Transform</td>
<td>12</td>
</tr>
<tr>
<td>2.1.3</td>
<td>Inverse Problems</td>
<td>13</td>
</tr>
<tr>
<td>2.1.4</td>
<td>Plane Waves</td>
<td>14</td>
</tr>
<tr>
<td>2.1.5</td>
<td>Cylindrical Waves</td>
<td>15</td>
</tr>
<tr>
<td>2.2</td>
<td>PARTICLE VELOCITY MEASUREMENT</td>
<td>18</td>
</tr>
<tr>
<td>2.2.1</td>
<td>Pressure Gradient Method</td>
<td>18</td>
</tr>
<tr>
<td>2.2.2</td>
<td>Microflown Sensor</td>
<td>19</td>
</tr>
<tr>
<td>3</td>
<td>NEARFIELD ACOUSTIC HOLOGRAPHY</td>
<td>23</td>
</tr>
</tbody>
</table>
3.1 OVERVIEW

3.2 SEPARABLE GEOMETRIES
   3.2.1 Planar Geometry
   3.2.2 Cylindrical Geometry

3.3 STATIONARY SOURCES

3.4 TRANSIENT PROBLEMS

3.5 NAH MEASUREMENTS
   3.5.1 Finite Aperture Effects
   3.5.2 Spatial Sampling
   3.5.3 Wavenumber Filter
   3.5.4 Measurement Noise and the Standoff Distance

4 ENERGY-BASED RECONSTRUCTION PROCEDURES
   4.1 OVERVIEW
   4.2 HERMITE INTERPOLATION
      4.2.1 Curves
      4.2.2 Surfaces

5 ANALYTICAL IMPLEMENTATION
   5.1 SYNTHETIC FIELD CREATION
   5.2 ERROR EVALUATION
   5.3 ANALYTICAL RESULTS

6 EXPERIMENTAL VALIDATION
   6.1 PLANAR TEST CASE
      6.1.1 Setup
      6.1.2 Data Acquisition
      6.1.3 Data Post Processing
      6.1.4 Results
   6.2 CYLINDRICAL TEST CASE
      6.2.1 Setup
      6.2.2 Data Acquisition
      6.2.3 Data Post Processing
      6.2.4 Results
   6.3 DISCUSSION OF RESULTS
LIST OF TABLES

Table 6-1  Comparison Between Analytical and Actual Modal Frequencies ........... 60
Table 6-2  Knowles Microphone Calibration Parameter Values .......................... 64
Table 6-3  Microflown Element Calibration Parameter Values ............................ 65
LIST OF FIGURES

Figure 1-1 Energy-based NAH System Development ............................................. 5
Figure 2-1 Analogy Between Temporal Frequency and Wavenumber .................... 9
Figure 2-2 Differential Volume Element ............................................................... 10
Figure 2-3 Scanning Electron Microscope Image of Microflown Element ............... 19
Figure 2-4 Microflown Cross-sectional Temperature Profile .................................. 20
Figure 2-5 Directional Sensitivity of Microflown Sensor ....................................... 21
Figure 2-6 Microflown Amplitude and Phase Correction Plots: a) Sensitivity and b) Phase ................................................................. 22
Figure 3-1 Fourier NAH Algorithm Flowchart ..................................................... 25
Figure 3-2 Valid Region of the Homogeneous Wave Equation .............................. 26
Figure 3-3 Tukey Window Taper for $x_0 = 1$ and $L_x/2 = 2.0$ ............................ 31
Figure 3-4 Wavenumber Filter Shape for Three $\alpha$ Values with $k_c = 20$ .......... 33
Figure 4-1 Comparison of Conventional and Energy-Based NAH Procedures ....... 36
Figure 4-2 Cubic Hermite Basis Functions in Parameter Space .............................. 40
Figure 4-3 Corner Points Defining a Surface Patch .............................................. 46
Figure 4-4 Sample Bicubic Hermite Surface Patch with Required Inputs Shown.... 48
Figure 4-5 Complex Surface Composed of Nine Bicubic Hermite Patches ............ 48
Figure 5-1 Flowchart Illustrating Analytical Implementation ................................ 50
Figure 5-2 Simply Supported Plate Chosen as the Source for the Analytical Modeling ................................................................. 51
Figure 5-3 Description of the Geometric Quantities in Rayleigh's Integral .......... 52
Figure 5-4  Illustration of Error Evaluation Method ................................................... 53
Figure 5-5  Reference Pressure at the 2 cm Estimation Plane........................................... 54
Figure 5-6  Estimated Pressure from Conventional NAH with a 20 x 20 Grid of Pressure Measurements at 5 cm................................................................. 55
Figure 5-7  Pressure Error at the Estimation Plane............................................................ 56
Figure 5-8  Comparison of the Estimation Error for Conventional and Energy-based NAH Reconstructions ........................................................................ 57
Figure 6-1  Planar Experimental Setup:  a) Aluminum Plate in Steel Frame with Wood Frame and Sensor Shown and b) 20 mm Diameter Piezoelectric Patch Used to Excite Plate .......................................................... 60
Figure 6-2  USP Sensor Images:  a) USP Sensor and b) Close Up of the Three Orthogonal Microflown Elements ........................................................................ 61
Figure 6-3  Knowles FG Series Electret Condenser Microphone ........................................ 61
Figure 6-4  Experimental Measurement Grid ................................................................ 63
Figure 6-5  Measured Pressure Field at 5 cm ................................................................. 66
Figure 6-6  Measured Pressure Gradient Field in x-direction at 5 cm ............................... 66
Figure 6-7  Measured Pressure Gradient Field in y-direction at 5 cm ............................... 67
Figure 6-8  Pressure Field at 5 cm after Hermite Interpolation Using the Full 11 x 17 Measurement Grid .............................................................................. 68
Figure 6-9  Windowed Pressure Field at 5 cm with $x_w$ at 0.15 m .................................. 69
Figure 6-10 Windowed and Zero-padded Pressure Field at the Measurement Plane... 69
Figure 6-11 Wavenumber Domain Magnitudes .............................................................. 70
Figure 6-12 Magnitudes of Wavenumber Domain Components after Propagation to the 2 cm Estimation Plane ................................................................. 71
Figure 6-13 Magnitudes of Wavenumber Domain Components after Propagation and Filtering at the 2 cm Estimation Plane ....................................................... 72
Figure 6-14 Reconstructed Pressure Field at 2 cm from a Measurement at 5 cm Using ENAH ........................................................................................................ 73
Figure 6-15 Measurement Grid Locations:  a) Locations Used in Conventional NAH Reconstruction and b) Locations Used in ENAH Reconstruction ... 74
Figure 6-16 Reference Pressure at the 2 cm Estimation Plane................................. 75

Figure 6-17 Reconstructions: a) from 11 x 17 Array of Pressure Measurements Using NAH and b) from 6 x 9 Array of Energy-based Measurements Using ENAH................................................................. 75

Figure 6-18 Pressure Error: a) for NAH Reconstruction and b) for ENAH Reconstruction ............................................................................................................................... 76

Figure 6-19 Cylinder Test Setup..................................................................................... 78

Figure 6-20 Measurement Grid Locations: a) Locations Used in Conventional NAH Reconstruction and b) Locations Used in ENAH Reconstruction .. 80

Figure 6-21 Reference Pressure at 2 cm Estimation Contour....................................... 80

Figure 6-22 Reconstructions: a) from 11 x 14 Array of Pressure Measurements Using NAH and b) from 7 x 7 Array of Energy-based Measurements Using ENAH........................................................................ 81

Figure 6-23 Pressure Error: a) for NAH Reconstruction and b) for ENAH Reconstruction ............................................................................................................................... 81


1 INTRODUCTION

This chapter presents background and objectives related to this thesis. Explanation is given of the main limitations of nearfield acoustic holography (NAH) that are addressed, as well as the goals specific to this research. The remaining chapters of this thesis are also outlined.

1.1 PROBLEM STATEMENT

Over the past century, acoustic imaging has become an invaluable tool in a variety of disciplines. A significant push in underwater imaging began with the sinking of the Titanic in 1912. Reginald A. Fessenden\(^1\), who worked for the Submarine Signal Company, patented the first functioning echo ranging device in the United States in 1914. The crude device consisted of an electric oscillator that emitted a low-frequency noise and then switched to a receiver to listen for echoes. It was able to detect an iceberg underwater from 2 miles away.

In 1928, Sergei Y. Sokolov proposed an acoustic imaging technique for flaw detection in metals. He demonstrated that sound waves could be used as a new form of microscope, based on a reflective principle. In the reflection technique, a pulsed sound wave is transmitted from one side of the sample, reflected off the far side, and returned to a receiver located at the starting point. Upon impinging on a flaw or crack in the material, the signal is reflected and its traveling time altered. The actual delay becomes a measure
of the flaw's location. A map of the material can then be generated to illustrate the location and geometry of the flaws. The same technique, with few modifications and refinements, is now used in medical tomography to acquire high resolution images of organs and tissues within the body.

Since around 1950 when holography was first developed, it has become an increasingly powerful research tool. Acoustic holography first appeared in the mid 1960s. It refers to the calculation of acoustic quantities in three-dimensional space from a two-dimensional pressure measurement. The acoustic holography equations were formulated at the same time as the laser-based optical holography equations. Acoustic holography measurements were made in the farfield, more than an acoustic wavelength removed from the sources. With measurements being made in the farfield, the wavelength of the radiation limits the spatial resolution of the reconstruction. Because of this, only source details greater than the acoustic wavelength can be retrieved. This means, for example, that two point sources cannot be resolved if they are separated by less than a wavelength. In optics this limitation does not pose any difficulties since the optical wavelengths are so small. However, in acoustics, where a large class of long wavelength radiators (vibrating machinery, musical instruments, etc.) are considerably smaller than the radiated wavelength, this limitation prevents the localization of features that are crucial to understanding the vibration of the structure. Therefore, conventional acoustic holography is only an approximation to the inverse problem of reconstructing sound fields on source surfaces.

In 1980, Maynard and Williams developed an experimental measurement technique called nearfield acoustic holography (NAH) which eliminated the
wavelength resolution limit. This measurement technique has revolutionized experimental acoustics. NAH is a methodology that enables the reconstruction of the normal velocity components on the surface of a vibrating object in three-dimensional space based on the acoustic pressure measured on a two-dimensional conformal surface in the “nearfield”. In the nearfield of a sound source, the acoustic field is composed of two kinds of waves: propagating waves that are organized and radiate out from the source; and evanescent waves that decay exponentially from the source surface and do not carry energy into the farfield. By making nearfield measurements, and therefore capturing the evanescent waves before they have decayed completely, localized information about the sources is obtained that allows for sub-wavelength source resolution. Because NAH provides complete acoustic information cost effectively, it has drawn a great deal of attention in the diagnostics of noise and vibration sources\textsuperscript{7,8}.

As is common with most measurement techniques, NAH has inherent characteristics that allow or limit its use in certain types of applications. Traditional NAH requires nearfield measurements to be made on a conformal surface enclosing the sources. This may be accomplished using a conformal microphone array. The precision of source reconstruction is reliant upon accurate measurement of the pressure field at this hologram surface. Complex acoustic fields that require fine spatial resolution demand large microphone arrays. Precision pressure microphones are expensive, making large two-dimensional arrays cost prohibitive in many cases. Most users are also limited by the number of channels of data that can be acquired simultaneously. If the operating conditions of the test object are steady state, an alternative to large arrays is to scan the sound pressure field over a conformal contour using a smaller sub-array of microphones.
The scans are phase-locked using stationary reference transducers. This approach can provide very fine resolution of the acoustic field but is time intensive and introduces additional system control and data post-processing complexity.

Many potential applications exist for an improved NAH measurement method in the automobile and aerospace industries. These industries provide numerous large-scale applications where employing time-consuming scanning methods is not cost-effective. This is especially the case for airplane in-flight passenger noise tests, where the expense of operating the airplane is extremely high. Therefore, even a small savings in data acquisition time would prove to be very beneficial.

1.2 OBJECTIVE AND GOALS

The objective of this research is to develop and construct a NAH system using energy-based measurements. This method will subsequently be referred to as Energy-Based Nearfield Acoustic Holography (ENAH). The primary goal associated with this objective is to increase the cost-effectiveness of NAH by reducing the number of required measurements. A secondary goal is to reduce the channel count for NAH systems.

1.3 HYPOTHESIS

The two thesis goals will be accomplished through the steps outlined in Figure 1-1. This illustration divides the development process into five sub steps. The first step is to check the sensor calibration data provided by the manufacturer to ensure that particle velocity measurements can be made with sufficient accuracy. The second step is to develop the vibrating source that will be investigated. This will be done both analytically and experimentally for this study. In the third step, the data is acquired. For the
analytical case, this is accomplished by sampling the synthetic field at the chosen measurement locations. Post processing of the data is carried out in step four. The velocity field information from the measurement is incorporated into the pressure field using Hermite surface interpolation\(^9\). With the interpolation completed, the final step is to apply an NAH algorithm to reconstruct the pressure field at the desired locations.

![Energy-based NAH System Development](image)

**Figure 1-1**  Energy-based NAH System Development

### 1.4 Thesis Outline

The remainder of this thesis describes the developed procedures in detail along with the validations performed. Chapter 2 presents the background necessary to
understand NAH methods and concepts, along with recent advancements in particle velocity sensing capabilities utilized in this research. Chapter 3 introduces current NAH reconstruction methods including an in depth presentation of the Fourier-based NAH method that is used for this study. Chapter 4 provides a detailed description of the proposed energy-based reconstruction procedure, including relevant theory and background. Chapter 5 describes an analytical implementation of ENAH, where the procedure is applied to the theoretical field generated by a simply supported rectangular plate driven at its center. The error evaluation method chosen for comparison between ENAH and traditional NAH reconstructions is also introduced. Chapter 6 presents two test cases chosen for experimental validation of the analytical results. Conclusions for this research are made in Chapter 7 as well as recommendations for possible further work in this area. Chapter 8 provides a list of references and Chapter 9 is the Appendix, which includes other information relevant to recreating the experimental and analytical results presented.
2 BACKGROUND AND LITERATURE REVIEW

This chapter presents an overview of fundamental equations and theory relating to acoustic field reconstruction. Background for the area of Fourier acoustics is presented. Two methods for measuring particle velocity are also introduced.

2.1 FUNDAMENTALS IN FOURIER ACOUSTICS

The behavior of sound is first discussed, since it serves as a starting point for acoustics in general. The Fourier transform is presented since it is at the foundation of NAH method to be implemented. The inherent problems associated with acoustic inverse problems are introduced, which are important to understanding the process of back propagation to the vibrating surface from a measurement in the field.

2.1.1 Fundamental Equations

Acoustic vibrations in both fluids and solid structures essentially involve the propagation of wave motion through a supporting media. Fahy\textsuperscript{10} describes a mechanical wave as a “phenomenon in which a physical quantity propagates in a supporting medium, without the net transport of the medium”. This propagation may be characterized kinematically as relative displacements from positions of equilibrium of the particles of the supporting medium together with the speed and direction of propagation. The general
complex exponential form of a simple harmonic wave $g(x,t)$ traveling in the positive $x$-direction is given by

$$g^+(x,t) = \text{Re} \left\{ \tilde{B} e^{i \left( \omega t - \frac{\omega x}{c} \right)} \right\} \quad (2.1)$$

where $\tilde{B}$ is the complex wave amplitude, $\omega$ the circular frequency, and $t$ the time. For a wave of the same amplitude traveling in the negative $x$-direction, $g(x,t)$ takes the form,

$$g^-(x,t) = \text{Re} \left\{ \tilde{B} e^{i \left( \omega t + \frac{\omega x}{c} \right)} \right\} \quad (2.2)$$

where $c$ is termed the phase velocity of the wave. This term is used because an observer traveling in the direction of wave propagation at this speed sees no change of phase. The spatial period of a simple harmonic wave is described by its wavelength $\lambda$. However, the spatial variation may better be expressed by a quantity that represents phase change per unit distance. This quantity, called the wavenumber, is symbolized by the letter $k$ and has the dimension of reciprocal length.

$$k = \frac{\omega}{c} = \frac{2\pi}{\lambda} \quad (2.3)$$

The wavenumber is actually the magnitude of a vector that indicates the direction of propagation as well as the spatial phase variation. This quantity is of vital importance in the study of two and three-dimensional wave fields. The analogy between temporal frequency $\omega$ and spatial frequency $k$ is illustrated in Figure 2-1. Just as any form of temporal variation can be analyzed into a spectrum of complex frequency components,
any form of spatial variation can be Fourier analyzed into a spectrum of complex wavenumber components.

![Analogy Between Temporal Frequency and Wavenumber](image)

**Figure 2-1 Analogy Between Temporal Frequency and Wavenumber**

The wave equation governing the propagation of small disturbances through a homogeneous, inviscid, compressible fluid may be written in Cartesian coordinates \((x, y, z)\) in terms of the acoustic pressure \(p\) as

\[
\frac{\partial^2 p}{\partial x^2} + \frac{\partial^2 p}{\partial y^2} + \frac{\partial^2 p}{\partial z^2} - \frac{1}{c^2} \frac{\partial^2 p}{\partial t^2} = 0
\]

(2.4)

where \(c\) is the frequency independent speed of sound of the medium. The right hand side of the equation indicates that there are no sources in the volume in which the equation is valid. The wave equation is derived from linearized forms of the continuity and momentum equations.
Another important equation, called Euler’s equation, provides a useful relationship between pressure and particle velocity. The following derivation of Euler’s equation is also helpful in developing an understanding of the physical meaning of acoustic pressure and particle velocity. Figure 2-2 shows an infinitesimal volume element of fluid $\Delta x \Delta y \Delta z$. The force arrows indicate the direction of force for positive pressure.

![Differential Volume Element](image)

**Figure 2-2  Differential Volume Element**

All six faces of the cube experience forces due to the pressure in the fluid. The following is the convention for pressure:

$$p > 0 \rightarrow \text{Compression}$$

$$p < 0 \rightarrow \text{Rarefaction}.$$  

At a specific point in a fluid, a positive pressure indicates that an infinitesimal volume surrounding the point is under compression, and forces are exerted outward from this volume. It follows that if the pressure at the left face of the cube in Figure 2-2 is positive, then a force will be exerted in the positive $x$-direction of magnitude $p(x, y, z) \Delta y \Delta z$. The pressure at the opposite face $p(x + \Delta x, y, z)$ is exerted in the negative $x$-direction. The
force on the right face of the cube is obtained by expanding \( p(x + \Delta x, y, z) \) in a Taylor series to first order. The total force exerted on the volume in the \( x \)-direction is

\[
[p(x, y, z) - p(x + \Delta x, y, z)] \Delta y \Delta z = -\Delta x \Delta y \Delta z \frac{\partial p}{\partial x}
\] (2.5)

Using Newton’s equation in one dimension, \( f = ma = m \frac{\partial \dot{u}}{\partial t} \), where \( f \) is the force, \( m = \rho_0 \Delta x \Delta y \Delta z \), and \( \rho_0 \) is the fluid density, yields

\[
\rho_0 \frac{\partial \dot{u}}{\partial t} = -\frac{\partial p}{\partial x}
\] (2.6)

Carrying out the same analysis in the \( y \) and \( z \)-directions yields the following two equations.

\[
\rho_0 \frac{\partial \dot{v}}{\partial t} = -\frac{\partial p}{\partial y}
\] (2.7)

\[
\rho_0 \frac{\partial \dot{w}}{\partial t} = -\frac{\partial p}{\partial z}
\] (2.8)

Combining Eqs. (2.6)-(2.8) and using the gradient operator gives the compact form of Euler’s equation.

\[
\rho_0 \frac{\partial \vec{v}}{\partial t} = -\nabla \vec{p}
\] (2.9)

where \( \vec{v} \) represents the velocity vector with components \( \dot{u}, \dot{v}, \dot{w} \), as shown in Eq. (2.10).

\[
\vec{v} = \dot{u}\hat{i} + \dot{v}\hat{j} + \dot{w}\hat{k}
\] (2.10)
2.1.2 Fourier Transform

At the heart of Fourier acoustics is the Fourier transform. The Fourier transform is a method for the decomposition of a function \( f(x) \) into its amplitude and frequency components. The two-sided Fourier transform of \( f(x) \) is given by Eq. (2.11),

\[
F(k_x) = \int_{-\infty}^{\infty} f(x) e^{-ik_x x} dx
\]  \hspace{1cm} (2.11)

where \( k_x \) represents the wavenumber in the \( x \)-direction. Equation (2.12) describes the inverse Fourier transform of \( F(k_x) \). It suggests that given the amplitude-frequency spectrum of a function, the original function may be reconstructed.

\[
f(x) = \frac{1}{2\pi} \int_{-\infty}^{\infty} F(k_x) e^{ik_x x} dk_x
\]  \hspace{1cm} (2.12)

The Fourier transform is especially useful for solving the convolution-type problems that are commonly encountered in acoustic wave propagation such as Rayleigh’s integral. This is true because convolution in temporal space is equivalent to multiplication in the frequency domain. This property is expressed symbolically in Eq. (2.13).

\[
\int_{-\infty}^{\infty} \left[ \int_{-\infty}^{\infty} f(x-x') g(x') dx' \right] e^{-ik_x x} dx = F(k_x) G(k_x)
\]  \hspace{1cm} (2.13)

This convolution theorem is easily proven by changing the order of integration and applying the shift theorem to the left hand side of Eq. (2.13).
In Fourier acoustics, functions that vary in two-dimensions are generally of interest. The two-dimensional function \( f(x, y) \) has the two-dimensional Fourier transform given by Eq. (2.14). The inverse transform of \( F(k_x, k_y) \) is shown in Eq. (2.15).

\[
F(k_x, k_y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) e^{-i(k_x x + k_y y)} \, dx \, dy \quad (2.14)
\]

\[
f(x, y) = \frac{1}{4\pi^2} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} F(k_x, k_y) e^{i(k_x x + k_y y)} \, dk_x \, dk_y \quad (2.15)
\]

### 2.1.3 Inverse Problems

A major attraction of NAH is its solution to the inverse problem. The inverse problem backtracks the pressure and velocity fields in space and time towards the source surface. However, the solution for the reconstruction is unstable or non-unique because of the finite number of pressure field measurements, measurement noise, round-off errors, and mainly the presence of evanescent waves. Evanescent waves decay exponentially from the source. Therefore, when these waves are propagated back to the
surface they are amplified exponentially. Methods for obtaining a stable solution are presented in Chapter 3.

