Real Time Bottom Reverberation Simulation in Deep and Shallow Ocean Environments

by

Thomas Edward Miller

Submitted to the Joint Program in Applied Ocean Science & Engineering
in partial fulfillment of the requirements for the degree of

Master of Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

and the

WOODS HOLE OCEANOGRAPHIC INSTITUTION

September 2015

Author . . . .
Joint Program in Applied Ocean Science & Engineering
Massachusetts Institute of Technology
& Woods Hole Oceanographic Institution
August 17, 2015

Certified by . . . .
Henrik Schmidt
Professor of Mechanical and Ocean Engineering
Massachusetts Institute of Technology
Thesis Supervisor

Accepted by . . . .
David E. Hardt
Chairman, Committee for Graduate Students
Massachusetts Institute of Technology

Accepted by . . . .
Henrik Schmidt
Chairman, Joint Committee for Applied Ocean Science & Engineering
Massachusetts Institute of Technology
Woods Hole Oceanographic Institution

Real Time Bottom Reverberation Simulation in Deep and Shallow Ocean Environments

by

Thomas Edward Miller

Submitted to the Joint Program in Applied Ocean Science & Engineering
Massachusetts Institute of Technology
& Woods Hole Oceanographic Institution
on August 17, 2015, in partial fulfillment of the
requirements for the degree of
Master of Science

Abstract

Due to the costs involved and time required to perform experiments at sea, it is important to provide accurate simulations of the ocean environment. Using the ray tracing code, BELLHOP, the Mission Oriented Operating Suite (MOOS), methods outlined by the Naval Research Laboratory (NRL) for bottom reverberation, and MATLAB, a model will be developed to incorporate the effects of bottom reverberation into the BELLHOP suite of code. This will be accomplished by using BELLHOP to generate a ray trace and eigen ray file. Then a MATLAB script will take the BELLHOP information and calculate the reverberation level using the NRL model by measuring the amplitude and reverberation at a receiver array simulated on the ocean floor. These reverberation values will then be used to determine the reverberation level at the source due to these bottom interactions. Testing of the simulation will include deep and shallow ocean profiles and multiple sound speed profiles (SSP). Following this testing, the goal is to implement the model in existing C++ code used for the testing of AUV systems. The ability to accurately model the ocean will not only allow for testing of autonomy code in the laboratory, but also make it possible to refine and calibrate code making ship time more efficient.

Thesis Supervisor: Henrik Schmidt
Title: Professor of Mechanical and Ocean Engineering
Massachusetts Institute of Technology

Acknowledgments

I would like to thank Dr. Henrik Schmidt for his guidance and patience. Thanks are also due to the United States Navy for providing funding for my participation in the MIT/WHOI Joint Program and for allowing me the time away from the ship to complete the requirements for a Master's degree at MIT. This is an opportunity for which I will always be grateful. I would also like to thank Sheida Danesh, and Erin Fischell for writing much of the baseline code utilized in this work, as well as the other members of LAMSS who have done much of the work that has gone before and set me up for success: Toby Schneider, Stephanie Petillo, Alon Yaari, Sheida Danesh, Thom Howe, and Nick Rypkema. To all of the great friends I've made at MIT, thank you for your love and support through times of stress and difficulty. Thank you to Beth for being a great mother to our two children. Lastly and most importantly, thank you Lucy and Violet, my two wonderful daughters, for being you.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

But more wonderful than the lore of old men and the lore of books is the secret lore of ocean.
H.P. Lovecraft

1.1 Motivations for Bottom Reverberation Simulation

Sea trials cost immense amounts of time, manpower, and money. Because of this, it is important to construct simulations that adequately represent the ocean environment in order to ensure code modules are ready, in advance. The Laboratory for Autonomous Marine Sensing Systems (LAMSS) at MIT develops modules for the Mission Oriented Operating Suite (MOOS) middleware for use on board autonomous underwater vehicles (AUV). If these modules are to be used during actual ocean experimentation, it is imperative that the simulator be able to adequately represent the effects of bottom interaction and an active sonar system. The existence of a robust ocean simulation allows for the calibration and refinement of sonar systems and autonomy programming before going to sea. This will save ship time that would otherwise be wasted troubleshooting and testing. The costs saved are invaluable to the current funding environment.