### 2.1.4 Plane Waves

The characteristic property of a plane wave is that each acoustic quantity has constant amplitude and phase on any plane perpendicular to the direction of propagation\(^\text{12}\). In finding the plane wave solutions of the wave equation, it is convenient to convert the wave equation to the frequency domain. This transform yields Eq. (2.16), the Helmholtz equation,

\[ \nabla^2 \bar{p} + k^2 \bar{p} = 0 \quad (2.16) \]

where the acoustic wavenumber is \( k = \omega/c \), \( \bar{p} \) is the function \( \bar{p}(x, y, z, \omega) \), and \( \nabla^2 \) is the Laplacian operator defined as

\[ \nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2} \quad (2.17) \]

The following solution for \( \bar{p} \)

\[ \bar{p}(\omega) = A(\omega) e^{i(k_x x + k_y y + k_z z)} \quad (2.18) \]

satisfies Eq. (2.16) as long as

\[ k^2 = k_x^2 + k_y^2 + k_z^2, \quad (2.19) \]

where \( A(\omega) \) is an arbitrary constant. Since \( k \) is a constant, the three wavenumbers on the right hand size of Eq. (2.19) are not independent of each other. The convention is to choose \( k_z \) as the dependent variable so that

\[ k_z^2 = k^2 - k_x^2 - k_y^2 \quad (2.20) \]
It is important to realize that the plane wave solution still satisfies the wave equation when \( k_x \) or \( k_y > k \), a condition under which the plane waves turn into non-propagating evanescent waves. When this is the case, \( k_z \) becomes

\[
k_z = \pm i \sqrt{k_x^2 + k_y^2 - k^2} = \pm ik'_z
\]

(2.21)

where \( k'_z \) is real and the plane wave, turned evanescent, has the form

\[
\vec{p} = Ae^{\pm k'_z z} e^{i(k_x x + k_y y)}
\]

(2.22)

If the sources exist in the half space defined by \( z < 0 \), then the \( e^{+k'_z z} \) solution is non-physical since it is not bounded at \( +\infty \). Discarding this solution, Eq. (2.22) becomes

\[
\vec{p} = Ae^{-k'_z z} e^{i(k_x x + k_y y)}
\]

(2.23)

This shows the exponential amplitude decay in \( z \) that is characteristic of evanescent waves.

### 2.1.5 Cylindrical Waves

The application of the wave equation in cylindrical coordinates leads to the solution of many problems of interest in acoustics. The vibration and radiation from submarines and aircraft fuselages are significant problems that conform closely to cylindrical symmetry.

Of interest are the solutions of the homogeneous, time-dependent wave equation

\[
\left( \nabla^2 - \frac{1}{c^2} \frac{\partial^2}{\partial t^2} \right) p = 0
\]

(2.24)

in cylindrical coordinates \((r, \phi, z)\). In Eq. (2.24), the Laplacian operator\(^{13}\) in cylindrical coordinates is defined as
\[ \nabla^2 = \frac{\partial^2}{\partial r^2} + \frac{1}{r} \frac{\partial}{\partial r} + \frac{1}{r^2} \frac{\partial^2}{\partial \phi^2} + \frac{\partial^2}{\partial z^2} \tag{2.25} \]

Solutions to the wave equation are obtained using separation of variables\(^{14}\). This method assumes that the solution can be written as a product of solutions of functions of each coordinate and time as shown in Eq. (2.26).

\[ p(r, \phi, z, t) = R(r)\Phi(\phi)Z(z)T(t) \tag{2.26} \]

Introducing this solution into Eq. (2.24) and dividing through by \(R\Phi Z T\) leads to

\[ \left( \frac{1}{R} \frac{d^2 R}{dr^2} + \frac{1}{rR} \frac{dR}{dr} + \frac{1}{r^2 \Phi} \frac{d^2 \Phi}{d\phi^2} \right) + \left( \frac{1}{Z} \frac{d^2 Z}{dz^2} \right) = \frac{1}{c^2 T} \frac{d^2 T}{dt^2} \tag{2.27} \]

Since \(r, \phi, z,\) and \(t\) are all independent, the separated terms must be equal to a constant for the equality to be satisfied. The arbitrary separation constants, \(k\) and \(k_z\), are chosen satisfying the following equations,

\[ \frac{1}{c^2 T} \frac{d^2 T}{dt^2} = -k^2 \tag{2.28} \]

\[ \frac{1}{Z} \frac{d^2 Z}{dz^2} = -k_z^2 \tag{2.29} \]

\[ \frac{1}{R} \left( \frac{d^2 R}{dr^2} + \frac{1}{rR} \frac{dR}{dr} \right) + \frac{1}{r^2 \Phi} \frac{d^2 \Phi}{d\phi^2} = -k^2 + k_z^2 \tag{2.30} \]

Defining an additional constant \(k_r\) as

\[ k_r = \sqrt{k^2 - k_z^2} \tag{2.31} \]

allows Eq. (2.30) to be written as

\[ \frac{r^2}{R} \left( \frac{d^2 R}{dr^2} + \frac{1}{rR} \frac{dR}{dr} \right) + k_r^2 r^2 = -\frac{1}{\Phi} \frac{d^2 \Phi}{d\phi^2} \tag{2.32} \]
Once again the right and left hand sides must be equal to constants for the equality to hold. Choosing \( n^2 \) as one of these constants leads to

\[
\frac{1}{\Phi} \frac{d^2 \Phi}{d\phi^2} = -n^2
\]

(2.33)

Substituting Eq. (2.33) into Eq. (2.32) yields the well known Bessel equation, Eq. (2.34).

\[
\frac{d^2 R}{dr^2} + \frac{1}{r} \frac{dR}{dr} + \left( k^2 - \frac{n^2}{r^2} \right) R = 0
\]

(2.34)

The solutions of Eq. (2.34) are known and are given by the Bessel functions of the first and second kinds, \( J_n(k,r) \) and \( Y_n(k,r) \). The index of the Bessel function corresponds to the separation constant in the \( \Phi \) equation.

A linear combination of these functions provides the traveling wave solution. These combinations are termed the Hankel functions of the first and second kind,

\[
H_n^{(1)}(k,r) = J_n(k,r) + iY_n(k,r)
\]

(2.35)

\[
H_n^{(2)}(k,r) = J_n(k,r) - iY_n(k,r)
\]

(2.36)

With time dependence \( e^{-i\omega t} \), \( H_n^{(1)}(k,r) \) corresponds to an outgoing wave and \( H_n^{(2)}(k,r) \) to an incoming wave. Combining the solutions to Eqs. (2.28), (2.29), (2.33), and (2.34) provides the general solution to the cylindrical wave equation.

\[
p(r,\phi,z,\omega) = \sum_{n=-\infty}^{\infty} e^{ink\phi} \frac{1}{2\pi} \int_{-\infty}^{\infty} A_n e^{i\omega z} H_n^{(1)}(k,r) + B_n e^{i\omega z} H_n^{(2)}(k,r) \, dk
\]

(2.37)
2.2 PARTICLE VELOCITY MEASUREMENT

The main difficulty of energy-based sensing lies in the ability to measure the acoustic particle velocity. Presently, the primary technique for particle velocity estimation is via finite difference approximations. Recently, a new particle velocity transducer known as a Microflown sensor has been developed which functions similar to a hot wire anemometer. The theories behind the pressure gradient method and the Microflown sensor are introduced below since the proposed reconstruction method is independent of the chosen acoustic particle velocity sensor. This study utilizes the Microflown sensor to measure the particle velocity.

2.2.1 Pressure Gradient Method

The finite difference method relies on two closely spaced pressure measurements to determine the particle velocity using Euler’s equation, \( \text{(2.9)} \). The convenient frequency domain form of Euler’s equation is obtained by taking the Fourier transform of Eq. (2.9),

\[
i \omega \rho_0 \vec{v} = \nabla \vec{p}
\]

where \( \omega \) is the angular frequency. This expression is convenient because often data are measured and processed as individual spectral components in the frequency domain. The particle velocity may then be directly computed with the pressure gradient, angular frequency, and fluid density known. The accuracy of this method depends on error in the pressure difference, scattering and diffraction of the sensor, and microphone phase mismatch.
2.2.2 Microflown Sensor

The Microflown transducer consists of two thin, parallel wires spaced five microns apart that are heated to approximately 300°C. Figure 2-3 shows a scanning electron microscope image of a Microflown element. This transducer has some resemblance to the well-known hot wire anemometer, but with at least two distinct differences. First, the Microflown has two wires making the sensor directional. In addition, the sensor is subjected to two heat transfer mechanisms, diffusion and heat convection, where an anemometer is only subject to convection.

![Figure 2-3 Scanning Electron Microscope Image of Microflown Element](image)

Heat transfer occurs as air particles flow across the two wires in the element. The first wire crossed will heat the air slightly which results in the second wire not being cooled to quite the same degree. Figure 2-4 illustrates a typical cross-sectional temperature profile for a Microflown element with a particle velocity applied across it. This temperature difference is used to determine the particle velocity. Jacobsen and de
Bree\textsuperscript{17} showed that results comparable to finite difference intensity approximations are possible using the Microflown sensor to measure the particle velocity.

Since particle velocity is a vector quantity, the sensor has a polar pattern relating to its directional sensitivity. The shape of this pattern is shown in Figure 2-5. Only half of the response is plotted because it is symmetric. This plot shows that if the sensor element is ±10° off perpendicular to the direction of particle velocity, the error will be approximately ±1.5%.

![Figure 2-4 Microflown Cross-sectional Temperature Profile](image)
The frequency response of the Microflown element is not flat. The sensitivity, $S_m$, and phase correction, $Phase_m$, for the sensor are governed by Eqs. (2.39) and (2.40),

$$S_m = \frac{LFS}{\sqrt{1 + \frac{f_{CF1}^2}{f^2}} \sqrt{1 + \frac{f^2}{f_{CF2}^2}} \sqrt{1 + \frac{f^2}{f_{CF3}^2}}}$$ \hspace{1cm} (2.39)

$$Phase_m = \tan^{-1} \frac{C_1}{f} - \tan^{-1} \frac{f}{C_2} - \tan^{-1} \frac{f}{C_3}$$ \hspace{1cm} (2.40)

where $LFS$ is the sensitivity at 250 Hz, $f_{CFi}$ and $C_i$ are values that correspond to corner frequencies of the element, and $f$ is the frequency of the measured velocity. Figure 2-6 shows sensitivity and phase correction plots for a Microflown element using typical values for $LFS$, $f_{CFi}$, and $C_i$. The actual values of these variables for the sensor used in this research are contained in Section 6.1.3. The initial roll-off in the sensitivity plot is caused by diffusion effects, which can be estimated by a first order low pass frequency response that has a diffusion or thermal lag corner frequency. The second high frequency roll-off is related to the heat capacity of the wires. It shows an exact first order low pass
behavior that has a heat capacity corner frequency. Amplitude and phase corrections will be implemented as part of the data post processing.

Figure 2-6 Microflown Amplitude and Phase Correction Plots: a) Sensitivity and b) Phase
This chapter provides an overview of current NAH measurement techniques along with their limitations. The NAH theories for planar and cylindrical geometries are presented. A number of practical aspects of NAH measurements are also discussed. Topics presented in this chapter are outlined as follows:

- Overview of Current NAH Methods
- Separable Geometries
- Stationary Sources
- Transient Problems
- NAH Measurements

### 3.1 Overview

There are three approaches in current NAH theory to solving the inverse problem. The first is a wave number decomposition method implemented with the discrete Fourier transform\(^ {18} \), which is suitable for separable geometries of the wave equation. The main limitation of this method is that it is only applicable to problems whose geometry conforms closely to a plane, cylinder, or sphere. This method also decreases in its practicality at high frequencies because it relies on a spatial sampling of the pressure field.
The remaining two methods deal with more complex geometries. The singular value decomposition (SVD) in conjunction with inverse boundary elements (IBEM)\textsuperscript{19-20} is commonly used in these cases. The main drawback of IBEM is that it requires the source surface to be discretized with a minimum number of nodes per wavelength in order to achieve a desired resolution in reconstruction. Accordingly, enough measurements of the pressure field must be made to determine the acoustic quantities on these nodes. For a complex structure vibrating at mid to high frequencies, the number of nodes required to describe the surface acoustic field may be large. Hence the number of measurements is also large, which makes the reconstruction process very time consuming.

The Helmholtz equation least squares (HELS)\textsuperscript{21-22} method also applies to complex geometries. HELS is based upon spherical wave expansions which do not use IBEM. Because the expansion uses spherical waves, the reconstruction of an arbitrarily shaped object may be unsatisfactory. The more the surface deviates from a sphere, the worse the reconstruction becomes. Determination of the optimum number of expansions to be used is also not straightforward.

### 3.2 Separable Geometries

Separable solutions of the acoustic wave equation are available for planar, cylindrical, and spherical geometries. The Fourier-based NAH method has been successfully applied to problems where the sources conform closely to one of these separable geometries\textsuperscript{5,23}. This work investigates only planar and cylindrical problems. A description of the Fourier-based NAH implementation for these geometries is presented
in the following subsections. Figure 3-1 shows the required steps for application of the Fourier NAH algorithm. The theory behind each step is discussed below.

![Fourier NAH Algorithm Flowchart](image)

**Figure 3-1   Fourier NAH Algorithm Flowchart**

### 3.2.1 Planar Geometry

The basic assumption behind NAH is that the sound field can be decomposed into two wave-types: plane waves and evanescent waves. Consider a general unknown,
steady state pressure distribution \( p(x,y,z) \) in a source-free half space, \( z > 0 \). This pressure can be expressed as a sum of plane and evanescent waves. In this discussion, the pressure distribution is assumed to be in the frequency domain, and the frequency dependence is suppressed for simplicity.

In planar NAH the sound field is measured over a plane. In essence, the measured pressure is used as a boundary condition to solve the homogeneous wave equation, Eq. (2.4). The employment of the homogeneous wave equation assumes that the region exterior to the source surface is source and reflection free, or that the sources radiate into the free-field, as illustrated by Figure 3-2.

![Figure 3-2: Valid Region of the Homogeneous Wave Equation](image)

If the free-field assumption is valid, the solution to the acoustic wave equation for the pressure at any point exterior to the measurement surface, \( p(r) \), (where \( r = r(x,y,z) \)),
can be expressed in terms of the pressure on the measurement surface $S$, $p(r')$, using the first Rayleigh integral equation.

$$p(r) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} p(r') G_d(r-r')dx\,dy$$  \hfill (3.1)$$

In Eq. (3.1), $G_d$ is known as the Dirichlet Green’s function, and acts as the transfer function of the sound pressure field from one plane to another. Since Rayleigh’s integral equation is in the form of a convolution, the computationally efficient two-dimensional Fourier transform to the wavenumber domain is used for its evaluation. Applying the convolution theorem to Eq. (3.1) yields

$$P(k_x,k_y,z) = P(k_x,k_y,z') g_d(k_x,k_y,z-z')$$  \hfill (3.2)$$

where $P$ represents the two-dimensional spatial transform, $z \geq z'$, and $g_d$ has a known closed form solution for the planar case given by

$$g_d = e^{ik(z-z')}$$  \hfill (3.3)$$

Therefore, for planes exterior to the measurement plane, the pressure can be determined by employing Eq. (3.2) directly. However, for planes between the measurement and source plane, $0 \leq z \leq z'$, a deconvolution of Eq. (3.1) must be performed. In the wavenumber domain, this amounts to inverting Eq. (3.2) yielding

$$P(k_x,k_y,z) = P(k_x,k_y,z') \frac{1}{g_d(k_x,k_y,z'-z)}$$  \hfill (3.4)$$

The space domain form of the pressure distribution $p(x,y,z)$ is recovered by performing an inverse two-dimensional Fourier transform on the left hand sides of Eqs. (3.2) and (3.4).
3.2.2 Cylindrical Geometry

The formulation for cylindrical geometries is quite similar to the planar case discussed above. The main difference is with the Green’s function that defines how the cylindrical waves propagate. The relationship governing the propagation is given by Eq. (3.5)

\[ P_n(r, k_z) = \frac{H_n^{(1)}(k, r)}{H_n^{(1)}(k, r')} P_n(r', k_z) \]

where \( r' \) is the radius of the measured pressure contour, \( r \) is the radius of the desired contour, and \( n \) corresponds to the circumferential spectral line. The only restriction on \( r \) is that it be greater than or equal to the radius of the source.

3.3 Stationary Sources

If the operating conditions of the test object are steady state, the measurement surface can be scanned with a sub-array of microphones. The scans must be phase-locked with reference transducers in order to maintain the phasing of the source to the microphone measurements. These transducers are most often either microphones or accelerometers. This is accomplished by measuring the averaged auto-power and cross-power spectral densities between the references and the field measurements. The averaged auto-power spectrum provides the magnitudes for each frequency. The cross-power spectrum is used to obtain the relative phases between each measurement point.

Once the phase-locked data is acquired, it can be separated into the different mutually uncorrelated partial pressure fields using the singular value decomposition. This means that the partial pressure fields can be analyzed separately and combined using a linear superposition. In order to fully characterize the sound field, there must be at least
as many references as there are incoherent sources contributing to the sound field. Therefore, the reference transducers should be placed so that each source contribution is detected.

3.4 Transient Problems

For non-steady state problems, the entire field must be measured at the same time. Because this requires large arrays and high channel count data acquisition hardware, transient problems are often not practical for solution using NAH.

3.5 NAH Measurements

There are many practical aspects of making NAH measurements that must be taken into account to obtain unique and stable solutions to the inverse problem. The filtering and measurement procedures required to obtain meaningful results from NAH are presented below.

3.5.1 Finite Aperture Effects

The theory presented above requires pressure measurements to be made over an infinite plane or an infinite cylindrical surface. This is, of course, impossible in practice. The area over which the pressure measurement is made is called the measurement aperture. Limiting the size of the aperture causes erroneous results when the field is back propagated because of spectral leakage with the two-dimensional FFT. Some rules have evolved over the years, through experience with the NAH technique, relating to aperture size. It is recommended that the measurement aperture be at least twice as large as the source, such that the field is measured an additional 50% beyond the source in each direction. This generally guarantees that the measured pressure field drops off
significantly towards the edges of the aperture, as long as the measurement standoff
distance is small. The measurement then is effectively infinite, since the difference
between the small measured pressure at the aperture edges and zero is insignificant,
compared to the maximum value of the pressure field in the interior of the aperture.

Even when the aperture is larger than the source, there will likely exist a
discontinuity at the aperture edges. This occurs because the pressure measurements will
generally not be zero at the edges. This discontinuity represents a region in space where
there appears to be a high concentration of large, evanescent wavenumbers. The inverse
propagator amplifies these wavenumbers exponentially back toward the source. A
tapered window is used to reduce the size of this discontinuity and to bring the pressure
to zero at the aperture edges. The taper should be confined to as small a region as
possible, so the measured pressure over the source is not altered. The window chosen for
this research is the Tukey window\textsuperscript{24}. The taper for the right aperture edge is given by

\[
 f(x) = \begin{cases} 
 1/2 - 1/2 \cos\left(\pi \left(\frac{x-L_x/2}{x_w}\right)\right) & \text{if } L_x/2 - x_w < x < L_x/2 \\
 1 & \text{if } x \leq L_x/2 - x_w \\
 0 & \text{if } x > L_x/2
 \end{cases}
\]  

(3.6)

where \(x_w\) is the width of the window taper and \(x = L_x/2\) is the right end of the
measurement aperture. Figure 3-3 shows the Tukey window for \(x_w = 1\) and \(L_x/2 = 2.0\).

Once the taper is applied to the four edges, an additional step to help improve the
reconstruction is to enlarge the aperture by zero padding. Enlarging the aperture
effectually interpolates the spectrum by reducing the spacing between discrete spectral
components. It is typical to double the aperture in both dimensions to \(2L_x\) in the \(x\)-
direction and \(2L_y\) in the \(y\)-direction.
3.5.2 Spatial Sampling

One common aspect of all three NAH implementations is that the accuracy of reconstruction is dependent upon adequate representation of the pressure field on the measurement surface. The theoretical NAH formulation requires a continuous sampling of the pressure field over the measurement aperture. However, any measurement system will sample with a discrete interval. The discrete Fourier transform must therefore be used and this discretization leads to aliasing.

The wavenumber decomposition method and IBEM rely on a spatial sampling for field characterization which can cause mid to high frequency problems to become quite
cumbersome. This is due to the fact that the microphone spacing must be less than or equal to a half wavelength of the highest frequency of interest to avoid spatial aliasing. This is related to the Shannon and Nyquist sampling theorem\textsuperscript{25}.

The overall dimensions of the measurement aperture set the wavenumber resolution. The aperture dimensions should be at least one wavelength of the lowest spatial frequency of interest. All frequencies below this lower frequency limit are leaked into the other spectral lines according to the sampling window shape.

### 3.5.3 Wavenumber Filter

The back-propagating inverse Green’s functions strongly amplify the high spatial frequency evanescent waves. To correct this problem NAH imposes a filter in wavenumber space which limits the inclusion of high wavenumber components. The chosen window is the two-dimensional Harris cosine window.

\[
K_{\text{window}}(k_x, k_y) = \begin{cases} 
1 - \frac{1}{2} e^{-\left(\frac{1}{\sqrt{k_x^2 + k_y^2}}\right)^\alpha} & \text{for } \sqrt{k_x^2 + k_y^2} < k_c \\
\frac{1}{2} e^{-\left(\frac{1}{\sqrt{k_x^2 + k_y^2}}\right)^\alpha} & \text{for } \sqrt{k_x^2 + k_y^2} > k_c
\end{cases} \tag{3.7}
\]

In Eq. (3.7), \( k_c \) sets the cut-off wavenumber and \( \alpha \) controls the rate of decay of the window. The filter cut-off wavenumber must be chosen so that the noise effects are reduced, but the data of interest is not removed. Therefore, the cut-off may be determined based on the signal-to-noise ratio of the measurement in conjunction with the standoff distance of the measurement contour from the source. Figure 3-4 shows the window shape for three typical values of \( \alpha \) with \( k_y = 0 \) and \( k_c = 20 \).
3.5.4 Measurement Noise and the Standoff Distance

Spatial noise not correlated with the spatial data results in a uniform noise floor in the wavenumber domain. Noise that falls into this category includes microphone positioning errors, microphone calibration errors, phase mismatch, random noise, and random deviations from steady state for scanned data. In the section above regarding the wavenumber filter, there was no restriction placed on the maximum value chosen for $k_c$. In practice, $k_c$ is limited by the standoff distance and the signal-to-noise ratio of the measured pressure. The filter cut-off must be chosen so that amplified noise effects are minimized, but the data of interest is not removed.
4 ENERGY-BASED RECONSTRUCTION PROCEDURES

This chapter presents in detail the developed procedures for the proposed energy-based reconstruction method. This method can be used in conjunction with current NAH algorithms to obtain accurate characterization of the pressure field with significantly fewer measurements. In the process of explaining each step, the underlying fundamental principles are also presented for the reader unfamiliar with these concepts.

4.1 OVERVIEW

Current NAH reconstruction methods are based solely on measurement of the pressure field. Since pressure is a scalar quantity, it does not provide directional information for the field. Particle velocity measurements, on the other hand, supply first derivative information for the pressure field via Euler’s equation, Eq. (2.9). The in-plane velocities make derivative information available that is used to interpolate between measurement locations. This effectually simulates a finer mesh of pressure measurements. Figure 4-1 shows the correlation between conventional NAH and the proposed energy-based approach.
4.2 HERMITE INTERPOLATION

The chosen interpolation method is taken from the area of geometric modeling. For ease of programming and computability along with other reasons specific to geometric modeling, the preferred way to perform interpolation is with parametric equations. For example, a three-dimensional curve is defined by $x = x(r, s), y = y(r, s)$, and $z = z(r, s)$. It is generally convenient to normalize the
domain of the parametric variables, $r$ and $s$, by restricting their value to the closed interval between 0 and 1, inclusive. We express this condition symbolically as $r, s \in [0,1]$, establishing the bounding curves and creating a surface patch between them. These curves have a natural vector representation given by Eq. (4.1).

$$f(r,s) = [x(r,s) \ y(r,s) \ z(r,s)]$$

Farin\textsuperscript{26} points out that a piecewise lower order polynomial interpolation approach is superior in speed and accuracy to their higher order counterparts. Therefore, bicubic polynomial interpolation is selected. Hermite surface patches are chosen for interpolation between measurement locations because they match both function values and slopes at the specified corner points.

### 4.2.1 Curves

Bicubic Hermite surfaces are made up of an orthogonal net of cubic Hermite curves. Therefore, a preliminary discussion of these curves is necessary to provide the foundation upon which the surface interpolation is built. The algebraic form of a parametric cubic curve is given by the polynomials in Eq. (4.2).

$$\begin{align*}
x(r) &= a_r r^3 + b_r r^2 + c_r r + d_r \\
y(r) &= a_y r^3 + b_y r^2 + c_y r + d_y \\
z(r) &= a_z r^3 + b_z r^2 + c_z r + d_z
\end{align*}$$

The 12 scalar coefficients, called algebraic coefficients, determine a unique curve. Using vector notation to obtain a more compact form, Eq. (4.2) becomes

$$\mathbf{f}(r) = \mathbf{a} r^3 + \mathbf{b} r^2 + \mathbf{c} r + \mathbf{d}$$
where $\mathbf{f}(r)$ is the position vector of any point on the curve and $\mathbf{a}$, $\mathbf{b}$, $\mathbf{c}$, and $\mathbf{d}$ are the vector equivalents of the scalar algebraic coefficients. The algebraic coefficients are not the most convenient way of controlling the shape of a curve, nor do they provide an intuitive sense of the curve shape. The Hermite form allows for the definition of conditions at its endpoints, or boundaries. Using the endpoints $\mathbf{f}(0)$ and $\mathbf{f}(1)$, the corresponding tangent vectors $\mathbf{f}^\prime(0)$ and $\mathbf{f}^\prime(1)$ (where the $r$ superscript denotes the derivative with respect to $r$), and Eq. (4.3) yields the following four equations

\[
\begin{align*}
\mathbf{f}(0) &= \mathbf{d} \\
\mathbf{f}(1) &= \mathbf{a} + \mathbf{b} + \mathbf{c} + \mathbf{d} \\
\mathbf{f}^\prime(0) &= \mathbf{c} \\
\mathbf{f}^\prime(1) &= 3\mathbf{a} + 2\mathbf{b} + \mathbf{c}
\end{align*}
\]

(4.4)

where substituting $r = 0$ into Eq. (4.3) yields $\mathbf{f}(0)$, and substituting $r = 1$ into the equation yields $\mathbf{f}(1)$. Differentiating $\mathbf{f}(r)$ with respect to $r$ obtains

$\mathbf{f}^\prime(r) = 3\mathbf{a}r^2 + 2\mathbf{b}r + \mathbf{c}$. Substituting $r = 0$ and $r = 1$ into this yields $\mathbf{f}^\prime(0)$ and $\mathbf{f}^\prime(1)$ respectively. Solving this set of four simultaneous vector equations in four unknown vectors yields the algebraic coefficients in terms of the boundary conditions.

\[
\begin{align*}
\mathbf{a} &= 2\mathbf{f}(0) - 2\mathbf{f}(1) + \mathbf{f}^\prime(0) + \mathbf{f}^\prime(1) \\
\mathbf{b} &= -3\mathbf{f}(0) + 3\mathbf{f}(1) - 2\mathbf{f}^\prime(0) - \mathbf{f}^\prime(1) \\
\mathbf{c} &= \mathbf{f}^\prime(0) \\
\mathbf{d} &= \mathbf{f}(0)
\end{align*}
\]

(4.5)

Substituting these equations for the algebraic coefficient vectors into Eq. (4.3) and rearranging terms produces Eq. (4.6).
\[ f(r) = (2r^3 - 3r^2 + 1)f(0) + (-2r^3 + 3r^2)f(1) + \left(r^3 - 2r^2 + r\right)f'(0) + \left(r^3 - r^2\right)f'(1) \quad (4.6) \]

This equation is simplified by making the following substitutions.

\[
\begin{align*}
B_1(r) &= 2r^3 - 3r^2 + 1 \\
B_2(r) &= -2r^3 + 3r^2 \\
B_3(r) &= r^3 - 2r^2 + r \\
B_4(r) &= r^3 - r^2
\end{align*} \quad (4.7)
\]

Using these simplifications and subscripts to represent the endpoint \( r \) values, Eq. (4.6) becomes

\[ f(r) = B_1(r)f_0 + B_2(r)f_1 + B_3(r)f'_0 + B_4(r)f'_1 \quad (4.8) \]

Equation (4.8) is called the geometric form, and the vectors \( f_0, f_1, f'_0, \) and \( f'_1 \) are the geometric coefficients. The \( B_i(r) \) terms are the Hermite basis functions. Figure 4-2 shows each basis function as a curve over the domain of the parameter \( r \).
These basis functions have three important characteristics. First, they are universal for all cubic Hermite curves. Second, they are only dependent on the parameter, making them identical for each of the three real space coordinates. Finally, they allow the constituent boundary condition coefficients to be decoupled from each other. These functions blend the effects of the endpoints and tangent vectors to produce the intermediate point coordinate values over the parameter domain. Letting
the geometric form given in Eq. (4.8) can be transformed into the more computationally efficient matrix form, where $M_H$ is the Hermite basis transformation matrix and $G_H$ is the geometry matrix containing the curve boundary conditions.

$$f(r) = R M_H G_H$$

The geometry matrix in Eq. (4.10) is altered for each segment to obtain a series of cubic Hermite curves which are combined to form a composite curve with slope continuity at the endpoints.

### 4.2.2 Surfaces

A large complex surface can be defined by a composite collection of simpler patches. The algebraic form of a bicubic Hermite patch is given by the tensor product shown in Eq. (4.11).

$$f(r, s) = \sum_{i=0}^{3} \sum_{j=0}^{3} a_{ij} r^i s^j$$

The $a_{ij}$ are the three component algebraic coefficient vectors of the patch, where each component represents one of the three dimensions in real space. The subscripting corresponds to the order of the parameter variables that the coefficient is attributed to.

Expanding Eq. (4.11) and arranging the $a_{ij}$ terms in descending order produces Eq. (4.12), a 16 term polynomial in $r$ and $s$. 

$$R = \begin{bmatrix} r^3 & r^2 & r & 1 \\ 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

$$M_H = \begin{bmatrix} f_0 & f_1 & f'_0 & f'_1 \end{bmatrix}^T$$

$$G_H = \begin{bmatrix} f_0 & f_1 & f'_0 & f'_1 \end{bmatrix}^T$$
Because each of the 16 vector coefficients \( a_{ij} \) has three independent components, there are a total of 48 algebraic coefficients, or 48 degrees of freedom. In matrix notation, the algebraic form is

\[
f(r,s) = R A S^T
\]

(4.13)

where

\[
R = \begin{bmatrix} r^3 & r^2 & r & 1 \\ s^3 & s^2 & s & 1 \end{bmatrix}
\]

\[
S = \begin{bmatrix} a_{33} & a_{32} & a_{31} & a_{30} \\ a_{23} & a_{22} & a_{21} & a_{20} \\ a_{13} & a_{12} & a_{11} & a_{10} \\ a_{03} & a_{02} & a_{01} & a_{00} \end{bmatrix}
\]

(4.14)

Since the \( a \) elements are three-component vectors, the \( A \) matrix is actually a 4 x 4 x 3 array. As was found with Hermite curves, the algebraic coefficients of a Hermite patch determine its shape and position in space. Although the \( r, s \) parameter domain values are restricted between 0 and 1, the range of the variables in \( x, y, \) and \( z \) is not restricted, because the range of the algebraic coefficients is not limited. A unique point on the surface patch is generated each time a specific pair of \( r, s \) values are input into Eq. (4.13). These pairs of \( r, s \) values are then mapped back into real space.

Each patch is bounded by four curves, and each boundary curve is a cubic Hermite curve. Applying the same subscripting notation as implemented in Eq. (4.8),
these curves are denoted as: $f_{rs}$, $f_{s}^r$, $f_{r}^s$, $f_{r,s}$, because they arise at the limit values of the parametric variables. There are also four unique corner points $f_{00}$, $f_{01}$, $f_{10}$, and $f_{11}$. As was seen for curves, the geometric form is a more convenient and intuitive way to define a patch. The geometric form is derived in the same way as for curves. The boundary conditions of the patch are used to solve for the algebraic coefficients. These conditions include the four patch corner points $f_{00}$, $f_{01}$, $f_{10}$, $f_{11}$ and the eight tangent vectors $f_{00}^r$, $f_{00}^s$, $f_{10}^r$, $f_{10}^s$, $f_{01}^r$, $f_{01}^s$, $f_{11}^r$, $f_{11}^s$ which define the boundary curves. $B$ once again represents the Hermite basis functions, as in Eq. (4.8).

$$
\begin{align*}
\mathbf{f}(r,0) &= B(r) \begin{bmatrix} f_{00} & f_{10} & f_{00}^r & f_{10}^r \end{bmatrix}^T \\
\mathbf{f}(r,1) &= B(r) \begin{bmatrix} f_{01} & f_{11} & f_{01}^r & f_{11}^r \end{bmatrix}^T \\
\mathbf{f}(0,s) &= B(s) \begin{bmatrix} f_{00} & f_{01} & f_{00}^s & f_{01}^s \end{bmatrix}^T \\
\mathbf{f}(1,s) &= B(s) \begin{bmatrix} f_{10} & f_{11} & f_{10}^s & f_{11}^s \end{bmatrix}^T
\end{align*}
$$

(4.15)

These four curves provide 12 of the 16 vectors needed to specify the 48 degrees of freedom. Four additional vectors at the corner points, called twist vectors, are used to fully specify the patch. Mathematically, these vectors are defined as follows:

$$
\begin{align*}
\mathbf{f}_{00}^{rs} &= \frac{\partial^2 \mathbf{f}(r,s)}{\partial r \partial s} \quad \text{at} \quad r = 0, s = 0 \\
\mathbf{f}_{10}^{rs} &= \frac{\partial^2 \mathbf{f}(r,s)}{\partial r \partial s} \quad \text{at} \quad r = 1, s = 0 \\
\mathbf{f}_{01}^{rs} &= \frac{\partial^2 \mathbf{f}(r,s)}{\partial r \partial s} \quad \text{at} \quad r = 0, s = 1 \\
\mathbf{f}_{11}^{rs} &= \frac{\partial^2 \mathbf{f}(r,s)}{\partial r \partial s} \quad \text{at} \quad r = 1, s = 1
\end{align*}
$$

(4.16)

Calculating the mixed partial derivative of Eq. (4.12) yields
\[
\frac{\partial^2 f(r,s)}{\partial r \partial s} = 9a_{31}r^2s^2 + 6a_{32}r^2s + 3a_{33}r^2 + 6a_{23}rs + 2a_{21}r + 3a_{13}s^2 + 2a_{12}s + a_{11}
\] (4.17)

Evaluating Eq. (4.17) at the corner points obtains

\[
\begin{align*}
f_{00}^c &= a_{11} \\
f_{10}^c &= 3a_{31} + 2a_{21} + a_{11} \\
f_{01}^c &= 3a_{13} + 2a_{12} + a_{11} \\
f_{11}^c &= 9a_{33} + 6a_{32} + 3a_{31} + 6a_{23} + 4a_{22} + 2a_{21} + 3a_{13} + 2a_{12} + a_{11}
\end{align*}
\] (4.18)

Doing the same for the remaining 12 vectors provides the remaining 12 equations required to solve for the algebraic coefficients.

\[
\begin{align*}
f_{00} &= a_{00} \\
f_{10} &= a_{30} + a_{20} + a_{10} + a_{00} \\
f_{01} &= a_{03} + a_{02} + a_{01} + a_{00} \\
f_{11} &= a_{33} + a_{32} + a_{31} + a_{30} + a_{23} + a_{22} + a_{21} + a_{20} + a_{13} + a_{12} + a_{11} + a_{03} + a_{02} + a_{01} + a_{00} \\
f_{00}^c &= a_{10} \\
f_{01}^c &= a_{01} \\
f_{10}^c &= 3a_{30} + 2a_{20} + a_{10} \\
f_{10}^c &= a_{31} + a_{21} + a_{11} + a_{01} \\
f_{01}^c &= a_{13} + a_{12} + a_{11} + a_{10} \\
f_{01}^c &= 3a_{03} + 2a_{02} + a_{01} \\
f_{11}^c &= 3a_{33} + 3a_{32} + 3a_{31} + a_{30} + 2a_{23} + 2a_{22} + 2a_{21} + 2a_{20} + a_{13} + a_{12} + a_{11} + a_{10} \\
f_{11}^c &= 3a_{33} + 2a_{32} + a_{31} + 3a_{23} + 2a_{22} + a_{21} + 3a_{13} + 2a_{12} + a_{11} + 3a_{03} + 2a_{02} + a_{01}
\end{align*}
\] (4.19)

Solving this set of 16 simultaneous equations from Eqs. (4.18) and (4.19) for the algebraic coefficients in terms of the geometric inputs and rearranging terms yields
Recalling from Eq. (4.10) that $B(r)$ may be expressed as $\mathbf{RM}_H$, Eq. (4.20) may be further simplified to obtain the conventional geometric form given by

$$ f(r, s) = \mathbf{RM}_H \mathbf{G}_H \mathbf{M}_H S^T $$

(4.21)

where $\mathbf{G}_H$ is the Hermite geometry matrix shown in Eq. (4.22).

$$ \mathbf{G}_H = \begin{bmatrix} f_{00} & f_{01} & f_{00}^x & f_{01}^x \\ f_{10} & f_{11} & f_{10}^x & f_{11}^x \\ f_{00}^r & f_{01}^r & f_{00}^{rs} & f_{01}^{rs} \\ f_{10}^r & f_{11}^r & f_{10}^{rs} & f_{11}^{rs} \end{bmatrix} $$

(4.22)

The remaining intricacy of the interpolation relates to converting between real and parameter space. A simple method for mapping between the two domains is presented below. Figure 4-3 provides an example of a set of four corner points in $x$ and $y$ that could be used to define a patch.
In this case, the $r$ parameter corresponds to the $x$-direction and the $s$ parameter to the $y$-direction. Each $(x, y)$ coordinate pair inside the patch corresponds to an $(r, s)$ parameter pair. This parameter pair is obtained using Eq. (4.23).

$$
\begin{align*}
    r &= \frac{x - x_0}{\Delta x} \\
    s &= \frac{y - y_0}{\Delta y}
\end{align*}
$$

(4.23)

where $\Delta x$ and $\Delta y$ correspond to the spacing between corner points in $x$ and $y$ respectively.

The slopes at the endpoints must also be transformed to the parameter domain. This is accomplished by scaling the $r$ derivatives by $\Delta x$ and the $s$ derivatives by $\Delta y$ as shown in Eq. (4.24) for the corner point corresponding to $r = s = 0$.
\[
\begin{align*}
f_{00}^r &= \frac{\partial f_{00}}{\partial x} \cdot \frac{\Delta x}{\Delta r} \\
f_{00}^s &= \frac{\partial f_{00}}{\partial y} \cdot \frac{\Delta y}{\Delta s}
\end{align*}
\]  
(4.24)

where \(\Delta r\) and \(\Delta s\) equal one because they are restricted to vary from zero to one. The form is the same for the remaining three corner points of the patch.

Because measurements with energy-based sensors in general do not provide enough information to calculate twist vectors, they have been set to zero for this investigation. The Hermite geometry matrix from Eq. (4.22) then becomes

\[
G_H = \begin{bmatrix}
f_{00} & f_{01} & f_{00}^x & f_{01}^x \\
f_{10} & f_{11} & f_{10}^x & f_{11}^x \\
f_{00}^r & f_{01}^r & 0 & 0 \\
f_{10}^r & f_{11}^r & 0 & 0
\end{bmatrix}
\]  
(4.25)

This limits the patches to having only first derivative continuity at their edges. The results presented below indicate that adequate reconstructions are still obtained with this simplification. Figure 4-4 shows a sample bicubic Hermite patch and the required inputs at each corner point \(f_n\). Each patch represents the rectangular area between four corner point locations. The above interpolation is repeated for each segment of the surface and all the patches combined. Figure 4-5 shows an example of a 3 x 3 grid of patches combined to form a more complex composite surface.
Figure 4-4  Sample Bicubic Hermite Surface Patch with Required Inputs Shown

Figure 4-5  Complex Surface Composed of Nine Bicubic Hermite Patches
5 ANALYTICAL IMPLEMENTATION

In this chapter, an analytical model is developed to investigate the theoretical benefits of the energy-based reconstruction method. This model requires first that a synthetic acoustic field be created from a hypothetical source. The field is then sampled and the chosen NAH algorithm implemented. The error evaluation method is introduced. The ENAH reconstructions are then compared to results obtained through conventional NAH. A flowchart illustrating the process followed for the analytical implementation is shown in Figure 5-1.
Compute Standard Deviation of the Residuals and Normalize by the Maximum Pressure Value

Whole Field Estimation

Figure 5-1 Flowchart Illustrating Analytical Implementation
5.1 Synthetic Field Creation

A rectangular, simply supported plate is chosen as the hypothetical source because it has a simple closed-form radiation equation. The plate shown in Figure 5-2 is driven by a harmonic point source acting normal to the plate at its center. The surface displacement, \( w \), for the plate as a function of angular frequency, \( \omega \), is given by

\[
\begin{align*}
    w(x, y, \omega) &= -\frac{F}{\rho h} \sum_{m=1}^{\infty} \sum_{n=1}^{\infty} \frac{\Phi_{mn}(x_0, y_0) \Phi_{mn}(x, y)}{\omega^2 - \omega_{mn}^2} \\
    \Phi_{mn}(x, y) &= \frac{2}{\sqrt{L_x L_y}} \sin \left( \frac{m\pi x}{L_x} \right) \sin \left( \frac{n\pi y}{L_y} \right)
\end{align*}
\]

where \( F \) is the excitation force amplitude, \( \rho \) the plate material density per unit area, \( h \) the plate thickness, \( L_x \) the plate width, and \( L_y \) the plate length.

![Figure 5-2 Simply Supported Plate Chosen as the Source for the Analytical Modeling](image-url)
The subscripts \( m \) and \( n \) denote the plate mode numbers in \( x \) and \( y \) respectively. The natural frequencies \( \omega_{mn} \) of the plate are calculated using Eq. (5.3)

\[
\omega_{mn} = \sqrt{\frac{D}{\rho h}} \left[ \left( \frac{m\pi}{L_x} \right)^2 + \left( \frac{n\pi}{L_y} \right)^2 \right]
\]  

(5.3)

where \( D \) is the bending stiffness of the plate and is given by

\[
D = \frac{Eh^3}{12(1-\nu^2)}
\]  

(5.4)

In Eq. (5.4), \( E \) is the plate material modulus and \( \nu \) is Poisson’s ratio for the material.

Assuming that the plate is in an infinite rigid baffle, the radiated pressure can be expressed in terms of the plate surface displacement using Rayleigh’s integral. Figure 5-3 provides a clear description of the geometric quantities to be used in Eq. (5.5), where \( e^{-i\omega t} \) time dependence has been assumed and \( \rho_0 \) is the density of the fluid.

\[
p(x,y,\omega) = -\frac{\omega^2 \rho_0}{2\pi} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} w(x',y',\omega) \frac{e^{i|\mathbf{r'-r}|}}{|\mathbf{r'-r}|} dx'dy'
\]  

(5.5)

Figure 5-3  Description of the Geometric Quantities in Rayleigh's Integral
The pressure at a point in space, \( p(x, y, z) \), is computed by summing the contribution from each \( dx'dy' \) area element. Twenty modes in \( x \) and \( y \) are used in Eq. (5.1) to compute the plate surface displacement. Radiation from the plate is simulated using a discrete summation of Eq. (5.5) for a 32 x 32 grid of point sources on the plate. The field is then sampled at chosen measurement locations to obtain the pressure and gradient information to be used for interpolation. The selected NAH algorithm is then applied to reconstruct the field.

5.2 ERROR EVALUATION

The reconstruction error is evaluated by first calculating the pressure field at the measurement and estimation planes directly using Eq. (5.5) as illustrated in Figure 5-4. The direct calculation of the pressure field at the estimation plane serves as a reference against which the NAH reconstruction is compared.