1.2 Objectives

- Provide background for reverberation modeling.
- Select and test bottom reverberation models.
- Choose a single model to implement.
- Generate a MATLAB script implementing the chosen model.
- Verify script against model.
- Integrate the MATLAB script with BELLHOP suite.
- Apply model to multiple ocean environments with varying SSPs to measure the effects of reverberation from the bottom.

1.3 Organization

Chapter 2 will provide information on the use of MOOS, AUV, and BELLHOP programs. The focus of Chapter 3 will be a discussion on various methods used in sound propagation modeling. Chapter 4 will walk through the different types of acoustic reverberation and the Naval Research Laboratory (NRL) model chosen for the simulation. Results and conclusions will be discussed in Chapter 5.

Chapter 2

AUVs, MOOS, and BELLHOP

2.1 Autonomous Underwater Vehicles

In the past, large, expensive, manned vessels have been used to explore and observe the ocean environment. As technology advances and budgets become tighter, unmanned robotic vessels have replaced these larger platforms. These robotic platforms include Autonomous Surface Vessels (ASVs), Remotely Operated Vehicles (ROVs), and Autonomous Underwater Vehicles (AUVs). AUVs have seen an increase in use and are replacing older ROV systems. These AUVs can be used to perform underwater surveys, area surveillance, or mine countermeasures and undersea warfare missions. As a tool for underwater surveys, an AUV can replace divers and expensive survey ship operations, saving both money and lives. Using side scan and ordinary sonar systems, an AUV can proceed to depths that are much more difficult to reach by humans and produce better results. As an anti-submarine and mine counter measures tool, an AUV can successfully search, detect, track, and classify potential targets. This reduces risk to both submarines and submariners [10].

The members of LAMSS at MIT spend much of their time developing autonomy code and algorithms to enhance the abilities of current AUV systems. Popular middleware programs used for autonomous operations include: Lightweight Communications and Marshalling (LCS), developed by the 2006 MIT DARPA Urban Challenge team; Robotic Operating System (ROS), an open source collection of software whose roots can be found at Stanford University; and Mission Oriented Operating Suite (MOOS), developed jointly by Cambridge University and MIT.

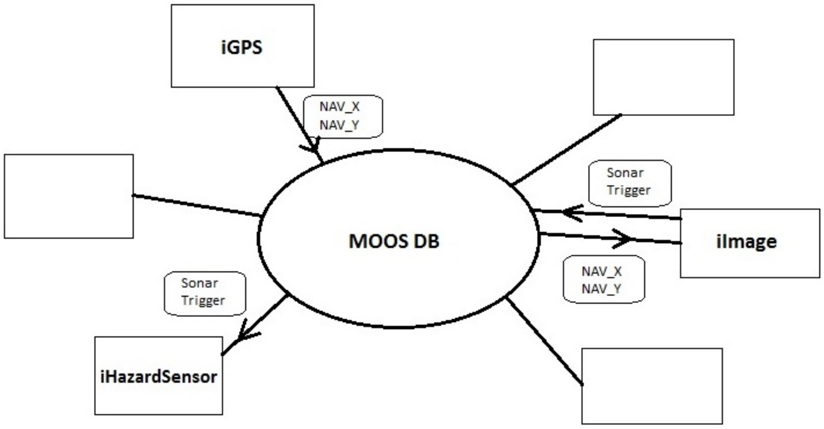

Figure 2-1: MOOS Publish/Subscribe Architecture

2.2 MOOS

LAMSS at MIT uses MOOS as its middleware system on its AUVs. This program is used to control all AUV functions from navigation to onboard processing of sonar signals and processes. MOOS was developed by Paul Newman while a Post-Doc at MIT in 2001-2002. The premise of this MOOS design was to develop a system that could be shared across multiple platforms and able to share information across independently developed processes. To accomplish this, MOOS was setup with a star pattern or publish/subscribe architecture. A central MOOS database, MOOSDB, connects to each application. The applications have no connection to each other, but instead publish information to the MOOSDB. Another application can then subscribe to that information and request it from the MOOSDB. In this way, applications can be developed independently and have no dependence on each other [6]. This substantially simplifies the programming process and makes it possible to use the MOOS middleware across platforms with little trouble.