![Figure 5-4 Illustration of Error Evaluation Method](image)
The reconstruction error is quantified by differencing the NAH estimation and the direct calculation at the estimation plane. The standard deviation of these residuals is then computed and normalized by the maximum pressure field value to obtain a single value representing the whole field error. This error is compared for the energy-based and pressure only NAH reconstructions. An example of this procedure is presented below. Figure 5-5 shows the reference pressure at 2 cm for an aluminum, simply supported plate with dimensions 30.5 cm x 45.7 cm x 0.3175 cm forced at 1094 Hz corresponding to the 3, 3 mode as shown in Figure 5-2 above. In practice the pressure field would be propagated all the way back to the source plane. However, in order to be able to verify these results experimentally, the reconstruction plane is set to 2 cm.

![Reference Pressure at Estimation Plane](image)

**Figure 5-5** Reference Pressure at the 2 cm Estimation Plane
The synthetic pressure field is sampled over a 20 x 20 grid at 5 cm and then back propagated to the estimation plane. The resulting reconstruction using conventional NAH is shown in Figure 5-6. The estimation in Figure 5-6 maintains the general modal shape of the field with some errors in magnitude. The result from differencing the estimation (Figure 5-6) and the direct calculation (Figure 5-5) to obtain the residuals is shown in Figure 5-7. The standard deviation of the residuals shown in Figure 5-7 normalized by the maximum pressure field value is 0.043.

![Image: Estimated Pressure from Conventional NAH with a 20 x 20 Grid of Pressure Measurements at 5 cm](image_url)

**Figure 5-6** Estimated Pressure from Conventional NAH with a 20 x 20 Grid of Pressure Measurements at 5 cm
5.3 **ANALYTICAL RESULTS**

For the analytical investigation, the number of sensors used to populate the measurement array is varied in both the $x$ and the $y$-directions in order to determine the possible reduction in sensor locations using energy-based measurements. The whole field error is then calculated for each sensor configuration. The results below correspond to the synthetic field generated by the $30.5 \times 45.7 \times 0.3175$ cm plate vibrating in the 3, 3 mode, which is used above to illustrate the error evaluation method. These dimensions are chosen to match the dimensions used for the experimental validation presented in Chapter 6. The measurement plane is set to 5 cm and the estimation plane to 2 cm above the plate. Figure 5-8 shows the resulting normalized whole field estimation error plots for measurement array sizes ranging from 10 x 10 to 20 x 20 for conventional and energy-based NAH reconstruction.
These plots indicate that the inclusion of field directional information in the reconstruction significantly improves the ability to reconstruct the field accurately. In fact, a 10 x 10 array of energy-based measurements has a slightly lower whole field error than a 20 x 20 array of pressure measurements. These results show that the number of measurement locations may be reduced by about 75% when energy-based sensing equipment is used. This reduction seems reasonable since twice the information is being used in each direction. If a three-channel probe is utilized to measure the field, a channel count reduction of 25% would also be realized for non-scanning systems. These results represent the theoretical optimal performance of the conventional and energy-based reconstructions because the measurements have zero positioning, amplitude, and phase error.
6 EXPERIMENTAL VALIDATION

Traditional Fourier-based NAH has had great success analyzing sources which conform closely to one of the separable geometries of the wave equation. Therefore, a planar and a cylindrical test case were investigated in order to validate the analytical results presented above. This chapter is intended to provide the reader with the information required to recreate the experiments and results obtained in this research. The procedures followed for each test case are presented below. The reconstruction results using conventional NAH and ENAH are presented along with a brief discussion about their correlation to the analytical implementation.

6.1 PLANAR TEST CASE

The first test case investigated was a rectangular plate forced at its center. The experimental setup and data acquisition procedures in validating the analytical results presented in Chapter 5 are discussed. The data post processing necessary to overcome the ill-posedness of the inverse problem is also introduced in detail.

6.1.1 Setup

An experimental setup was designed to approximate the simply supported plate used in the analytical investigation. Figure 6-1a shows the 30.5 cm x 45.7 cm x 0.3175 cm aluminum plate. It is attached along its edges to a heavy steel frame using cone point set screws located at 1.2 cm intervals. Using the cone point set screws provides the zero
displacement at the edges yet still allows the rotation about the points required to approximate the simply supported boundary condition. A 20 mm diameter piezoelectric patch, shown in Figure 6-1b, was used to excite the plate at its center at various frequencies to determine how well the simply supported boundary conditions were approximated. Table ?? shows the correlation between the analytical and actual frequencies for three modes investigated. For this test case, the plate is excited at 1090 Hz corresponding to the 3, 3 mode of the plate.

Table 6-1   Comparison Between Analytical and Actual Modal Frequencies

<table>
<thead>
<tr>
<th>Mode (x, y)</th>
<th>Analytical Frequency (Hz)</th>
<th>Actual Frequency (Hz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,5</td>
<td>1019.6</td>
<td>1009</td>
</tr>
<tr>
<td>3,3</td>
<td>1094.4</td>
<td>1090</td>
</tr>
<tr>
<td>3,6</td>
<td>2104.6</td>
<td>2038</td>
</tr>
</tbody>
</table>

Figure 6-1   Planar Experimental Setup:  a) Aluminum Plate in Steel Frame with Wood Frame and Sensor Shown and b) 20 mm Diameter Piezoelectric Patch Used to Excite Plate

The plate was hung in a fully anechoic chamber parallel to a wood frame that held the Microflown Ultimate Sound Probe (USP) sensor shown in Figure 6-2a. The USP
sensor consists of a single electret condenser microphone and three Microflown elements oriented orthogonally to each other, as shown in Figure 6-2b, to measure the three components of the particle velocity. Each Microflown element requires a single channel for acquisition. Therefore, the USP sensor requires three channels to measure the pressure and two in-plane particle velocities. The pressure sensor is a Knowles FG series microphone that measures 2.59 mm in diameter. This microphone is shown in Figure 6-3.

Figure 6-2  USP Sensor Images:  a) USP Sensor and b) Close Up of the Three Orthogonal Microflown Elements

Figure 6-3  Knowles FG Series Electret Condenser Microphone
The wood frame contained locating holes at 5 cm intervals on the upper and lower horizontal members through which a 6 mm metal rod passed. The USP sensor was attached to rod such that it was perpendicular to the frame and plate as shown in Figure 6-1a.

6.1.2 Data Acquisition

A single USP was used to scan the field to obtain the pressure and in-plane velocities required for reconstruction. The field was sampled at 2 cm and 5 cm from the plate as in the analytical case. The 2 cm measurement again serves as the reference against which the NAH reconstructions from the 5 cm measurement plane are compared. The vertical and horizontal step distance was set to 5 cm and the plate was overscanned in both directions yielding a 50 cm x 80 cm overall measurement array size. This array size resulted in a total of 187 measurements. The measurement grid in relation to the plate is shown in Figure 6-4. The location of the plate is shaded in gray while the measurement locations are illustrated by the black dots.
At each measurement location, averaged autopowers of the USP measurements and crosspowers between the reference and USP sensor were captured using a Data Physics SignalCalc 620 data acquisition system, where the input signal to the piezoelectric patch was used as the reference. Ten stable, one-second averages with a sample rate of 8124 Hz were taken at each measurement location resulting in a one Hz temporal frequency resolution.

6.1.3 Data Post Processing

The first task to complete once the data have been acquired is to apply the amplitude and phase corrections for the microphone and each Microflown element.
These corrections for the Knowles microphone are accomplished using Eqs. (6.1) and (6.2),

\[ S_k = \frac{\text{Sensitivity}}{\sqrt{1 + \left(\frac{f_{KCF1}}{f}\right)^2}} \]  
\[ \text{Phase}_k = \tan^{-1} \left( \frac{f_{KCF1}}{f} \right) \]  

where the values for Sensitivity and \( f_{KCF1} \) are specific to each microphone. For the sensor used in this study, these values are given in Table 6-2.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensitivity</td>
<td>15.3 mV/Pa</td>
</tr>
<tr>
<td>( f_{KCF1} )</td>
<td>56 Hz</td>
</tr>
</tbody>
</table>

The equations governing the amplitude and phase correction for the Microflown elements are given by

\[ S_m = \frac{LFS}{\sqrt{1 + \frac{f_{CF1}^2}{f^2}} \sqrt{1 + \frac{f_{CF2}^2}{f^2}} \sqrt{1 + \frac{f_{CF3}^2}{f^2}}} \]  
\[ \text{Phase}_m = \tan^{-1} \frac{C_1}{f} - \tan^{-1} \frac{f}{C_2} - \tan^{-1} \frac{f}{C_3} \]

where the necessary parameter values for the sensor used in this research are contained in Table 6-3.
The next step is to use the phase between the reference and sensor measurement obtained from the crosspower spectrum to correlate the measurements. This is accomplished by first choosing an arbitrary single measurement location to be the “zero phase” point. The phase at the “zero phase” point is then subtracted from the crosspower phase value at each measurement location. The term “zero phase” point is used here because when its phase is subtracted from itself the phase will then be zero. This process correlates the measurements so they appear to have been all measured simultaneously. It should be noted that this is only applicable to steady state cases.

Because the Hermite interpolation used in ENAH requires pressure gradient inputs, the in-plane velocities must be converted to pressure gradients before the algorithm is employed. Since spectral data were acquired, Euler’s equation (Eq. (2.9))

**Table 6-3  Microflown Element Calibration Parameter Values**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>LFS</td>
<td>27.5 V/(m/s)</td>
</tr>
<tr>
<td>$f_{CF1}$</td>
<td>102 Hz</td>
</tr>
<tr>
<td>$f_{CF2}$</td>
<td>1200 Hz</td>
</tr>
<tr>
<td>$f_{CF3}$</td>
<td>13000 Hz</td>
</tr>
<tr>
<td>$C_1$</td>
<td>102 Hz</td>
</tr>
<tr>
<td>$C_2$</td>
<td>1300 Hz</td>
</tr>
<tr>
<td>$C_3$</td>
<td>12000 Hz</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>LFS</td>
<td>35.8 V/(m/s)</td>
</tr>
<tr>
<td>$f_{CF1}$</td>
<td>103 Hz</td>
</tr>
<tr>
<td>$f_{CF2}$</td>
<td>1200 Hz</td>
</tr>
<tr>
<td>$f_{CF3}$</td>
<td>13000 Hz</td>
</tr>
<tr>
<td>$C_1$</td>
<td>103 Hz</td>
</tr>
<tr>
<td>$C_2$</td>
<td>1200 Hz</td>
</tr>
<tr>
<td>$C_3$</td>
<td>10000 Hz</td>
</tr>
</tbody>
</table>
transformed to the frequency domain is applied. The pressure and in-plane pressure gradients on the 5 cm measurement plane are shown in Figures 6.5 through 6.7 below for the 11 x 17 measurement grid.

**Figure 6-5** Measured Pressure Field at 5 cm

**Figure 6-6** Measured Pressure Gradient Field in \( x \)-direction at 5 cm
Once the above corrections have been made, the energy-based reconstruction method presented in Chapter 4 is applied. Figure 6-8 shows the pressure field at 5 cm, after Hermite interpolation using the 11 x 17 grid of pressure and gradient information is applied.
The interpolated field in Figure 6-8 is then windowed on all four edges using the Tukey window described in Section 3.5.1. The resulting pressure field with a window width $w_x$ of 0.15 m is shown in Figure 6-9. Comparing Figure 6-9 and Figure 6-8, it may be seen that the pressure is taken to zero gracefully at the edges without significantly affecting the higher amplitude portions of the field. The windowed field is then zero-padded to double its size in $x$ and $y$, to reduce the effect of the finite measurement aperture. The windowed and zero-padded field is shown in Figure 6-10.
Figure 6-9  Windowed Pressure Field at 5 cm with $x_w$ at 0.15 m

Figure 6-10  Windowed and Zero-padded Pressure Field at the Measurement Plane
A two-dimensional spatial Fourier transform is then applied to the windowed and zero-padded pressure field to transfer to the wavenumber domain, where the propagation becomes simple multiplication rather than convolution. The magnitude of the wavenumber domain components plotted versus $k_x$ and $k_y$ is presented in Figure 6-11. The larger magnitudes occur at relatively low wavenumbers because the measured field does not have a high degree of spatial variation over short distances. However, there are lower amplitude peaks around $k_y$ and $k_x = \pm 100$.

![Figure 6-11 Wavenumber Domain Magnitudes](image)

The next step is to multiply the wavenumber domain by the propagation function corresponding to each spectral component. The propagation function is the Dirichlet Green’s function given in Eq. (3.3), where the estimation plane $z'$ is 2 cm for this case. The resulting propagated wavenumber components are shown in Figure 6-12. The new
wavenumber domain at the estimation plane does not appear at all similar to Figure 6-11. Comparing the color bar scales of Figure 6-11 and Figure 6-12 reveals the problem of back propagation towards the source surface. Because the evanescent propagation functions grow exponentially as the wavenumbers increase, high frequency noise dominates the actual spectral components of the field. For this reason, a wavenumber filter is applied to control the amplification of high frequency noise. This filter was discussed previously in Section 3.5.3. Applying the filter to the propagated wavenumber components in Figure 6-12, with cutoff $k_c = 60$ and decay rate parameter $\alpha = 0.2$, results in the filtered wavenumber domain shown in Figure 6-13.

Figure 6-12   Magnitudes of Wavenumber Domain Components after Propagation to the 2 cm Estimation Plane
The final step is to perform an inverse two-dimensional Fourier transform on the propagated and filtered wavenumber components in Figure 6-13 to return to real space. This yields the propagated pressure field at the 2 cm estimation plane. The reconstructed field is shown in Figure 6-14.
6.1.4 Results

Figure 6-16 shows the reference pressure as measured on the 2 cm estimation plane. ENAH and conventional NAH were performed from the measurement at 5 cm. All reconstruction variables, such as window and filter parameters, were held constant to prevent biasing the results towards either reconstruction technique. An 11 x 17 array (187 measurement locations) of pressure measurements at 5 cm was used to reconstruct the pressure at the estimation plane using the traditional Fourier NAH method. This pressure only reconstruction is compared to the ENAH reconstruction using a 6 x 9 array (54 measurement locations with a grid spacing of 10 cm) of energy-based measurements spanning the same area. Figure 6-15 shows the measurement array locations for each
reconstruction technique. The resulting reconstructions are shown in Figure 6-17. Plots of the pressure error for the reconstructions are also presented below in Figure 6-18.

Figure 6-15  Measurement Grid Locations:  a) Locations Used in Conventional NAH Reconstruction and b) Locations Used in ENAH Reconstruction
Figure 6-16   Reference Pressure at the 2 cm Estimation Plane

Figure 6-17   Reconstructions:  a) from 11 x 17 Array of Pressure Measurements Using NAH and b) from 6 x 9 Array of Energy-based Measurements Using ENAH
Figures 6-17 and 6-18 indicate that both reconstruction techniques are able to produce accurate estimations of the pressure field at 2 cm. The normalized whole field error for the conventional NAH reconstruction is 0.051. The corresponding error for the ENAH estimation is 0.039. The ENAH technique yielded slightly better reconstruction results and required 71% fewer measurement locations.

6.2 CYLINDRICAL TEST CASE

Much of the procedure followed in the planar experimental validation is the same for the cylindrical test case. This section will therefore be limited to explaining the aspects specific to the cylindrical geometry. These aspects primarily involve the experimental setup and data acquisition procedures. The reconstruction results are
presented along with a brief discussion regarding their correlation to the analytical and planar test case results.

6.2.1 Setup

An experimental setup was designed to approximate a simply supported cylinder. A cylindrical ABS plastic tube was used for this test case. The tube dimensions are: 10.2 cm inner diameter, 10.8 cm outer diameter, and 50 cm length. Simply supported boundary conditions were approximated at the tube ends using tapered conical plugs. The taper of the plugs provided effectual “knife-edge” constraints. The tube was driven at 1524 Hz with the same 20 mm diameter piezoelectric patch used for the plate. The patch was placed at the vertical center of the cylinder. This frequency corresponds to the $3, 3 (z, \phi)$ mode of the cylinder.

The cylinder was attached to a geared motor in a fully anechoic chamber as shown in Figure 6-19. The motor allowed the cylinder to be rotated in specified degree increments. The Microflown USP sensor was attached to a vertical metal rod that remained stationary throughout the experiment. The cylinder was rotated in specified increments, with the USP sensor fixed in a vertical position. The USP sensor was then moved to a new vertical position and the rotations repeated.
6.2.2 Data Acquisition

A single USP was again used to scan the field to obtain the pressure and in-plane velocities required for reconstruction. Scans were made at 2 cm and 4 cm radial distances from the outer surface of the tube. The 2 cm measurement serves as the reference against which the reconstructions from the 4 cm measurement contour are compared. The vertical step distance was set to 10 cm and the incremental rotation angle was 27 degrees. The cylinder was overscanned in the \( z \)-direction yielding a 100 cm x 351° overall measurement array size. This 11 x 14 array resulted in a total of 154 measurement locations. The Data Physics SignalCalc 620 data acquisition system was used to take ten stable, one-second averages with a sample rate of 8124 Hz at each measurement location, resulting in a one Hz temporal frequency resolution.
6.2.3 Data Post Processing

As mentioned above, the post processing for the cylindrical test case follows the same procedure as for the planar example. In this case, the window width $w_{xw}$ was 0.10 meters and the wavenumber filter parameters were set to $k_c = 20$ and $\alpha = 0.1$.

6.2.4 Results

Figure 6-21 shows the reference pressure as measured on the 2 cm estimation contour. ENAH and conventional NAH were performed from the measurement at 4 cm. All reconstruction variables, such as window and filter parameters, were again held constant. An 11 x 14 array (154 measurement locations) of pressure measurements at 4 cm was used to reconstruct the pressure on the estimation contour using the traditional Fourier NAH method. This pressure only reconstruction is compared to the ENAH reconstruction using a 7 x 7 array (49 measurement locations) of energy-based measurements spanning the same area. Figure 6-20 shows the measurement array locations for each reconstruction technique.
Figure 6-20  Measurement Grid Locations:  a) Locations Used in Conventional NAH Reconstruction and b) Locations Used in ENAH Reconstruction

Figure 6-21  Reference Pressure at 2 cm Estimation Contour

The resulting reconstructions are shown in Figure 6-22. Plots of the pressure error for the reconstructions are also presented below in Figure 6-23. These figures indicate that both reconstruction techniques are again able to produce accurate
estimations of the pressure field at 2 cm. The normalized whole field error for the conventional NAH reconstruction is 0.030. The corresponding error for the ENAH estimation is 0.024. The ENAH technique yielded slightly better reconstruction results and required 68% fewer measurement locations.

Figure 6-22 Reconstructions: a) from 11 x 14 Array of Pressure Measurements Using NAH and b) from 7 x 7 Array of Energy-based Measurements Using ENAH

Figure 6-23 Pressure Error: a) for NAH Reconstruction and b) for ENAH Reconstruction
6.3 DISCUSSION OF RESULTS

The measurement location reduction seen in the experimental test cases are both about 70%. If a three channel energy-based probe such as the USP sensor is used, a 10% reduction in channel count is also realized. As would be expected, the experimental reduction in measurements fell short of the analytical prediction of 75% fewer measurements. A number of factors inherent in experimental measuring cause this. For NAH reconstructions, these factors include sensor positioning errors, calibration errors, source signal drift, and measurement noise. The ENAH technique developed was relatively insensitive to these errors and was able to realize most of the theoretical reduction.
7 CONCLUSIONS

This chapter summarizes the results from the analytical investigation and test cases. Recommendations are made for future work and development, and a review is conducted of previous and future publications relating to this research.

7.1 SUMMARY

A novel NAH method employing energy-based sensing has been developed to reduce inefficiencies found in current NAH measurement techniques. The inefficiencies addressed in this work relate mainly to the requirement that the sensor spacing be less than or equal to half a wavelength of the highest spatial frequency of interest. This requirement causes the number of measurement locations for mid to high frequency problems to become very large. An analytical investigation of the proposed ENAH reconstruction method indicated that a reduction in measurement locations of up to 75% is possible. The planar test case analyzed showed a measurement reduction of 71%, while the cylindrical test case yielded a reduction of 68%. These reductions are very significant when NAH is performed by scanning the field with a sub-array of sensors, because the sub-array does not have to be repositioned as many times to measure the field. This greatly reduces the time required, and therefore the expense, of using NAH as an analysis tool.
7.2 RECOMMENDATIONS

Many opportunities for future research into the expansion of the ENAH technique developed in this work are available. Investigations into the sensitivity of ENAH to sensor positioning errors, calibration errors, and so forth, would be very interesting and useful in determining the robustness of this method. The next logical step seems to be application to arbitrary geometry problems using IBEM or HELS algorithms. An algorithm specific to energy-based measurements could also be developed, that uses the pressure and velocity information as boundary values to numerically solve the radiation equations. This would be similar to the IBEM or HELS algorithms but would incorporate the additional velocity information. Further work in the area of energy-based sensing, such as a comparison between a pressure gradient sensor and the Microflown for measuring the particle velocity, would be very useful because ENAH relies on accurate particle velocity measurement. The Microflown sensor provided adequate particle velocity measurements in this research, but its high cost may be prohibitive for some users. A pressure gradient sensor is significantly easier to manufacture and is far less expensive. Another possible area of further development would be to determine a cost-effective method to integrate the radial velocity component into the reconstruction. Incorporation of this information could provide even greater measurement location reductions.

7.3 PUBLICATIONS

Portions of this research have been presented at the 147\textsuperscript{th} and 148\textsuperscript{th} Meetings of the Acoustical Society of America. A paper will be published in the proceedings of the 2005 Rocky Mountain Space Grant Consortium Fellowship Symposium which will be
held on May 9, 2005. A draft manuscript has been submitted for the 20th Biennial Conference on Mechanical Vibration and Noise to be held September 25-28, 2005. A journal article has also been submitted to the *Journal of the Acoustical Society of America* and is under review.
REFERENCES


APPENDIX
APPENDIX

This Appendix contains the MatLab code developed for this research. The codes for the analytical investigation and the experimental validations are provided for those interested in recreating the results presented in the body of the thesis or wishing to use them as a starting point for further research.

PROPAGATION CODE

This section contains the MatLab m-files used in the analytical and experiment investigations.

Analytical Implementation

The analytical study required the creation of three MatLab m-files. The first, ANALYTICALpressREF.m, computes the reference pressure field against which the NAH and ENAH reconstructions are compared. The planarNAH.m file performs conventional NAH. The planarENAH.m file implements the energy-based reconstruction technique. These three files were used to obtain the analytical results presented in Section 5.3.
clear;

F = 400; % excitation force amplitude [N]
f = 1094; % excitation frequency [Hz]
omega = 2*pi*f; % excitation frequency [rad/sec]
x_0 = 0.3048/2; % x position of point force [m]
y_0 = 0.4572/2; % y position of point force [m]
rho = 2700; % plate density [kg/m^3]
h = 0.003175; % plate thickness [m]
m_p = rho*h; % mass per unit area [kg/m^2]
E = 71000000000; % Young's Modulus [Pa]
u = 0.33; % Poisson's ratio
L_x = 0.3048; % x plate dimension [m]
L_y = 0.4572; % y plate dimension [m]

m = 1:1:20; % plate modes used in surface displacement
n = 1:1:20; % summation

dx = L_x/31; % Discretize plate into 31 point sources
dy = L_y/31; % in x and y
x = -L_x/2:dx:L_x/2; % location of x point sources for plate
y = -L_y/2:dy:L_y/2; % location of y point sources for plate

D = (E*h^3)/(12 - 12*u*u); % Plate Bending Stiffness

wsum = zeros(length(x),length(y)); % Initialize surface displacement matrix

% Sum contributions to surface displacement from each mode
for ii = 1:length(m)
    for jj = 1:length(n)
        omega_mn = sqrt(D/m_p)*((m(ii)*pi/L_x)^2 + (n(jj)*pi/L_y)^2);
        Omega_mn_x_0_y_0 = (2/sqrt(L_x*L_y))*sin(m(ii)*pi*x_0/L_x)*sin(n(jj)*pi*y_0/L_y);
        for kk = 1:length(x)
            for mm = 1:length(y)
                Omega_mn = (2/sqrt(L_x*L_y))*sin(m(ii)*pi/L_x*(x(kk)-L_x/2))*sin(n(jj)*pi/L_y*(y(mm)-L_y/2));
                w(kk,mm) = Omega_mn_x_0_y_0 * Omega_mn/(omega^2 - omega_mn^2);
            end
        end
        wsum = wsum + w;
    end
end
% Surface Displacement Profile
wfinal = -(F/rho*h)*wsum;

% Plot surface displacement profile
figure(1)
pcolor(x*100,y*100,wfinal')
view([0 90])
shading interp
axis image
xlabel('x (cm)')
ylabel('y (cm)')
title('Plate Surface Displacement')

rho_air = 1.21;     % density of air [kg/m^3]
S = dx * dy;        % surface point area
C = 343;            % speed of sound [m/s]
k = omega/c;        % acoustic wavenumber [m^-1]
Lx_array = -L_x:2*L_x/127:L_x;  % x field meas. locations
Ly_array = -L_y:2*L_y/127:L_y;  % y field meas. locations
[X,Y] = meshgrid(Lx_array,Ly_array);

H = 0.02;           % measurement standoff distance [m]
Psum = zeros(length(Lx_array),length(Ly_array));
% Initialize Pressure field matrix

% Discrete summation of Rayleigh's integral equation
for ii = 1:length(Lx_array)
    for jj = 1:length(Ly_array)
        for kk = 1:length(x)
            for mm = 1:length(y)
                r = sqrt((Lx_array(ii) - x(kk))^2 + (Ly_array(jj) - y(mm))^2 + H^2);
P(ii,jj) = -(omega*omega*rho_air/(2*pi)) * wfinal(kk,mm)*S*(cos(k*r) + i*sin(k*r))/r;
                Psum(ii,jj) = Psum(ii,jj) + P(ii,jj);
            end
        end
    end
end

Press = real(Psum)';

% Plot Reference Pressure field
figure(2)
pcolor(Lx_array*100,Ly_array*100,Press)
view([0 90])
axis image
caxis([-0.55 0.55])
colorbar
shading interp
xlabel('x (cm)')
ylabel('y (cm)')
title('Reference Pressure at 2 cm')

% Save Reference Pressure field to workspace
save referencePressure Press -ascii -double
planarNAH.m

This program performs conventional NAH on the field radiated from a simply supported rectangular plate. The referencePressure text file is loaded to compare against the reconstruction. The normalized whole field error is printed to the screen. The number of x and y sensors may be varied.

```matlab
clear;

F = 400;            % excitation force amplitude [N]
f = 1094;           % excitation frequency [Hz]
omega = 2*pi*f;     % excitation frequency [rad/sec]
x_0 = 0.3048/2;     % x position of point force [m]
y_0 = 0.4572/2;     % y position of point force [m]
rho = 2700;         % plate density [kg/m^3]
h = 0.003175;       % plate thickness [m]
m_p = rho*h;        % mass per unit area [kg/m^2]
E = 71000000000;    % Young's Modulus [Pa]
u = 0.33;           % Poisson's ratio
L_x = 0.3048;       % x plate dimension [m]
L_y = 0.4572;       % y plate dimension [m]

m = 1:1:20;         % plate modes used in surface displacement
n = 1:1:20;         % summation

dx = L_x/31;        % discretize plate into 31 point sources in x and y
dy = L_y/31;
x = -L_x/2:dx:L_x/2; % location of x point sources for plate
y = -L_y/2:dy:L_y/2; % location of y point sources for plate

D = (E*h^3)/(12 - 12*u*u); % Plate Bending Stiffness

wsum = zeros(length(x),length(y)); % Initialize surface displacement matrix

% Sum contributions to surface displacement from each mode
for ii = 1:length(m)
    for jj = 1:length(n)
        omega_mn = sqrt(D/m_p)*((m(ii)*pi/L_x)^2 + (n(jj)*pi/L_y)^2);
        Omega_mn_x_0_y_0 = (2/sqrt(L_x*L_y))*sin(m(ii)*pi/L_x*(x_0 - L_x/2))... *sin(n(jj)*pi/L_y*(y_0 - L_y/2));
        for kk = 1:length(x)
            for mm = 1:length(y)
                Omega_mn = (2/sqrt(L_x*L_y))*sin(m(ii)*pi/L_x*(x(kk) - L_x/2))... *sin(n(jj)*pi/L_y*(y(mm) - L_y/2));
                w(kk,mm) = Omega_mn_x_0_y_0 * Omega_mn/(omega^2 - omega_mn^2);
            end
            end
        end
        wsum = wsum + w;
    end
    end
end
```
%Surface Displacement Profile
wfinal = -(F/rho*h)*wsum;

%Constants
rho_air = 1.21;     %density of air [kg/m^3]
S = dx * dy;        %surface point area
c = 343;            %speed of sound [m/s]
k = omega/c;        %acoustic wavenumber [m^-1]

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%MEASUREMENT ARRAY SIZE
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
xensors = 20;
yensors = 20;
Lx_array = -L_x:(2*L_x)/(xsensors - 1):L_x;  %x measurement array
Ly_array = -L_y:(2*L_y)/(ysensors - 1):L_y;  %y measurement array
[X,Y] = meshgrid(Lx_array,Ly_array);
H = 0.05;           %measurement standoff distance [m]
estplane = 0.02;    %estimation plane [m]

%Initialize pressure field matrix
Psum = zeros(length(Lx_array),length(Ly_array));

%Compute pressure field at measurement locations
for ii = 1:length(Lx_array)
    for jj = 1:length(Ly_array)
        for kk = 1:length(x)
            for mm = 1:length(y)
                r = sqrt((Lx_array(ii) - x(kk))^2 +...
                    (Ly_array(jj) - y(mm))^2 + H^2);
                P(ii,jj) = -(omega*omega*rho_air/(2*pi))*wfinal(kk,mm)*...
                    S*(cos(k*r) + i*sin(k*r))/r;
                Psum(ii,jj) = Psum(ii,jj) + P(ii,jj);
            end
        end
    end
end

%Create array for desired interpolation points
xarraysize = 2*L_x;
yarraysize = 2*L_y;
newxarray = -L_x:xarraysize/127:L_x;
newyarray = -L_y:yarraysize/127:L_y;

%Interpolate pressure field using linear interpolation
P = interp2(Lx_array,Ly_array,real(Psum'),newxarray,newyarray','linear');
P = P';
% Apply Tukey window
x_w = 0.15; % width of the spatial window taper [m]

% apply to right edge
Lx_end = newxarray(length(newxarray));
Lx_start = newxarray(1);
Ly_end = newyarray(length(newyarray));
Ly_start = newyarray(1);
ii = 1;
while newxarray(ii) < Lx_end - x_w
    ii = ii + 1;
end
for ii = ii:length(newxarray)
    for jj = 1:length(newyarray)
        P(ii,jj) = P(ii,jj)*
            (0.5 - 0.5*cos(pi*(newxarray(ii) - Lx_end)/x_w));
    end
end

% apply to top edge
ii = 1;
while newyarray(ii) < Ly_end - x_w
    ii = ii + 1;
end
for ii = ii:length(newyarray)
    for jj = 1:length(newxarray)
        P(jj,ii) = P(jj,ii)*
            (0.5 - 0.5*cos(pi*(newyarray(ii) - Ly_end)/x_w));
    end
end

% apply to left edge
ii = 1;
while newxarray(ii) < Lx_start + x_w
    for jj = 1:length(newyarray)
        P(ii,jj) = P(ii,jj)*
            (0.5 - 0.5*cos(pi*(newxarray(ii) - Lx_start)/x_w));
    end
    ii = ii + 1;
end

% apply to bottom edge
ii = 1;
while newyarray(ii) < Ly_start + x_w
    for jj = 1:length(newxarray)
        P(jj,ii) = P(jj,ii)*
            (0.5 - 0.5*cos(pi*(newyarray(ii) - Ly_start)/x_w));
    end
    ii = ii + 1;
end
% 2-D SPATIAL FFT
%Pad with zeros to 256 by 256
newP = zeros(256,256);
for ii = 64:191
    for jj = 64:191
        newP(ii,jj) = P(ii-63,jj-63);
    end
end

%Take 2-D FFT
Pkspace = (4/(256*256)) * fft2(newP);

%Create kx and ky arrays
deltakx = (2*pi)/(4*Lx_end);
deltaky = (2*pi)/(4*Ly_end);
incr = [0:1:128,zeros(1,127)];
kx = deltakx .* incr;
ky = deltaky .* incr;
xx = 2;
for ii = 130:256
    kx(1,ii) = -kx(1,ii-xx);
    ky(1,ii) = -ky(1,ii-xx);
    xx = xx + 2;
end

%Calculate kz
for ii = 1:length(incr)
    for jj = 1:length(incr)
        if kx(ii)*kx(ii) + ky(jj)*ky(jj) <= k*k
            kz(ii,jj) = sqrt(k*k - kx(ii)*kx(ii) - ky(jj)*ky(jj));
        else
            kz(ii,jj) = i*sqrt(kx(ii)*kx(ii) + ky(jj)*ky(jj) - k*k);
        end
    end
end

%Apply k-Space window
k_c = 75; %cutoff wavenumber
alpha = 0.2; %control of filter decay rate
%create windowing function
for ii = 1:length(incr)
    for jj = 1:length(incr)
        if sqrt(kx(ii)*kx(ii) + ky(jj)*ky(jj)) < k_c
            K_w(ii,jj) = 1 - 0.5*exp(-((sqrt(kx(ii)*kx(ii) + ky(jj)*ky(jj)))/k_c))/alpha;
        else
            K_w(ii,jj) = 0.5*exp((sqrt(kx(ii)*kx(ii) + ky(jj)*ky(jj)))/k_c))/alpha;
        end
    end
end

Pknewz = zeros(256,256);
%Multiply by inverse Green's function
for ii = 1:length(incr)
for jj = 1:length(incr)
    Pknewz(ii,jj) = Pkspace(ii,jj)*exp(-i*kz(ii,jj)*
        (H - estplane));
end
end

%Multiply k-Space spectrum by k-Space filter
Pknewz = Pknewz.*K_w;

%Inverse spatial 2D FFT
p_estimation = ((256*256)/4)*ifft2(Pknewz);
newxsize = -2*L_x:4*L_x/255:2*L_x;
realp_est = real(p_estimation);

%Adjust size of realp_est back to original measurement array size
for ii = 64:191
    for jj = 64:191
        newP2(ii-63,jj-63) = realp_est(ii,jj);
    end
end

%Plot estimated pressure field
figure(10)
pcolor(newxarray*100,newyarray*100,newP2)
title('Estimated Pressure from NAH')
xlabel('x (cm)')
ylabel('y (cm)')
shading interp
caxis([-0.55 0.55])
axis image
colorbar
view([0 90])

%Load reference pressure and compute residuals
ref = load('referencePressure');
residuals = ref - newP2;

%Plot pressure error
figure(6)
pcolor(newxarray*100,newyarray*100,residuals)
title('Pressure Error from NAH')
xlabel('x (cm)')
ylabel('y (cm)')
shading interp
axis image
caxis([-0.55 0.55])
colorbar
view([0 90])

%Print normalized whole field error to screen
error = std(std(residuals))/max(max(abs(ref)))
planarENAH.m

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%This program performs ENAH on the field radiated from a simply supported rectangular plate. The referencePressure text file is loaded to compare against the reconstruction. The normalized whole field error is printed to the screen. The number of x and y sensors may be varied.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear;

F = 400; % excitation force amplitude [N]
f = 1094; % excitation frequency [Hz]
omega = 2*pi*f; % excitation frequency [rad/sec]
x_0 = 0.3048/2; % x position of point force [m]
y_0 = 0.4572/2; % y position of point force [m]
rho = 2700; % plate density [kg/m^3]
h = 0.003175; % plate thickness [m]
m_p = rho*h; % mass per unit area [kg/m^2]
E = 71000000000; % Young's Modulus [Pa]
nu = 0.33; % Poisson's ratio
L_x = 0.3048; % x plate dimension [m]
L_y = 0.4572; % y plate dimension [m]

m = 1:1:20; % plate modes used in surface displacement
n = 1:1:20; % summation

dx = L_x/31; % Discretize plate into 31 point sources
dy = L_y/31; % in x and y
x = -L_x/2:dx:L_x/2; % location of x point sources for plate
y = -L_y/2:dy:L_y/2; % location of y point sources for plate

D = (E*h^3)/(12 - 12*nu*nu); % Plate Bending Stiffness

wsum = zeros(length(x),length(y)); % Initialize surface displacement %matrix

% Sum contributions to surface displacement from each mode
for ii = 1:length(m)
    for jj = 1:length(n)
        omega_mn = sqrt(D/m_p)*((m(ii)*pi/L_x)^2 + (n(jj)*pi/L_y)^2);
        Omega_mn_x_0_y_0 = (2/sqrt(L_x*L_y))*sin(m(ii)*pi*x_0/L_x)*...
            *sin(n(jj)*pi*y_0/L_y);
        for kk = 1:length(x)
            for mm = 1:length(y)
                Omega_mn = (2/sqrt(L_x*L_y))*sin(m(ii)*pi/L_x*(x(kk) - L_x/2))...
                    *sin(n(jj)*pi/L_y*(y(mm) - L_y/2));
                w(kk,mm) = Omega_mn_x_0_y_0 * Omega_mn/(omega^2 - omega_mn^2);
            end
        end
        wsum = wsum + w;
    end
end
% Surface Displacement Profile
\[ w_{\text{final}} = -(F/\rho h)*w_{\text{sum}} \]

% Constants
\[ \rho_{\text{air}} = 1.21; \quad \text{density of air [kg/m}^3] \]
\[ S = \text{dx} \times \text{dy}; \quad \text{surface point area} \]
\[ c = 343; \quad \text{speed of sound [m/s]} \]
\[ k = \omega/c; \quad \text{acoustic wavenumber [m}^{-1}] \]

%%%%%%%%%%%%%%%%%%%%%%%%%%
% MEASUREMENT ARRAY SIZE  
%%%%%%%%%%%%%%%%%%%%%%%%%%
xsensors = 20;
ysensors = 20;
Lx_array = -L_x:(2*L_x)/(xsensors - 1):L_x; \quad \% \text{x measurement array locations}
Ly_array = -L_y:(2*L_y)/(ysensors - 1):L_y; \quad \% \text{y measurement array locations}
[X,Y] = meshgrid(Lx_array,Ly_array);

% Create array of points needed to find pressure gradients for fitting
\[ \text{deltax} = 0.005; \]
\[ \text{jj} = 1; \]
\% for \text{ii} = 1:length(Lx_array)
\% \quad \text{xvalue(jj)} = Lx_array(ii) - \text{deltax};
\% \quad \text{xvalue(jj+1)} = Lx_array(ii) + \text{deltax};
\% \quad \text{jj} = \text{jj} + 2;
end
\% for \text{ii} = 1:length(Ly_array)
\% \quad \text{deltay} = 0.005;
\% \quad \text{jj} = 1;
\% for \text{jj} = 1:length(Ly_array)
\% \quad \text{yvalue(jj)} = Ly_array(jj) - \text{deltay};
\% \quad \text{yvalue(jj+1)} = Ly_array(jj) + \text{deltay};
\% \quad \text{jj} = \text{jj} + 2;
end
\[ H = 0.05; \quad \% \text{measurement standoff distance [m]} \]
\[ \text{estplane} = 0.02; \quad \% \text{estimation plane [m]} \]

% Initialize required pressure matrices
\[ \text{Psum} = \text{zeros(length(Lx_array),length(Ly_array))}; \]
\[ \text{Px1sum} = \text{zeros(length(Lx_array),length(Ly_array))}; \]
\[ \text{Px2sum} = \text{zeros(length(Lx_array),length(Ly_array))}; \]
\[ \text{Py1sum} = \text{zeros(length(Lx_array),length(Ly_array))}; \]
\[ \text{Py2sum} = \text{zeros(length(Lx_array),length(Ly_array))}; \]

% Create pressure field at measurements locations and necessary pressure gradients
\[ \text{xx} = 1; \]
\% for \text{ii} = 1:length(Lx_array)
\% \quad \text{yy} = 1;
\% \quad for \text{jj} = 1:length(Ly_array)
\% \% for \text{kk} = 1:length(x)
\% \% \quad for \text{mm} = 1:length(y)
\% \% \% \quad \text{r} = \sqrt{(Lx_array(jj) - x(kk))^2 + ...
\% \% \% \quad (Ly_array(jj) - y(mm))^2 + H^2);}
P(ii,jj) = -(omega*omega*rho_air/(2*pi))*...
    wfinal(kk,mm)*S*(cos(k*r) + i*sin(k*r))/r;
Psum(ii,jj) = Psum(ii,jj) + P(ii,jj);
rx1 = sqrt((xvalue(xx) - x(kk))^2 +...
    (Ly_array(jj) - y(mm))^2 + H^2);
rx2 = sqrt((xvalue(xx+1) - x(kk))^2 +...
    (Ly_array(jj) - y(mm))^2 + H^2);
ry1 = sqrt((Lx_array(ii) - x(kk))^2 +...
    (yvalue(yy) - y(mm))^2 + H^2);
ry2 = sqrt((Lx_array(ii) - x(kk))^2 +...
    (yvalue(yy+1) - y(mm))^2 + H^2);
Px1(ii,jj) = -(omega*omega*rho_air/(2*pi))*...
    wfinal(kk,mm)*S*(cos(k*rx1) + i*sin(k*rx1))/rx1;
Pxy2(ii,jj) = -(omega*omega*rho_air/(2*pi))*...
    wfinal(kk,mm)*S*(cos(k*rx2) + i*sin(k*rx2))/rx2;
Py1(ii,jj) = -(omega*omega*rho_air/(2*pi))*...
    wfinal(kk,mm)*S*(cos(k*ry1) + i*sin(k*ry1))/ry1;
Py2(ii,jj) = -(omega*omega*rho_air/(2*pi))*...
    wfinal(kk,mm)*S*(cos(k*ry2) + i*sin(k*ry2))/ry2;
Px1sum(ii,jj) = Px1sum(ii,jj) + Px1(ii,jj);
Pxy2sum(ii,jj) = Pxy2sum(ii,jj) + Pxy2(ii,jj);
Py1sum(ii,jj) = Py1sum(ii,jj) + Py1(ii,jj);
Py2sum(ii,jj) = Py2sum(ii,jj) + Py2(ii,jj);
end
end
yy = yy + 2;
end
xx = xx + 2;
end

% Compute pressure gradients
dPdx = (Px2sum - Px1sum)/(2*deltax);
dPdy = (Py2sum - Py1sum)/(2*deltay);
clear Px1sum Px2sum Py1sum Py2sum Px1 Pxy2 Py1 Py2;

% Hermite surface fitting
[row column] = size(Psum);
umxsegments = row - 1;
umysegments = column - 1;

Press = real(Psum);
dPdxreal = real(dPdx);
dPdyreal = real(dPdy);

% Hermite basis matrix
M = [2 -2 1 1;-3 3 -2 -1;0 0 1 0;1 0 0 0];

% Slope scaling factors
scale_t = abs(Lx_array(1)) - abs(Lx_array(2));
scale_s = abs(Ly_array(1)) - abs(Ly_array(2));
yarraysize = 2*L_y;
xarraysize = 2*L_x;
newxarray = -L*xarraysize/127:L_x;
newyarray = -L*yarraysize/127:L_y;
P = zeros(length(newxarray),length(newyarray));
for ii = 1:numxsegments
    for jj = 1:numysegments
        Q = [Press(ii,jj) Press(ii+1,jj) dPdxreal(ii,jj)*scale_t
             dPdxreal(ii+1,jj)*scale_t;...
             Press(ii,jj+1) Press(ii+1,jj+1) dPdxreal(ii,jj+1)*scale_t
             dPdxreal(ii+1,jj+1)*scale_t;...
             dPdyreal(ii,jj)*scale_s dPdyreal(ii+1,jj)*scale_s 0 0;...
             dPdyreal(ii,jj+1)*scale_s dPdyreal(ii+1,jj+1)*scale_s 0 0];
        mm = 1;
        while newyarray(mm) < Ly_array(jj)
            mm = mm + 1;
        end
        while mm <= length(newyarray) && newyarray(mm) >= Ly_array(jj) &&
            newyarray(mm) < Ly_array(jj+1)
            S(mm) = (newyarray(mm) - Ly_array(jj))/(Ly_array(jj+1) -
                    Ly_array(jj));
            kk = 1;
            while newxarray(kk) < Lx_array(ii)
                kk = kk + 1;
            end
            while kk <= length(newxarray) && newxarray(kk) >=
                Lx_array(ii) && newxarray(kk) < Lx_array(ii+1)
                T(kk) = (newxarray(kk) - Lx_array(ii))/(Lx_array(ii+1) -
                        Lx_array(ii));
                P(kk,mm) = [S(mm)^3 S(mm)^2 S(mm) 1]*M*Q*M'*[T(kk)^3;T(kk)^2;T(kk);1];
                kk = kk + 1;
            end
        end
        mm = mm + 1;
    end
end
P = P';