Figure 2-1 illustrates the process. In the example, three applications interact through the MOOSDB. iGPS is an application that interacts with a GPS unit to determine the AUVs physical location. iImage is a theoretical application that produces a sonar image when the vehicle reaches a specific location and iHazardSensor is a theoretical application that begins analyzing sonar data for the presence of a hazardous object when sonar data is available. iGPS produces the MOOS variables NAV_X and NAV_Y and publishes them to the MOOSDB at a predetermined rate. iImage subscribes to NAV_X and NAV_Y, so every time a new set is published by iGPS, the values are obtained from the MOOSDB for use by iImage. When the AUV reaches the point at which iImage is set to operate, iImage sends a signal to fire the sonar and informs the MOOSDB that a sonar image is available for processing with the variable SONAR_TRIGGER. iHazardSensor is subscribed to SONAR_TRIGGER and sees the change in state, at which point, it performs its task. The beauty of this process is that the applications can be changed and as long as variable names remain the same, no other processes or applications are affected. For instance, if the iGPS application were replaced with iCompass, as long as the output of iCompass is named NAV_X and NAV_Y, iImage is able to perform its job.

2.3 BELLHOP

BELLHOP is a model for predicting acoustic pressure fields in ocean environments using ray tracing developed by Michael Porter. BELLHOP takes an environmental file, describing the sound speed profile (SSP), and produces a variety of outputs. The possible outputs can include eigenrays, transmission loss, time-series, etc. The BELLHOP code can be implemented across multiple operating systems and programming languages.

For the purposes of this work, BELLHOP will be used in conjunction with Matlab. The Matlab code will be used to create and environmental file (.env). Within the Matlab script, BELLHOP will be called to process the .env file. BELLHOP will output a .ray file. This file will describe the motion of each ray as it travels through the water column based on dynamics of the ocean environment. BELLHOP will determine if any eigenrays, rays that pass through the receiver, are present. For each eigenray, an arrival file, .arr, will be produced that describes the amplitude, travel time, number of arrivals, number of top and bottom interactions, and take off and receiver angles [8]. The specifics of the code implementation will be discussed in Chapter 4.

Chapter 3

Propagation Techniques

BELLHOP uses ray tracing in order to model underwater sound propagation. The following sections discuss the history of ray theory and other propagation methods. The advantages and disadvantages of each will be discussed.

3.1 Ray Tracing

Ray theory began in optics to explain the propagation of light. Study of ray theory can be traced as far back as Euclid, and as such, ray theory was understood long before it was mathematically formalized by Snell's Law and Maxwell's Equation, 1626 and 1861 respectively. Snell's Law was first used to describe the interactions of light when it passes through two media with different indexes of refraction. Sound waves passing through the ocean behave in much the same and Snell's Law can be used to describe the motion of sound waves, as well.

3.1.1 Rays and Snell's Law

Equation 3.1, the mathematical representation of Snell's Law, shows the relationship between the sound speed in a medium, c, and the cosine of the grazing angle, θ, to be constant.

$$\frac{c_1}{\cos\theta_1} = \frac{c_2}{\cos\theta_2} = const. \tag{3.1}$$

This equation is represented in figure 3-1. Defining the angular frequency, ω, in equation 3.2 and the wavenumber, k, in equation 3.3, we can modify equation 3.1 to define the

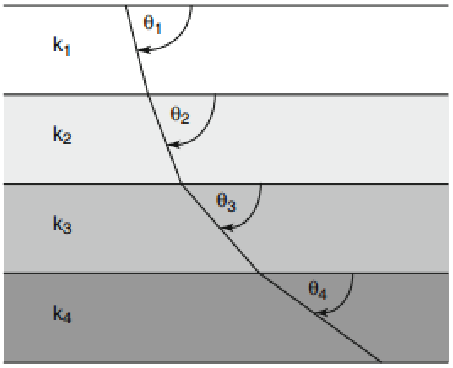

Figure 3-1: Snell's Law

horizontal phase speed, V_x, of the rays in equation 3.4. In figure 3-1, you can intuitively see that the horizontal phase speed of each layer must also be constant. If it were not, the boundary between layers would be in motion.

$$\omega = 2\pi f \tag{3.2}$$

$$k = \frac{\omega}{c} \tag{3.3}$$

$$\frac{omega}{k_1 \cos\theta_1} = \frac{omega}{k_2 \cos\theta_2} = V_x \tag{3.4}$$

In 3-1, you can also see that if each layer has a higher sound speed that the rays begin to bend upward. For linearly increasing sound speed, rays bend in circular arcs. This can be seen in equations 3.5, 3.6 and 3.7. Where *r(s)* is the range, R is the radius of the arc, $theta_0$ is the launch angle, *theta* is the incident angle and the ray's end, *z(s)* is the depth, *c(0)* is the surface sound speed, g is the sound speed gradient in a linear SSP, and *c(*z_0*)* is the sound speed at the launch depth. In a non-linear profile, sound is "lazy" and bends toward the lower sound speed [4].

$$r(s) = R\sin\theta_0 - R\sin\theta \tag{3.5}$$

$$z(s) = R\cos\theta - \frac{c(0)}{g} \tag{3.6}$$

$$R = \frac{c(z_0)}{g \cos \theta_0} \tag{3.7}$$

As a ray travels unobstructed, it will travel in the way described above. However, sound traveling through the ocean undergoes many interactions. The major interactions involve sound traveling into an ocean boundary; i.e. the surface, the bottom, or a volume boundary such as a school of fish or other inclusion in the water. Understandably the type of ocean environment can have a large impact on the types of interactions the sound undergoes. In a shallow water environment, constant meetings of the ray with the surface and the bottom will cause large transmission losses and the ray will subsequently not travel far. In a deep environment, the ray is likely to undergo very few interactions with the ocean surface or bottom and will travel much farther. The placement of the source will also affect the path which the ray travels. For instance, due to the fact that sound refracts toward the lower sound speed, if the source were put at a minimum in the SSP, the ray would travel over a large distance with little interaction with the environment. The frequency of the sound affects how far it will travel, as well. Higher frequencies undergo large amounts of attenuation and travel a short distance compared to lower frequencies [1].

3.1.2 Derivation of Ray Tracing Equations

In the previous section, Snell's Law was briefly discussed. It forms the basis for ray tracing equations, but does not tell the whole story. Now, all of the ray tracing calculations will be done within BELLHOP. Included is a derivation of the basic equations for understanding and the method followed in reference [4] will be used.

Starting with the Helmholtz equation, equation 3.8, *c(x)* is the sound speed and *omega* is the angular frequency of the source at x_0. In order to solve this equation, two separate equations must be solved. One is the Transport equation, which solves for the amplitude across the ray bundle, and the other is the Eikonal equation, which solves for the actual ray path. These equations come from taking the second derivative of equation 3.9, a series solution for the Helmholtz equation known as the ray series.

$$\nabla^2 p + \frac{\omega^2}{c^2(\mathbf{x})} p = -\delta(\mathbf{x} - \mathbf{x}_0) \tag{3.8}$$

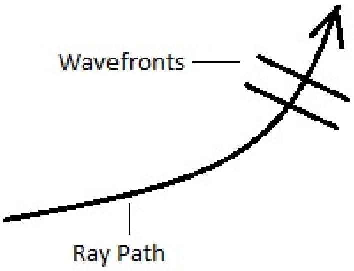

Figure 3-2: Wavefronts perpendicular to a ray path

$$p(\mathbf{x}) = e^{i\omega\tau(x)} \sum_{j=0}^{\infty} \frac{A_j(\mathbf{x})}{(i\omega)^j} \tag{3.9}$$

After taking the second derivative of equation 3.9, the result is inserted into equation 3.8. Then a first order approximation is made to simplify the result. The Eikonal equation is pulled from part of that first order approximation of the second derivative and can be seen in equation 3.10.