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%NAH RECONSTRUCTION ALGORITHM   %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Apply Tukey window
x_w = 0.15;         %width of the spatial window taper [m]

%apply to right edge
Lx_end = newxarray(length(newxarray));
Lx_start = newxarray(1);
Ly_end = newyarray(length(newyarray));
Ly_start = newyarray(1);
ii = 1;
while newxarray(ii) < Lx_end - x_w
    ii = ii + 1;
end
for ii = ii:length(newxarray)
    for jj = 1:length(newyarray)
        P(ii,jj) = P(ii,jj)*...
            (0.5 - 0.5*cos(pi*(newxarray(ii) - Lx_end)/x_w));
    end
end

%apply to top edge
ii = 1;
while newyarray(ii) < Ly_end - x_w
    ii = ii + 1;
end
for ii = ii:length(newyarray)
    for jj = 1:length(newxarray)
        P(jj,ii) = P(jj,ii)*...
            (0.5 - 0.5*cos(pi*(newyarray(ii) - Ly_end)/x_w));
    end
end

%apply to left edge
ii = 1;
while newxarray(ii) <  Lx_start + x_w
    ii = ii + 1;
end
for ii = 1:length(newxarray)
    for jj = 1:length(newyarray)
        P(ii,jj) = P(ii,jj)*...
            (0.5 - 0.5*cos(pi*(newxarray(ii) - Lx_start)/x_w));
    end
ii = ii + 1;
end

%%%%%%%%%%%%%%%%%%%%%%%
% 2-D SPATIAL FFT    %
%%%%%%%%%%%%%%%%%%%%%%%
%Pad with zeros to 256 by 256
newP = zeros(256,256);
for ii = 64:191
    for jj = 64:191
        newP(ii,jj) = P(ii-63,jj-63);
    end
end

%Take 2-D FFT
Pkspace = (4/(256*256)) * fft2(newP);

%Create kx and ky arrays
deltakx = (2*pi)/(4*Lx_end);
deltaky = (2*pi)/(4*Ly_end);
incr = [0:1:128,zeros(1,127)];
kx = deltax .* incr;
y = deltay .* incr;
xx = 2;
for ii = 130:256
    kx(l,ii) = -kx(l,ii-xx);
y(l,ii) = -y(l,ii-xx);
    xx = xx + 2;
end

%Calculate kz
for ii = 1:length(incr)
    for jj = 1:length(incr)
        if kx(ii)*kx(ii) + ky(jj)*ky(jj) <= k*k
            kz(ii,jj) = sqrt(k*k - kx(ii)*kx(ii) - ky(jj)*ky(jj));
        else
            kz(ii,jj) = i*sqrt(kx(ii)*kx(ii) + ky(jj)*ky(jj) - k*k);
        end
    end
end

%Apply k-Space window
k_c = 75;               %cutoff wavenumber
alpha = 0.2;            %control of filter decay rate
%create windowing function
for ii = 1:length(incr)
    for jj = 1:length(incr)
        if sqrt(kx(ii)*kx(ii) + ky(jj)*ky(jj)) < k_c
            K_w(ii,jj) = 1 - 0.5*exp(-(1 - (sqrt(kx(ii)*kx(ii) +... 
                       ky(jj)*ky(jj))/k_c))/alpha);
        else
            K_w(ii,jj) = 0.5*exp((1-(sqrt(kx(ii)*kx(ii) +... 
                       ky(jj)*ky(jj))/k_c))/alpha);
        end
    end
end

Pknewz = zeros(256,256);
%Multiply by inverse Green's function
for ii = 1:length(incr)
    for jj = 1:length(incr)
        Pknewz(ii,jj) = Pkspace(ii,jj)*exp(-i*kz(ii,jj)*... 
                        (H - estplane));
    end
end

%Multiply k-Space spectrum by k-Space filter
Pknewz = Pknewz.*K_w;

%Inverse spatial 2D FFT
p_estimation = ((256*256)/4)*ifft2(Pknewz);
newxsize = -2*L_x:4*L_x/255:2*L_x;
realp_est = real(p_estimation);
% Adjust size of realp_est back to original measurement array size
for ii = 64:191
    for jj = 64:191
        newP2(ii-63,jj-63) = realp_est(ii,jj);
    end
end

% Plot estimated pressure field
figure(10)
pcolor(newxarray*100,newyarray*100,newP2)
title('Estimated Pressure from ENAH')
xlabel('x (cm)')
ylabel('y (cm)')
shading interp
caxis([-0.55 0.55])
axis image
colorbar
view([0 90])

% Load reference pressure and compute residuals
ref = load('referencePressure');
residuals = ref - newP2;

% Plot pressure error
figure(6)
pcolor(newxarray*100,newyarray*100,residuals)
title('Pressure Error from ENAH')
xlabel('x (cm)')
ylabel('y (cm)')
shading interp
axis image
caxis([-0.55 0.55])
colorbar
view([0 90])

% Print normalized whole field error to screen
error = std(std(residuals))/max(max(abs(ref)))

---

**Planar Test Case**

The planar test case requires three separate files to obtain the results presented in Section 6.1.4. The first, pressREF.m, generates the reference pressure field from the 2 cm measurement against which the NAH and ENAH reconstructions are compared and must be located in the directory that contains the 2 cm data. The planarNAHexp.m file performs conventional NAH from the 5 cm measurement data. The planarENAHexp.m file implements the energy-based reconstruction technique. The reconstruction files must
be located in the directory containing the 5 cm measurement data. The ascii text file NAHref created by pressREF.m must also be placed in the directory with the reconstruction files because they load this file for the error comparison.

pressREF.m

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%This program creates the reference pressure field against which conventional NAH (planarNAH.m) and ENAH (planarENAH) reconstructions are compared. An output file, NAHref, is created that is read into the NAH and ENAH reconstruction codes.
%NOTE: This file must be located in the directory with the sampled pressure and velocity data folders.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear;

freq = 1090; % Forcing frequency [Hz]
rho = 1.21; % Air density [kg/m^3]
omega = 2*pi*freq; % Forcing frequency [rad/sec]

%Extract pressure and velocity data at the forcing frequency
for ii = 1:187
get pressure data
eval([ strcat(['addpath 1090ASCII00' num2str(ii) ]) ]);  
[A,B] = textread('G1, 1sv00000.txt', '%f %f', -1, 'headerlines',7);  
pressamp(ii) = B(freq+1);  
if A,B] = textread('G5, 1sv00000.txt', '%f %f %f', -1, 'headerlines',7);  
pressphi(ii) = atan2(B(freq+1),A(freq+1));
%get x velocity data 
[A,B] = textread('G2, 2sv00000.txt', '%f %f', -1, 'headerlines',7);  
v_xamp(ii) = B(freq+1);  
[v_xamp(ii) = B(freq+1)];  
[v_xphi(ii) = atan2(B(freq+1),A(freq+1));
%get y velocity data 
[A,B] = textread('G4, 4sv00000.txt', '%f %f', -1, 'headerlines',7);  
v_yamp(ii) = B(freq+1);  
[v_yamp(ii) = B(freq+1)];  
[v_yphi(ii) = atan2(B(freq+1),A(freq+1));
end

%Amplitude and phase correction for pressure
fCF1 = 56; % Hz
Sens = 0.0153; % V/Pa
S_mic = Sens/(sqrt(1 + (fCF1/freq)^2)); % V/Pa
phase_mic = atan2(56^2,1090); % rad
%Amplitude and phase correction for vel_x
LFS = 27.5; %V/(m/s)
fcf1 = 102; %Hz
fcf2 = 1200; %Hz
fcf3 = 13000; %Hz
C1 = 102;
C2 = 1300;
C3 = 12000;
S_velx = LFS/(sqrt(1+(fcf1/freq)^2)*sqrt(1+(freq/fcf2)^2)*sqrt(1+(freq/fcf3)^2)); %V/(m/s)
phase_velx = atan2(C1,freq) - atan2(freq,C2) - atan2(freq,C3);

%Amplitude and phase correction for vel_y
LFS = 35.8; %V/(m/s)
fcf1 = 103; %Hz
fcf2 = 1200; %Hz
fcf3 = 13000; %Hz
C1 = 103;
C2 = 1200;
C3 = 10000;
S_vely = LFS/(sqrt(1+(fcf1/freq)^2)*sqrt(1+(freq/fcf2)^2)*sqrt(1+(freq/fcf3)^2)); %V/(m/s)
phase_vely = atan2(C1,freq) - atan2(freq,C2) - atan2(freq,C3);

%Zero phase points
pressphiref = pressphi(1)*phase_mic;
v_xphiref = v_xphi(1)*phase_velx;
v_yphiref = v_yphi(1)*phase_vely;

%Measurement array parameters
dx = 2;
dy = 2;
xmeasarraysize = 20; %size of measurement array in x (inches)
ymeasarraysize = 32; %size of measurement array in y (inches)
x = -xmeasarraysize/2:dx:xmeasarraysize/2;
y = -ym easarraysize/2:dy:ym easarraysize/2;
kk = 1;

%Create pressure, velocity, and all phase matrices
for ii = 1:length(x)
    for jj = 1:length(y)
        %pressure
        pamplitude(ii,jj) = pressamp(kk)/S_mic;
        prelphase(ii,jj) = pressphi(kk)*phase_mic - pressphiref;
        %x velocity
        dpdx_amp(ii,jj) = v_xamp(kk)*omega*rho/S_velx;
        v_xrelphase(ii,jj) = v_xphi(kk)*phase_velx - v_xphiref;
        %y velocity
        dpdy_amp(ii,jj) = v_yamp(kk)*omega*rho/S_vely;
        v_yrelphase(ii,jj) = v_yphi(kk)*phase_vely - v_yphiref;
        kk = kk + 1;
    end
end
incr = 0.25;
pamplitudenew = (pamplitude.*cos(prelphase+incr));
dpdx_ampnew = dpdx_amp.*cos(v_xrelphase+incr);
\[
\text{dpdy}\_\text{ampnew} = \text{dpdy}\_\text{amp} \cdot \cos(v\_\text{yrelphase} + \text{incr} + \pi);
\]

%Hermite surface fitting
[row column] = size(pamplitudenew);
numxsegments = row - 1;
numysegments = column - 1;
Press = pamplitudenew;
dPdxreal = dpdx\_ampnew;
dPdyreal = dpdy\_ampnew;

Hermite basis matrix
M = [2 -2 1 1; -3 3 -2 -1; 0 0 1 0; 1 0 0 0];

%Slope scaling factors
Lx\_array = x*0.0254;
Ly\_array = y*0.0254;
scale_t = (abs(Lx\_array(1)) - abs(Lx\_array(2)));
scale_s = (abs(Ly\_array(1)) - abs(Ly\_array(2)));
xarraysize = Lx\_array(length(Lx\_array)) - Lx\_array(1);
yarraysize = Ly\_array(length(Ly\_array)) - Ly\_array(1);
dxarraysize = xarraysize/127;
dyarraysize = yarraysize/127;
newxarray = Lx\_array(1):dxarraysize:Lx\_array(length(Lx\_array));
newyarray = Ly\_array(1):dyarraysize:Ly\_array(length(Ly\_array));
P = zeros(length(newxarray),length(newyarray));

for jj = 1:numxsegments
    for kk = 1:numysegments
        Q = [Press(jj,kk) Press(jj+1,kk) ...
             dPdxreal(jj,kk)*scale_t dPdxreal(jj+1,kk)*scale_t; ...
             Press(jj,kk+1) Press(jj+1,kk+1) ...
             dPdyreal(jj,kk)*scale_s dPdyreal(jj+1,kk)*scale_s; ...
             dPdyreal(jj,kk+1)*scale_s dPdyreal(jj+1,kk+1)*scale_s];
        mm = 1;
        while newyarray(mm) < Ly\_array(kk)
            mm = mm + 1;
        end
        while mm <= length(newyarray) && newyarray(mm) > Ly\_array(kk)
            S(mm) = (newyarray(mm) - Ly\_array(kk)) / (Ly\_array(kk+1) - Ly\_array(kk));
        end
        nn = 1;
        while newxarray(nn) < Lx\_array(jj)
            nn = nn + 1;
        end
        while nn <= length(newxarray) && newxarray(nn) > Lx\_array(jj)
            T(nn) = (newxarray(nn) - Lx\_array(jj)) / (Lx\_array(jj+1) - Lx\_array(jj));
            P(nn,mm) = [S(mm)^3 S(mm)^2 S(mm) 1]*M*Q*[T(nn)^3;T(nn)^2;T(nn);1];
            nn = nn + 1;
        end
        mm = mm + 1;
    end
end

110
```matlab
P = P';

%Save NAHref to workspace
save NAHref P -ascii -double

planarNAHexp.m

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%This program performs conventional NAH on the measured pressure field %
%at 5 cm. The NAHref text file is loaded to compare against the %
%reconstruction. The normalized whole field error is printed to the %
%screen. %
%NOTE: This file must be located in the directory with the sampled %
%pressure and velocity data folders from the 5 cm measurement. %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear;

H = 0.05; %distance to measurement plane [m]
estplane = 0.026; %distance to estimation plane [m]
c = 343; %speed of sound in air [m/s]
freq = 1090; %frequency of excitation [Hz]
rho = 1.21; %density of air [kg/m^3]
omega = 2*pi*freq; %angular frequency [rad/sec]
k = omega/c; %acoustic wavenumber [m^-1]

for ii = 1:187
  %Read in pressure data
  [A,B] = textread('G1, 1sv00000.txt', '%f %f', -1, 'headerlines', 7);
  pressamp(ii) = B(freq+1);
  [f,A,B] = textread('G5, 1sv00000.txt', '%f %f %f', -1, 'headerlines', 7);
  pressphi(ii) = atan2(B(freq+1),A(freq+1));
  eval([ strcat([ 'rmpath 1090ASCII00' num2str(ii) ])]);
end

%Amplitude and phase correction for pressure
fCF1 = 56; %Hz
Sens = 0.0153; %V/Pa
S_mic = Sens/(sqrt(1 + (fCF1/freq)^2)); %V/Pa
phase_mic = atan2(56^2,1090); %radians

%Zero phase reference phase
pressphiref = pressphi(1)*phase_mic;

dx = 2;
dy = 2;
xmeasarraysize = 20; %size of measurement array in x (inches)
ymeasarraysize = 32; %size of measurement array in y (inches)
x = -xmeasarraysize/2:dx:xmeasarraysize/2;
```

111
y = -ymeasarraysize/2:dy:ymeasarraysize/2;
kk = 1;

%Create pressure and phase matrices
for ii = 1:length(x)
    for jj = 1:length(y)
        %pressure
        pamplitude(ii,jj) = pressamp(kk)/S_mic;
        prelphase(ii,jj) = pressphi(kk)*phase_mic - pressphiref;
        kk = kk + 1;
    end
end

Lx_array = x*0.0254;
Ly_array = y*0.0254;
xarraysize = Lx_array(length(Lx_array))-Lx_array(1);
yarraysize = Ly_array(length(Ly_array))-Ly_array(1);
dxarraysize = xarraysize/127;
dyarraysize = yarraysize/127;
newxarray = Lx_array(1):dxarraysize:Lx_array(length(Lx_array));
newyarray = Ly_array(1):dyarraysize:Ly_array(length(Ly_array));

phaseincr = 0.25;
pamplitudenew = (pamplitude.*cos(prelphase+phaseincr));

%Linear interpolation between data points
P = interp2(Lx_array,Ly_array,pamplitudenew',newxarray,newyarray','linear') ;

%%%%%%%%%%%%%%%%%%%%%%%%
%NAH RECONSTRUCTION %
%%%%%%%%%%%%%%%%%%%%%%%%

%Apply Tukey window
x_w = 0.15;        %width of the spatial window taper [m]

%Apply to right edge
Lx_end = newxarray(length(newxarray));
Lx_start = newxarray(1);
Ly_end = newyarray(length(newyarray));
Ly_start = newyarray(1);
ii = 1;
while newxarray(ii) <  Lx_end - x_w
    ii = ii + 1;
end
for ii = ii:length(newxarray)
    for jj = 1:length(newyarray)
        P(ii,jj) = P(ii,jj) *...
                  (0.5 - 0.5*cos(pi*(newxarray(ii) - Lx_end)/x_w));
    end
end
%Apply to top edge
ii = 1;
while newyarray(ii) < Ly_end - x_w
    ii = ii + 1;
end
for ii = ii:length(newyarray)
    for jj = 1:length(newxarray)
        P(jj,ii) = P(jj,ii) * ...
                (0.5 - 0.5*cos(pi*(newyarray(ii) - Ly_end)/x_w));
    end
end

%Apply to left edge
ii = 1;
while newxarray(ii) <  Lx_start + x_w
    for jj = 1:length(newyarray)
        P(ii,jj) = P(ii,jj) * ...
                (0.5 - 0.5*cos(pi*(newxarray(ii) - Lx_start)/x_w));
    end
    ii = ii + 1;
end

%Apply to bottom edge
ii = 1;
while newyarray(ii) <  Ly_start + x_w
    for jj = 1:length(newxarray)
        P(jj,ii) = P(jj,ii) * ...
                (0.5 - 0.5*cos(pi*(newyarray(ii) - Ly_start)/x_w));
    end
    ii = ii + 1;
end

%%%%%%%%%%%%%%%%%%%%%%
% 2-D SPATIAL FFT    %
%%%%%%%%%%%%%%%%%%%%%%

%Pad with zeros to 256 by 256
newP = zeros(256,256);
for ii = 64:191
    for jj = 64:191
        newP(ii,jj) = P(ii-63,jj-63);
    end
end

%Take 2-D FFT
Pkspace = (4/(256*256)) * fft2(newP);

%Create kx and ky arrays
deltakx = (2*pi)/(4*Lx_end);
deltaky = (2*pi)/(4*Ly_end);
incr = [0:1:128,zeros(1,127)];
kx = deltakx .* incr;
ky = deltaky .* incr;
xx = 2;
for ii = 130:256
    kx(1,ii) = -kx(1,ii-xx);
    ky(1,ii) = -ky(1,ii-xx);
\[ xx = xx + 2; \]
end

%Create \( k_z \)
for \( ii = 1:\text{length}(\text{incr}) \)
    for \( jj = 1:\text{length}(\text{incr}) \)
        if \( k_x(ii)k_x(ii) + k_y(jj)k_y(jj) \leq k^2 \)
            \[ k_z(ii,jj) = \sqrt{k^2 - k_x(ii)k_x(ii) - k_y(jj)k_y(jj)}; \]
        else
            \[ k_z(ii,jj) = i\sqrt{k_x(ii)k_x(ii) + k_y(jj)k_y(jj) - k^2}; \]
        end
    end
end

%Apply k-Space window
\( k_c = 60; \)               %cutoff wavenumber
\( \alpha = 0.2; \)            %control of filter decay rate

%Create windowing function
for \( ii = 1:\text{length}(\text{incr}) \)
    for \( jj = 1:\text{length}(\text{incr}) \)
        if \( \sqrt{k_x(ii)k_x(ii) + k_y(jj)k_y(jj)} < k_c \)
            \[ K_w(ii,jj) = 1 - 0.5* \exp(-1 - (\sqrt{k_x(ii)k_x(ii) + k_y(jj)k_y(jj)})/k_c))/\alpha); \]
        else
            \[ K_w(ii,jj) = 0.5*\exp((-1-(\sqrt{k_x(ii)k_x(ii)})/k_c))/\alpha); \]
        end
    end
end

\[ P_{knewz} = \text{zeros}(256,256); \]
%Multiply by inverse green's function
for \( ii = 1:\text{length}(\text{incr}) \)
    for \( jj = 1:\text{length}(\text{incr}) \)
        \[ P_{knewz}(ii,jj) = P_{kspace}(ii,jj)*\exp(-i*k_z(ii,jj)*(H - \text{estplane})); \]
    end
end

%Multiply by k-Space filter
\( P_{knewz} = P_{knewz}.*K_w; \)

%Inverse spatial 2D FFT
\( p_{\text{estimation}} = ((256*256)/4)*\text{ifft2}(P_{knewz}); \)
newxsize = -
\[ \text{xmeasarraysize*0.0254:(2*\text{xmeasarraysize*0.0254})/255:}\text{xmeasarraysize*0.0254}; \]
newysize = -
\[ \text{ymeasarraysize*0.0254:(2*\text{ymeasarraysize*0.0254})/255:}\text{ymeasarraysize*0.0254}; \]
realp_est = real(p_{\text{estimation}});
%Adjust size of realp_est back to original measurement array size
for ii = 64:191
    for jj = 64:191
        newP2(ii-63,jj-63) = realp_est(ii,jj);
    end
end

%Load reference pressure
ref = load('NAHref');
residuals = ref - newP2;

%Print whole field error to the screen
error = std(std(residuals))/(max(max(abs(ref))))

planarENAHexp.m

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%This program performs energy-based NAH on the measured pressure field%
%at 5 cm. The NAHref text file is loaded to compare against the      %
%reconstruction. The normalized whole field error is printed to the %
%screen.                                                              

%NOTE: This file must be located in the directory with the sampled %
%pressure and velocity data folders from the 5 cm measurement.        
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear;

H = 0.05;               %distance to measurement plane [m]
estplane = 0.026;       %distance to estimation plane [m]
c = 343;                %speed of sound in air [m/s]
freq = 1090;            %frequency of excitation [Hz]
rho = 1.21;             %density of air [kg/m^3]
omega = 2*pi*freq;      %angular frequency [rad/sec]
k = omega/c;            %acoustic wavenumber [m^-1]

for ii = 1:187
    %Read in pressure data
    eval(['strcat([' addpath 1090ASCII00' num2str(ii) '] ) ]);
    [A,B] = textread('G1, 1sv00000.txt', '%f %f', -1, 'headerlines',
    7);
    pressamp(ii) = B(freq+1);
    [f,A,B] = textread('G5, 1sv00000.txt', '%f %f %f', -1, 'headerlines',
    7);
    pressphi(ii) = atan2(B(freq+1),A(freq+1));
    %Read in x velocity data
    [A,B] = textread('G2, 2sv00000.txt', '%f %f', -1, 'headerlines',
    7);
    v_xamp(ii) = B(freq+1);
    [f,A,B] = textread('G5, 2sv00000.txt', '%f %f %f', -1, 'headerlines',
    7);
    v_xphi(ii) = atan2(B(freq+1),A(freq+1));
    %Read in y velocity data
    [A,B] = textread('G4, 4sv00000.txt', '%f %f', -1, 'headerlines',
    7);
    v_yamp(ii) = B(freq+1);
    [f,A,B] = textread('G5, 4sv00000.txt', '%f %f %f', -1, 'headerlines',
    7);
v_yphi(ii) = atan2(B(freq+1),A(freq+1));
eval([ strcat([ 'rmpath 1090ASCII00' num2str(ii) ]) ]); end

% Amplitude and phase correction for pressure
fCF1 = 56; % Hz
Sens = 0.0153; % V/Pa
S_mic = Sens/(sqrt(1 + (fCF1/freq)^2)); % V/Pa
phase_mic = atan2(56^2,1090);

% Amplitude and phase correction for vel_x
LFS = 27.5; % V/(m/s)
fcf1 = 102; % Hz
fcf2 = 1200; % Hz
fcf3 = 13000; % Hz
C1 = 102;
C2 = 1300;
C3 = 12000;
S_velx = LFS/(sqrt(1+(fcf1/freq)^2)*sqrt(1+(freq/fcf2)^2)*sqrt(1+(freq/fcf3)^2)); % V/(m/s)
phase_velx = atan2(C1,freq) - atan2(freq,C2) - atan2(freq,C3);

% Amplitude and phase correction for vel_y
LFS = 35.8; % V/(m/s)
fcf1 = 103; % Hz
fcf2 = 1200; % Hz
fcf3 = 13000; % Hz
C1 = 103;
C2 = 1200;
C3 = 10000;
S_vely = LFS/(sqrt(1+(fcf1/freq)^2)*sqrt(1+(freq/fcf2)^2)*sqrt(1+(freq/fcf3)^2)); % V/(m/s)
phase_vely = atan2(C1,freq) - atan2(freq,C2) - atan2(freq,C3);

% Zero phase reference points
pressphiref = pressphi(1)*phase_mic;
v_xphiref = v_xphi(1)*phase_velx;
v_yphiref = v_yphi(1)*phase_vely;

dx = 2;
dy = 2;
xmeasarraysize = 20; % size of measurement array in x (inches)
ymeasarraysize = 32; % size of measurement array in y (inches)
x = -xmeasarraysize/2:dx:xmeasarraysize/2;
y = -ymeasarraysize/2:dy:ymeasarraysize/2;
kk = 1;

% Create pressure, velocity, and all phase matrices
for ii = 1:length(x)
  for jj = 1:length(y)
    % Pressure
    pamplitude(ii,jj) = pressamp(kk)/S_mic;
    prelphase(ii,jj) = pressphi(kk)*phase_mic - pressphiref;
    % x velocity
    dpdx_amp(ii,jj) = v_xamp(kk)*omega*rho/S_velx;
    v_xrelphase(ii,jj) = v_xphi(kk)*phase_velx - v_xphiref;
    % y velocity

\[ dpdy\_amp(ii,jj) = v\_yamp(kk)\cdot \omega\cdot \rho / S\_vely; \]
\[ v\_yrelphase(ii,jj) = v\_yphi(kk)\cdot \text{phase\_vely} - v\_yphiref; \]
\[ kk = kk + 1; \]
\end

%Adjust measurement array size to 9 x 6
dx = 4;
dy = 4;
xmeasarraysize = 20; %size of measurement array in x (inches)
ymeasarraysize = 32; %size of measurement array in y (inches)
x = -xmeasarraysize/2:dx:xmeasarraysize/2;
y = -ymeasarraysize/2:dy:ymeasarraysize/2;
kk = 1;

%Create new pressure, velocity, and all phase matrices
for ii = 1:length(x)
    mm = 1;
    for jj = 1:length(y)
        %Pressure
        pamplitude2(ii,jj) = pamplitude(kk,mm);
        prelphase2(ii,jj) = prelphase(kk,mm);
        %x velocity
        dpdx_amp2(ii,jj) = dpdx_amp(kk,mm);
        v\_xrelphase2(ii,jj) = v\_xrelphase(kk,mm);
        %y velocity
        dpdy_amp2(ii,jj) = dpdy_amp(kk,mm);
        v\_yrelphase2(ii,jj) = v\_yrelphase(kk,mm);
        mm = mm + 2;
    end
    kk = kk + 2;
end

phaseincr = 0.25;
pamplitudenew = (pamplitude2.*cos(prelphase2+phaseincr));
dpdx\_ampnew = dpdx\_amp2.*cos(v\_xrelphase2+phaseincr);
dpdy\_ampnew = dpdy\_amp2.*cos(v\_yrelphase2+phaseincr+\pi);

%Hermite surface fitting
[row column] = size(pamplitudenew);
numxsegments = row - 1;
numysegments = column - 1;
Press = pamplitudenew;
dPdxreal = dpdx\_ampnew;
dPdyreal = dpdy\_ampnew;

%Hermite basis matrix
M = \begin{bmatrix} 2 & -2 & 1 & 1; -3 & 3 & -2 & -1; 0 & 0 & 1 & 0; 1 & 0 & 0 & 0 \end{bmatrix};

%Slope scaling factors
Lx\_array = x*0.0254;
Ly\_array = y*0.0254;
scale\_t = (abs(Lx\_array(1)) - abs(Lx\_array(2)));
scale\_s = (abs(Ly\_array(1)) - abs(Ly\_array(2)));
xarraysize = Lx\_array(length(Lx\_array))-Lx\_array(1);
yarraysize = Ly\_array(length(Ly\_array))-Ly\_array(1);
dxarraysize = xarraysize/127;
dyarraysize = yarraysize/127;
newxarray = Lx_array(1):dxarraysize:Lx_array(length(Lx_array));
newyarray = Ly_array(1):dyarraysize:Ly_array(length(Ly_array));
P = zeros(length(newxarray),length(newyarray));

for jj = 1:numxsegments
    for kk = 1:numysegments
        Q = [Press(jj,kk) Press(jj+1,kk) dPdxreal(jj,kk)*... 
            scale_t dPdxreal(jj+1,kk)*scale_t;... 
            Press(jj,kk+1) Press(jj+1,kk+1) dPdxreal(jj,kk+1)*... 
            scale_t dPdxreal(jj+1,kk+1)*scale_t;... 
            dPdyreal(jj,kk)*scale_s dPdyreal(jj+1,kk)*scale_s 0 0;... 
            dPdyreal(jj,kk+1)*scale_s dPdyreal(jj+1,kk+1)*scale_s 0 0];
        
        mm = 1;
        while newyarray(mm) < Ly_array(kk)
            mm = mm + 1;
        end
        while mm <= length(newyarray) && newyarray(mm) > Ly_array(kk+1)
            S(mm) = (newyarray(mm) - Ly_array(kk))/... 
                (Ly_array(kk+1) - Ly_array(kk));
        end
        while nn <= length(newxarray) && newxarray(nn) > Lx_array(jj)
            T(nn) = (newxarray(nn) - Lx_array(jj))/... 
                (Lx_array(jj+1) - Lx_array(jj));
        end
        P(nn,mm) = [S(mm)^3 S(mm)^2 S(mm) 1]*M*... 
            Q*M'*[T(nn)^3;T(nn)^2;T(nn);1];
        
        nn = nn + 1;
    end
    mm = mm + 1;
end

P = P';

%%%%%%%%%%%%%%%%%%%%%%%%
%NAH RECONSTRUCTION    
%%%%%%%%%%%%%%%%%%%%%%%%
%Apply Tukey window

x_w = 0.15;         %width of the spatial window taper [m]

%Apply to right edge
Lx_end = newxarray(length(newxarray));
Lx_start = newxarray(1);
Ly_end = newyarray(length(newyarray));
Ly_start = newyarray(1);
ii = 1;
while newxarray(ii) < Lx_end - x_w
    ii = ii + 1;
end
for ii = ii:length(newxarray)
for jj = 1:length(newyarray)
    P(ii,jj) = P(ii,jj) * ...
             (0.5 - 0.5*cos(pi*(newxarray(ii) - Lx_end)/x_w));
end
end

%Apply to top edge
ii = 1;
while newyarray(ii) < Ly_end - x_w
    ii = ii + 1;
end
for ii = ii:length(newyarray)
    for jj = 1:length(newxarray)
        P(jj,ii) = P(jj,ii) * ...
                 (0.5 - 0.5*cos(pi*(newyarray(ii) - Ly_end)/x_w));
    end
end

%Apply to left edge
ii = 1;
while newxarray(ii) < Lx_start + x_w
    ii = ii + 1;
end
for ii = 1:length(newxarray)
    for jj = 1:length(newyarray)
        P(ii,jj) = P(ii,jj) * ...
                 (0.5 - 0.5*cos(pi*(newxarray(ii) - Lx_start)/x_w));
    end
end

%Apply to bottom edge
ii = 1;
while newyarray(ii) < Ly_start + x_w
    ii = ii + 1;
end

%Pad with zeros to 256 by 256
newP = zeros(256,256);
for ii = 64:191
    for jj = 64:191
        newP(ii,jj) = P(ii-63,jj-63);
    end
end

%Take 2-D FFT
Pkspace = (4/(256*256)) * fft2(newP);

%Create kx and ky arrays
deltakx = (2*pi)/(4*Lx_end);
deltaky = (2*pi)/(4*Ly_end);
incr = [0:1:128,zeros(1,127)];
kx = deltakx .* incr;
ky = deltaky .* incr;
xx = 2;
for ii = 130:256
    kx(1,ii) = -kx(1,ii-xx);
    ky(1,ii) = -ky(1,ii-xx);
    xx = xx + 2;
end

%Create kz
for ii = 1:length(incr)
    for jj = 1:length(incr)
        if kx(ii)*kx(ii) + ky(jj)*ky(jj) <= k*k
            kz(ii,jj) = sqrt(k*k - kx(ii)*kx(ii) - ky(jj)*ky(jj));
        else
            kz(ii,jj) = i*sqrt(kx(ii)*kx(ii) + ky(jj)*ky(jj) - k*k);
        end
    end
end

%Apply k-Space window
k_c = 60;               %cutoff wavenumber
alpha = 0.2;            %control of filter decay rate

%Create windowing function
for ii = 1:length(incr)
    for jj = 1:length(incr)
        if sqrt(kx(ii)*kx(ii) + ky(jj)*ky(jj)) < k_c
            K_w(ii,jj) = 1 - 0.5*... 
                        .exp(-(1 - (sqrt(kx(ii)*kx(ii) + 
                        ky(jj)*ky(jj))/k_c))/alpha);
        else
            K_w(ii,jj) = 0.5*exp((1-(sqrt(kx(ii)*kx(ii) + 
                        ky(jj)*ky(jj))/k_c))/alpha);
        end
    end
end

Pknewz = zeros(256,256);
%Multiply by inverse green's function
for ii = 1:length(incr)
    for jj = 1:length(incr)
        Pknewz(ii,jj) = Pkspace(ii,jj)*exp(-i*kz(ii,jj)*(H - 
estplane));
    end
end

%Multiply by k-Space filter
Pknewz = Pknewz.*K_w;

%Inverse spatial 2D FFT
p_estimation = ((256*256)/4)*ifft2(Pknewz);
newxsize = -
xmeasarraysize*0.0254:(2*xmeasarraysize*0.0254)/255:xmeasarraysize*0.0254;
newysize = -
ymeasarraysize*0.0254:(2*ymeasarraysize*0.0254)/255:ymeasarraysize*0.0254;

reallp_est = real(p_estimation);

%Adjust size of realp_est back to original measurement array size
for ii = 64:191
    for jj = 64:191
        newP2(ii-63,jj-63) = realp_est(ii,jj);
    end
end

%Load reference pressure
ref = load('NAHref');
residuals = ref - newP2;

%Print whole field error to the screen
error = std(std(residuals))/(max(max(abs(ref))))

---

**Cylindrical Test Case**

The cylindrical test case post processing requires four separate files to get the data ready for propagation using the ENAH algorithm. The first, finefield.m, creates a spatially dense pressure field from the 4 cm measurement that is sampled by makeCOARSEfield.m to obtain the matrices needed to create the propagation field, using propREADYfield.m. This propagation field is loaded by cylENAH.m, which then performs ENAH to reconstruct the field on the estimation plane. The file cylNAHref.m creates the reference pressure against which the reconstructions are compared. The conventional NAH implementation using cylNAH.m only requires that cylNAHref.m be executed before it may be used. It does not rely on makeCOARSEfield.m or propREADYfield.m.
finefield.m

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%This program creates the spatially dense pressure field to be used by %
%CYLpressREF.m and coarse2locations.                                  %
%NOTE:  This file must be located in the directory with the sampled %
%pressure and velocity data folders.                                  %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear;

freq = 1524;            %excitation frequency [Hz]
a = 2*0.0254;           %radius of cylinder [m]
r = 0.04 + a;           %standoff distance [m]
r_est = 0.02 + a;       %estimation distance [m]
rho = 1.21;             %density of air [kg/m^3]
omega = 2*pi*freq;      %excitation angular frequency [Hz]
c = 343;                %speed of sound [m/s]
k = omega/c;            %acoustic wavenumber [m^-1]

for ii = 1:154
  %Read in pressure data
  eval([ strcat([ 'addpath ASCII00' num2str(ii) ]) ]));
  [A,B] = textread('G1, 1sv00000.txt', '%f %f', -1, 'headerlines',
    7);
  pressamp(ii) = B(freq+1);
  [f,A,B] = textread('G5, 1sv00000.txt', '%f %f %f', -1, 
    'headerlines', 7);
  pressphi(ii) = atan2(B(freq+1),A(freq+1));
  %Read in z velocity data
  [A,B] = textread('G4, 4sv00000.txt', '%f %f', -1, 'headerlines',
    7);
  v_zamp(ii) = B(freq+1);
  [f,A,B] = textread('G5, 4sv00000.txt', '%f %f %f', -1, 
    'headerlines', 7);
  v_zphi(ii) = atan2(B(freq+1),A(freq+1));
  %Read in theta velocity data
  [A,B] = textread('G2, 2sv00000.txt', '%f %f', -1, 'headerlines',
    7);
  v_thetaamp(ii) = B(freq+1);
  [f,A,B] = textread('G5, 2sv00000.txt', '%f %f %f', -1, 
    'headerlines', 7);
  v_thetaphi(ii) = atan2(B(freq+1),A(freq+1));
  eval([ strcat([ 'rmpath ASCII00' num2str(ii) ]) ]));
end

%Amplitude and phase correction for pressure
fCF1 = 56;      %Hz
Sens = 0.0153;  %V/Pa
S_mic = Sens/(sqrt(1 + (fCF1/freq)^2));     %V/Pa
phase_mic = atan2(56^2,1090);

%Amplitude and phase correction for vel_theta
LFS = 27.5;      %V/(m/s)
fcf1 = 102;      %Hz
fcf2 = 1200;    %Hz
fcf3 = 13000;  %Hz
C1 = 102;
C2 = 1300;
C3 = 12000;
S_veltheta = LFS/(sqrt(1+(fcf1/freq)^2)*sqrt(1+(freq/fcf2)^2)*sqrt(1+(freq/fcf3)^2));   %V/(m/s)
phase_veltheta = atan2(C1,freq) - atan2(freq,C2) - atan2(freq,C3);

%Amplitude and phase correction for vel_z
LFS = 35.8;     %V/(m/s)
fcf1 = 103;     %Hz
fcf2 = 1200;    %Hz
fcf3 = 13000;   %Hz
C1 = 103;
C2 = 12000;
C3 = 10000;
S_velz = LFS/(sqrt(1+(fcf1/freq)^2)*sqrt(1+(freq/fcf2)^2)*sqrt(1+(freq/fcf3)^2));   %V/(m/s)
phase_velz = atan2(C1,freq) - atan2(freq,C2) - atan2(freq,C3);

%Zero phase reference point
pressphiref = pressphi(1)*phase_mic;
v_zphiref = v_zphi(1)*phase_velz;
v_thetaphiref = v_thetaphi(1)*phaseVeltheta;

dz = 4;
dtheta = 27;
z = -20:dz:20;
theta = 0:dtheta:351;
kk = 1;
%Create pressure, velocity, and all phase matrices
for ii = 1:length(z)
    for jj = 1:length(theta)
        %Pressure
        pamplitude(ii,jj) = pressamp(kk)/S_mic;
        prelphase(ii,jj) = pressphi(kk)*phase_mic - pressphiref;
        %z velocity
        dpdz_amp(ii,jj) = v_zamp(kk)*omega*rho/S_velz;
        v_zrelphase(ii,jj) = v_zphi(kk)*phase_velz - v_zphiref;
        %Theta velocity
        dpdtheta_amp(ii,jj) = v_thetaamp(kk)*omega*rho*r/S_veltheta;
        v_thetarelphase(ii,jj) = v_thetaphi(kk)*phase_veltheta - v_thetaphiref;
    
    kk = kk + 1;
end
end
pamplitudenew = (pamplitude.*cos(prelphase));
dpdz_amphnew = dpdz_amp.*cos(v_zrelphase);
dpdthetahamnew = dpdtheta_amp.*cos(v_thetarelphase+pi);

%Hermite surface fitting
[row column] = size(pamplitudenew);
numzsegments = row - 1;
numthetasegments = column;
Press = [pamplitudenew pamplitudenew(:,1)];
dPdzreal = [dpdz_amphnew dpdz_amphnew(:,1)];
\[ dPd\theta_{\text{real}} = [dpd\theta_{\text{amp new}} \ dpd\theta_{\text{amp new}}(:,1)]; \]

\% Herite basis matrix
\[ M = \begin{bmatrix} 2 & -2 & 1 & 1; -3 & 3 & -2 & -1; 0 & 0 & 1 & 0; 1 & 0 & 0 & 0 \end{bmatrix}; \]

\% Slope scaling factors
\[ \theta = [\theta 360]; \]
\[ L_{\theta \text{ array}} = z*0.0254; \]
\[ L_{\theta \text{ array}} = \theta; \]
\[ \text{scale}_t = (\text{abs}(L_{\theta \text{ array}}(1)) - \text{abs}(L_{\theta \text{ array}}(2)))*0.0254; \]
\[ \text{scale}_s = (\text{abs}(L_{\theta \text{ array}}(1)) - \text{abs}(L_{\theta \text{ array}}(2)))*\pi/180; \]
\[ \text{zarray size} = L_{\theta \text{ array}}(\text{length}(L_{\theta \text{ array}}))-L_{\theta \text{ array}}(1); \]
\[ \text{theta array size} = L_{\theta \text{ array}}(\text{length}(L_{\theta \text{ array}}))-L_{\theta \text{ array}}(1); \]
\[ \text{dzarray size} = \text{zarray size}/127; \]
\[ \text{dtheta array size} = \text{theta array size}/255; \]
\[ \text{newz array} = L_{\theta \text{ array}}(1):\text{dzarray size}:L_{\theta \text{ array}}(\text{length}(L_{\theta \text{ array}})); \]
\[ \text{newtheta array} = L_{\theta \text{ array}}(1):\text{dtheta array size}:L_{\theta \text{ array}}(\text{length}(L_{\theta \text{ array}})); \]
\[ P = \text{zeros}(\text{length(newz array)}, \text{length(newtheta array)}); \]

for \( jj = 1:\text{numz segments - 1} \)
for \( kk = 1:\text{numtheta segments} \)
\[ \text{scale}_s = (\text{abs}(L_{\theta \text{ array}}(kk)) - \text{abs}(L_{\theta \text{ array}}(kk+1)))*\pi/180; \]
\[ Q = \begin{bmatrix} \text{Press}(jj, kk) & \text{Press}(jj+1, kk) & \text{dPdz real}(jj, kk)*... \\
\text{scale}_t & \text{dPdz real}(jj+1, kk)*\text{scale}_t;... \\
\text{Press}(jj, kk+1) & \text{Press}(jj+1, kk+1) & \text{dPdz real}(jj, kk+1)*... \\
\text{scale}_t & \text{dPdz real}(jj+1, kk+1)*\text{scale}_t;... \\
\text{dPdtheta real}(jj, kk)*\text{scale}_s & \text{dPdtheta real}(jj+1, kk)*\text{scale}_s & 0;... \\
\text{dPdtheta real}(jj, kk+1)*\text{scale}_s & \text{dPdtheta real}(jj+1, kk+1)*\text{scale}_s & 0 \end{bmatrix}; \]
\[ mm = 1; \]
while newtheta array (mm) < L_{\theta \text{ array}} (kk) 
\[ mm = mm + 1; \]
end 
while mm <= length(newtheta array) && newtheta array (mm) >=...
\[ L_{\theta \text{ array}} (kk) && newtheta array (mm) < L_{\theta \text{ array}} (kk+1) \]
\[ S(mm) = (newtheta array (mm) - L_{\theta \text{ array}} (kk))/... \\
(L_{\theta \text{ array}} (kk+1) - L_{\theta \text{ array}} (kk)); \]
\[ nn = 1; \]
while newz array (nn) < L_{\theta \text{ array}} (jj) 
\[ nn = nn + 1; \]
end 
while nn <= length(newz array) && newz array (nn) >=...
\[ L_{\theta \text{ array}} (jj) && newz array (nn) < L_{\theta \text{ array}} (jj+1) \]
\[ T(nn) = (newz array (nn) - L_{\theta \text{ array}} (jj))/(L_{\theta \text{ array}} (jj+1) - L_{\theta \text{ array}} (jj)); \]
\[ P(nn, mm) = [S(mm)^{3} S(mm)^{2} S(mm)^{1}] * M * Q * M'; \]
end 
end

124
end

%Fill in last row of z
for jj = numzsegments
    for kk = 1:numthetasegments
        scale_s = (abs(Ltheta_array(kk)) -
                   abs(Ltheta_array(kk+1)))*pi/180;
        Q = Press(jj,kk) Press(jj+1,kk) dPdzreal(jj,kk)*...
           scale_t dPdzreal(jj+1,kk)*scale_t;...
           Press(jj,kk+1) Press(jj+1,kk+1) dPdzreal(jj,kk+1)*...
           scale_t dPdzreal(jj+1,kk+1)*scale_t;...
           dPdthetareal(jj,kk)*scale_s dPdthetareal(jj+1,kk)*scale_s 0
           0;...
           dPdthetareal(jj,kk+1)*scale_s dPdthetareal(jj+1,kk+1)*scale_s 0
           0];

        mm = 1;
        while newthetaarray(mm) < Ltheta_array(kk)
            mm = mm + 1;
        end
        while mm <= length(newthetaarray) && newthetaarray(mm) >=...
            Ltheta_array(kk) && newthetaarray(mm) <
            Ltheta_array(kk+1)
            S(mm) = (newthetaarray(mm) - Ltheta_array(kk))/...
                 (Ltheta_array(kk+1) - Ltheta_array(kk));
            nn = 1;
            while newzarray(nn) <= Lz_array(jj)
                nn = nn + 1;
            end
            while nn <= length(newzarray) && newzarray(nn) >=...
                Lz_array(jj) && newzarray(nn) <= Lz_array(jj+1)
                T(nn) = (newzarray(nn) - Lz_array(jj))/(Lz_array(jj+1
                - Lz_array(jj));
                P(nn,mm) = [S(mm)^3 S(mm)^2 S(mm)
                             1]*M*Q*M'*[T(nn)^3;T(nn)^2;T(nn);1];
                nn = nn + 1;
            end
            mm = mm + 1;
        end
end
P(:,length(P(1,:))) = P(:,1);

%Save finefield file
save finefield P -ascii -double

125
clear;

%Load finefield
field = load('finefield');
field(201,:) = field2(201,:);

%Sample the field at chosen locations
numzsensors = 6;
theta = 0:54:351;
LenZ = 40;
dz = (LenZ/numzsensors)*0.0254;
z = -0.508:dz:0.508;
dzfine = 0.200*0.0254; %meters
dthetafine = 1;
pamplitudenew = zeros(numzsensors+1,8);
k = 1;

%Populate pressure field matrix
for ii = 1:numzsensors+1
    for jj = 1:7
        mm = theta(jj) + 1;
        %Pressure
        pamplitudenew(ii,jj) = field(kk,mm);
        %Theta slope
        if mm == 1
            dpdtheta_amplitude(ii,jj) = (field(kk,mm) - field(kk,mm+1))/(dthetafine*pi/180);
        elseif mm < 361
            dpdtheta_amplitude(ii,jj) = (field(kk,mm) - field(kk,mm+1))/(dthetafine*pi/180);
        else
            dpdtheta_amplitude(ii,jj) = (field(kk,mm-1) - field(kk,mm))/(dthetafine*pi/180);
        end
        %z slope
        if kk == 1
            dpdz_amplitude(ii,jj) = (field(kk+1,mm) - field(kk,mm))/(dzfine);
        elseif kk < 201
            dpdz_amplitude(ii,jj) = (field(kk+1,mm) - field(kk,mm))/(dzfine);
        else
            dpdz_amplitude(ii,jj) = (field(kk,mm) - field(kk-1,mm))/(dzfine);
        end
    end
end
kk = floor(kk + (dz/dzfine));
pamplitudenew(:,8) = pamplitudenew(:,1);
dpdz_ampnew(:,8) = dpdz_ampnew(:,1);
dpdthetanew(:,8) = dpdthetanew(:,1);

%Save pressure and gradient files
save Pressure pamplitudenew -ascii -double
save dPdZ dpdz_ampnew -ascii -double
save dPdTheta dpdthetanew -ascii -double

propREADYfield.m

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%This program creates the propagation ready pressure field for %
]%cyLENAH.m.                                                        %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear;

%Sample the field at chosen locations
numzsensors = 6;
LenTheta = 360;
dthetafine = 1;
theta = 0:54:351;
LenZ = 40;
dz = (LenZ/numzsensors)*0.0254;
z = -0.508:dz:0.508;
dzfine = 0.200*0.0254;  %meters
%Read in sampled matrices
pamplitudenew = load('Pressure');
dpdz_ampnew = load('dPdZ');
dpdthetanew = load('dPdTheta');

%Hermite surface fitting
[row column] = size(pamplitudenew);
numzsegments = row - 1;
numthetasegments = column - 1;
Press = [pamplitudenew];
dPdzreal = [dpdz_ampnew];
dPdtheetareal = [dpdthetanew];

%Hermite basis matrix
M = [2 -2 1 1; -3 3 -2 -1; 0 0 1 0; 1 0 0 0];

%Slope scaling factors
theta = [theta 360];
Lz_array = z;
Lthetanew = theta;
scale_T = (abs(Lz_array(1)) - abs(Lz_array(2)));
scale_S = (abs(Lthetanew(1)) - abs(Lthetanew(2)))*pi/180;
zarraysize = Lz_array(length(Lz_array))-Lz_array(1);
thetarraysize = Lthetanew(length(Lthetanew))-Lthetanew(1);
dzarraysize = zarraysize/127;
dthetaarraysize = thetarraysize/255;
newzarray = Lz_array(1):dzarraysize:Lz_array(length(Lz_array));
newthetanew = Lthetanew(1):dthetaarraysize:Lthetanew(length(Lthetanew));
P = zeros(length(newzarray), length(newthetaarray));
for jj = 1:numzsegments
    for kk = 1:numthetasegments
        scale_s = (abs(Ltheta_array(kk)) - abs(Ltheta_array(kk+1)))*pi/180;
        Q = [Press(jj,kk) Press(jj+1,kk) dPdzreal(jj,kk)*...  
            scale_t dPdzreal(jj+1,kk)*scale_t;...  
            Press(jj,kk+1) Press(jj+1,kk+1) dPdzreal(jj,kk+1)*...  
            scale_t dPdzreal(jj+1,kk+1)*scale_t;...  
            dPdthetareal(jj,kk)*scale_s dPdthetareal(jj+1,kk)*scale_s 0 ...  
            dPdthetareal(jj,kk+1)*scale_s dPdthetareal(jj+1,kk+1)*scale_s 0 ];
        mm = 1;
        while newthetaarray(mm) < Ltheta_array(kk)
            mm = mm + 1;
        end
        while mm <= length(newthetaarray) && newthetaarray(mm) >=... 
            Ltheta_array(kk) && newthetaarray(mm) < Ltheta_array(kk+1)
            S(mm) = (newthetaarray(mm) - Ltheta_array(kk))/...  
                (Ltheta_array(kk+1) - Ltheta_array(kk));
            nn = 1;
            while newzarray(nn) < Lz_array(jj)
                nn = nn + 1;
            end
            while nn <= length(newzarray) && newzarray(nn) >=... 
                Lz_array(jj) && newzarray(nn) < Lz_array(jj+1)
                T(nn) = (newzarray(nn) - Lz_array(jj))/...  
                        (Lz_array(jj+1) - Lz_array(jj));
                P(nn,mm) = [S(mm)^3 S(mm)^2 S(mm) 1]*M*...  
                    Q*MM'*[T(nn)^3;T(nn)^2;T(nn);1];
                nn = nn + 1;
            end
            mm = mm + 1;
        end
    end
end
P(:,length(P(1,:))) = P(:,1);
%Make matrix to be used to fill in last row
P2 = zeros(length(newzarray), length(newthetaarray));
for jj = 1:numzsegments
    for kk = 1:numthetasegments
        scale_s = (abs(Ltheta_array(kk)) - abs(Ltheta_array(kk+1)))*pi/180;
        Q = [Press(jj,kk) Press(jj+1,kk) dPdzreal(jj,kk)*...  
            scale_t dPdzreal(jj+1,kk)*scale_t;...  
            Press(jj,kk+1) Press(jj+1,kk+1) dPdzreal(jj,kk+1)*...  
            scale_t dPdzreal(jj+1,kk+1)*scale_t;...  
            dPdthetareal(jj,kk)*scale_s dPdthetareal(jj+1,kk)*scale_s 0 ...  
            dPdthetareal(jj,kk+1)*scale_s dPdthetareal(jj+1,kk+1)*scale_s 0 ];
        mm = 1;
while newthetaarray(mm) < Ltheta_array(kk)
    mm = mm + 1;
end
while mm <= length(newthetaarray) && newthetaarray(mm) >= ...
    Ltheta_array(kk) && newthetaarray(mm) <
Ltheta_array(kk+1)
    S(mm) = (newthetaarray(mm) - Ltheta_array(kk))/...
            (Ltheta_array(kk+1) - Ltheta_array(kk));
    nn = 1;
    while newzarray(nn) <= Lz_array(jj)
        nn = nn + 1;
    end
    while nn <= length(newzarray) && newzarray(nn) >=...
        Lz_array(jj) && newzarray(nn) <= Lz_array(jj+1)
        T(nn) = (newzarray(nn) - Lz_array(jj))/...
                (Lz_array(jj+1) - Lz_array(jj));
        P2(nn,mm) = [S(mm)^3 S(mm)^2 S(mm) 1]*M*...
                    Q*M'*[T(nn)^3;T(nn)^2;T(nn);1];
        nn = nn + 1;
    end
    mm = mm + 1;
end
P2(:,length(P2(1,:))) = P2(:,1);
P(128,:) = P2(128,:);

%Save propreadyfield
save propreadyfield P -ascii -double

CYLpressREF.m

This program creates the reference pressure field against which %
conventional NAH (cylNAH.m) and ENAH (cylENAH) reconstructions %
are compared. An output file ,NAHref, is created that is read into %
the NAH and ENAH reconstruction codes.
%NOTE: This file must be located in the directory with the sampled %
pressure and velocity data folders.

clear;

%load finefield
field = load('finefield');
field2 = load('finefield2');
field(128,:) = field2(128,:);

save cylNAHref field -ascii -double
cylNAH.m

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%This program performs conventional NAH on the measured pressure field
%at 4 cm. The cylNAHref text file is loaded to compare against the %
%reconstruction. The normalized whole field error is printed to the %
screen.  
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear;

freq = 1524;            %excitation frequency [Hz]
a = 2*0.0254;           %radius of cylinder [m]
r = 0.04 + a;           %standoff distance [m]
r_est = 0.02 + a;       %estimation distance [m]
rho = 1.21;             %density of air [kg/m^3]
omega = 2*pi*freq;      %excitation angular frequency [Hz]
c = 343;                %speed of sound [m/s]
k = omega/c;            %acoustic wavenumber [m^-1]

for ii = 1:154
    %Read in pressure data
    eval(['addpath ASCII00' num2str(ii) ']);
    [A,B] = textread('G1, lsv00000.txt', '%f %f', -1, 'headerlines',
                     7);
    pressamp(ii) = B(freq+1);
    [f,A,B] = textread('G5, lsv00000.txt', '%f %f %f', -1,
                       'headerlines', 7);
    pressphi(ii) = atan2(B(freq+1),A(freq+1));
end

%Amplitude and phase correction for pressure
fCF1 = 56;      %Hz
Sens = 0.0153;  %V/Pa
S_mic = Sens/(sqrt(1 + (fCF1/freq)^2));     %V/Pa
phase_mic = atan2(56^2,1090);

%Zero phase reference point
pressphiref = pressphi(1)*phase_mic;

dz = 4;
dtheta = 27;
z = -20:dz:20;
theta = 0:dtheta:351;
kk = 1;

%Create pressure and phase
for ii = 1:length(z)
    for jj = 1:length(theta)
        %Pressure
        pamplitude(ii,jj) = pressamp(kk)/S_mic;
        prelphase(ii,jj) = pressphi(kk)*phase_mic - pressphiref;
        kk = kk + 1;
    end
end
pamplitudenew = (pamplitude.*cos(prelphase+incr));
theta = [theta 360];
Press = [pamplitudenew pamplitudenew(:,1)];
Lz_array = z*0.0254;
Ltheta_array = theta;
zarraysize = Lz_array(length(Lz_array))-Lz_array(1);
thetaarraysize = Ltheta_array(length(Ltheta_array))-Ltheta_array(1);
dzarraysize = zarraysize/127;
dthetaarraysize = thetaarraysize/255;
newzarray = Lz_array(1):dzarraysize:Lz_array(length(Lz_array));
newthetaarray = Ltheta_array(1):dthetaarraysize:Ltheta_array(length(Ltheta_array));
P = zeros(length(newzarray),length(newthetaarray));

%Perform linear interpolation of pressure
P = interp2(Ltheta_array,Lz_array,Press,newthetaarray,newzarray,'linear');

%%%%%%%%%%%%%%%%%%%%%
%NAH RECONSTRUCTION %
%%%%%%%%%%%%%%%%%%%%%

%Apply Tukey window
x_w = 0.10; %width of the spatial window taper [m]

%Apply to top edge
Lz_end = newzarray(length(newzarray));
Lz_start = newzarray(1);
ii = 1;
while newzarray(ii) < Lz_end - x_w
    ii = ii + 1;
end
for ii = ii:length(newzarray)
    for jj = 1:length(newthetaarray)
        P(ii,jj) = P(ii,jj) * ... 
                    (0.5 - 0.5*cos(pi*(newzarray(ii) - Lz_end)/x_w));
    end
end

%Apply to bottom edge
ii = 1;
while newzarray(ii) < Lz_start + x_w
    for jj = 1:length(newthetaarray)
        P(ii,jj) = P(ii,jj) * ... 
                    (0.5 - 0.5*cos(pi*(newzarray(ii) - Lz_start)/x_w));
    end
    ii = ii + 1;
end

%%%%%%%%%%%%%%%%%%%%%
% 2-D SPATIAL FFT %
%%%%%%%%%%%%%%%%%%%%%

%Pad with zeros in z only to 256 by 256
newP = zeros(256,256);
for ii = 64:191
    newP(ii,:) = P(ii-63,:);
end

%Take 2D FFT
Pkspace = (4/(256*256)) * fft2(newP);

%Create kz and m (hankel function order) arrays
deltakz = (2*pi)/(4*Lz_end);
incr = [0:1:128,zeros(1,127)];
kz = deltakz .^ incr;

%Order of hankel functions
m = [0:1:128,zeros(1,127)];
xx = 2;
for ii = 130:256
    kz(1,ii) = -kz(1,ii-xx);
    m(1,ii) = -m(1,ii-xx);
    xx = xx + 2;
end

%Generate hankel functions
for ii = 1:length(incr)
    for jj = 1:length(incr)
        if kz(ii)*kz(ii) <= k*k
            kr(ii,jj) = sqrt(k*k - kz(ii)*kz(ii));
            num = besselj(m(jj),kr(ii,jj)*r_est) +
            i*bessely(m(jj),kr(ii,jj)*r_est);
            den = besselj(m(jj),kr(ii,jj)*r) +
            i*bessely(m(jj),kr(ii,jj)*r);
            Hratio(ii,jj) = num/den;
        else
            kr(ii,jj) = sqrt(kz(ii)*kz(ii) - k*k);
            num = besselk(m(jj),kr(ii,jj)*r_est);
            den = besselk(m(jj),kr(ii,jj)*r);
            Hratio(ii,jj) = num/den;
        end
        check = isnan(Hratio(ii,jj));
        if check == 0
            Hratio(ii,jj) = 0;
        end
    end
end

%Apply k-Space window
K_w = zeros(256,256);
k_c = 20;               %cutoff wavenumber
alpha = 0.1;            %control of filter decay rate

%Create windowing function
for ii = 1:256
    for jj = 1:256
        if sqrt(kz(ii)*kz(ii) + m(jj)*m(jj)) < k_c
            K_w(ii,jj) = 1 - 0.5*exp(-(1 - (sqrt(kz(ii)*kz(ii) +
            m(jj)*m(jj))/k_c))/alpha);
        else
            K_w(ii,jj) = 0;
        end
    end
end
\[ K_w(ii,jj) = 0.5 \exp\left(1 - \left(\sqrt{kz(ii) * kz(ii) + m(jj) * m(jj)} / k_c\right) / \alpha\right); \]

\[ \text{end} \]
\[ \text{end} \]
\[ \text{end} \]

\[ \text{Pknewz} = \text{zeros}(256,256); \]
\[ \% \text{Multiply by inverse Green's function} \]
\[ \text{for ii} = 1:\text{length(incr)} \]
\[ \quad \text{for jj} = 1:\text{length(incr)} \]
\[ \quad \quad \text{Pknewz(ii,jj)} = \text{Pkspace(ii,jj)} * \text{Hratio(ii,jj)}; \]
\[ \quad \text{end} \]
\[ \text{end} \]
\[ \% \text{Multiply by k-Space filter} \]
\[ \text{Pknewz} = \text{Pknewz} .* K_w'; \]
\[ \% \text{Inverse spatial 2D FFT} \]
\[ \text{p_estimation} = ((256 * 256) / 4) * \text{ifft2(Pknewz)}; \]
\[ \text{newzsize} = -2 * 20 * 0.0254:2 * 40 * 0.0254; \]
\[ \text{realp_est} = \text{real(p_estimation)}; \]
\[ \% \text{Adjust size of realp_est back to original measurement array size} \]
\[ \text{for ii} = 64:191 \]
\[ \quad \text{newP2(ii-63,:) = realp_est(ii,:);} \]
\[ \text{end} \]
\[ \% \text{Load reference pressure field} \]
\[ \text{ref} = \text{load('cylNAHref')}; \]
\[ \text{residuals} = \text{ref} - \text{newP2}; \]
\[ \% \text{Print whole field error to the screen} \]
\[ \text{error} = \text{std(std(residuals))} / \text{max(max(abs(ref)))} \]

\text{cylENAH.m}
zarraysize = 1.016;
thetaarraysize = 360;
dzarraysize = zarraysize/127;
dthetaarraysize = thetaarraysize/255;
newzarray = -0.508:dzarraysize:0.508;
newthetaarray = 0:dthetaarraysize:360;

%Load propreadyfield
P = load('propreadyfield');

%%%%%%%%%%%%%%%%%%%%%
% NAH RECONSTRUCTION %
%%%%%%%%%%%%%%%%%%%%%

%Apply Tukey window
x_w = 0.10;         % width of the spatial window taper [m]

%Apply to top edge
Lz_end = newzarray(length(newzarray));
Lz_start = newzarray(1);
ii = 1;
while newzarray(ii) <  Lz_end - x_w
    ii = ii + 1;
end
for ii = ii:length(newzarray)
    for jj = 1:length(newthetaarray)
        P(ii,jj) = P(ii,jj) * ... 
                    (0.5 - 0.5*cos(pi*(newzarray(ii) - Lz_end)/x_w));
    end
end

%Apply to bottom edge
ii = 1;
while newzarray(ii) <  Lz_start + x_w
    for jj = 1:length(newthetaarray)
        P(ii,jj) = P(ii,jj) * ... 
                    (0.5 - 0.5*cos(pi*(newzarray(ii) - Lz_start)/x_w));
    end
    ii = ii + 1;
end

%%%%%%%%%%%%%%%%%%%%%
% 2-D SPATIAL FFT %
%%%%%%%%%%%%%%%%%%%%%

%Pad with zeros in z only to 256 by 256
newP = zeros(256,256);
for ii = 64:191
    newP(ii,:) = P(ii-63,:);
end

%Take 2D FFT
Pkspace = (4/(256*256)) * fft2(newP);

%Create kz and m (hankel function order) arrays
deltakz = (2*pi)/(4*Lz_end);
incr = [0:1:128,zeros(1,127)];
kz = deltakz .* incr;

%Order of hankel functions
m = [0:1:128, zeros(1,127)];
xx = 2;
for ii = 130:256
  kz(1,ii) = -kz(1,ii-xx);
  m(1,ii) = -m(1,ii-xx);
  xx = xx + 2;
end

%Generate hankel functions
for ii = 1:length(incr)
  for jj = 1:length(incr)
    if kz(ii)*kz(ii) <= k*k
      kr(ii,jj) = sqrt(k*k - kz(ii)*kz(ii));
      num = besselj(m(jj),kr(ii,jj)*r_est) +
        i*bessely(m(jj),kr(ii,jj)*r_est);
      den = besselj(m(jj),kr(ii,jj)*r) +
        i*bessely(m(jj),kr(ii,jj)*r);
      Hratio(ii,jj) = num/den;
    else
      kr(ii,jj) = sqrt(kz(ii)*kz(ii) - k*k);
      num = besselk(m(jj),kr(ii,jj)*r_est);
      den = besselk(m(jj),kr(ii,jj)*r);
      Hratio(ii,jj) = num/den;
    end
    check = isfinite(Hratio(ii,jj));
    if check == 0
      Hratio(ii,jj) = 0;
    end
  end
end

%Apply k-Space window
K_w = zeros(256,256);
k_c = 20;               %cutoff wavenumber
alpha = 0.1;            %control of filter decay rate
for ii = 1:256
  for jj = 1:256
    if sqrt(kz(ii)*kz(ii) + m(jj)*m(jj)) < k_c
      K_w(ii,jj) = 1 - 0.5*exp(-1 - (sqrt(kz(ii)*kz(ii) +
        m(jj)*m(jj))/k_c))/alpha;
    else
      K_w(ii,jj) = 0.5*exp((1-(sqrt(kz(ii)*kz(ii) +
        m(jj)*m(jj))/k_c))/alpha);
    end
  end
end

Pknewz = zeros(256,256);
%Multiply by inverse Green's function
for ii = 1:length(incr)
  for jj = 1:length(incr)
    Pknewz(ii,jj) = Pkspace(ii,jj)*Hratio(ii,jj);
  end
end
%Multiply by k-Space filter
Pknewz = Pknewz.*K_w';

%Inverse spatial 2D FFT
p_estimation = ((256*256)/4).*ifft2(Pknewz);
newzsize = -2*20*0.0254:(2*40*0.0254)/255:2*20*0.0254;
realp_est = real(p_estimation);

%Adjust size of realp_est back to original measurement array size
for ii = 64:191
    newP2(ii-63,:) = realp_est(ii,:);
end

%Load reference pressure field
ref = load('cylNAHref');
residuals = ref - newP2;

%Print whole field error to the screen
error = std(std(residuals))/(max(max(abs(ref))));
REFERENCES