$$(\nabla\tau)^2 = \frac{1}{c^2(\mathbf{x})} \tag{3.10}$$

The Eikonal equation represents the equation of a set of wavefronts perpendicular to the ray path as seen in figure 3-2. In order to solve this equation, the family of rays perpendicular to the wavefronts are used to define a new coordinate system, thereby simplifying the equation. The new coordinate system is constructed such that $\mathbf{x}_s$ is defined as the ray trajectory and is used to yield equation 3.11.

$$\frac{d\mathbf{x}}{ds} = c\nabla\tau \tag{3.11}$$

$$\frac{d}{ds}(\frac{1}{c}\frac{dx}{ds}) = -\frac{1}{c^2}\nabla c \tag{3.12}$$

The x-component of 3.11 is differentiated with respect to s and the Eikonal equation, 3.10, is substituted into the result. This process is repeated for the y- and z-component and yields equation 3.12. Equation 3.12 is simplified by assuming everything to be axis-symmetric and rewriting the equation in cylindrical coordinates where sound speed, c, is assumed to be a function of depth only. The result is the general ray tracing equations, equations 3.13 to

3.16.

$$\frac{dr}{ds} = c\xi(s) \tag{3.13}$$

$$\frac{dz}{ds} = c\zeta(s) \tag{3.14}$$

$$\frac{d\xi}{ds} = -\frac{1}{c^2}\frac{\partial c}{\partial r} \tag{3.15}$$

$$\frac{d\zeta}{ds} = -\frac{1}{c^2}\frac{\partial c}{\partial z} \tag{3.16}$$

Now that the Eikonal equation is solved, in general terms, it is time to solve the Transport equation, equation 3.17. This can be reduced to the form seen in 3.18 at which time the divergence theorem is applied to obtain the result of equation 3.19.

$$2\nabla\tau \bullet \nabla A_0 + (\nabla^2\tau)A_0 = 0 \tag{3.17}$$

$$\nabla \bullet (A_0^2 \nabla\tau) = 0 \tag{3.18}$$

$$\int_{dV} A_0^2 \nabla\tau \mathbf{n} dS \tag{3.19}$$

Reference [4] defines a ray tube as the volume enclosed by a family of rays. This means that $\mathbf{n}$ can be defined as pointing in the same direction of the ray. Therefore, $\mathbf{n} = \frac{dx}{ds}$ and $\nabla\tau \bullet \mathbf{n} = \frac{1}{c}$. Leaving us with the Transport equation of 3.20.

$$\int_{dV} A_0^2 \frac{1}{c} dS = constant \tag{3.20}$$

Because the value of the integral stays constant as the volume of the ray bundle changes, the Transport equation can be reduced to the final form of 3.21 and 3.22. The extra r is due to the assumption of cylindrical symmetry.

$$A_0(s) = A_0(0)\sqrt{\frac{c(s)}{c(0)}\frac{J(0)}{J(s)}} \tag{3.21}$$

$$J = r\sqrt{\left(\frac{\partial z}{\partial \theta_0}\right)^2 + \left(\frac{\partial r}{\partial \theta_0}\right)^2} \tag{3.22}$$

3.1.3 Ray Tracing Advantages and Disadvantages

The main advantage of ray tracing is the intuitiveness of the way it represents sound rays as they travel through the ocean. In a linear sound speed profile, it is easy to see that sound travels in circular arcs. In a non-linear profile, the general motion of the rays can still be intuitively traced because the physics dictate that sound will refract in the direction of the lower sound speed. Ray tracing is also computationally efficient. Only the last step needs to be retained in order to calculate the next step. This is especially true over long ranges.

The use of approximations, most notably the high-frequency approximation that produced equation 3.10 by taking a first order approximation of the summation of the second derivative of 3.9, is the major disadvantage of ray tracing. This makes ray tracing not as accurate at lower frequencies. Also, because the intensity of the ray is inversely proportional to the cross-sectional area of the ray tube, if the ray bends and the cross-sectional area goes to zero, the intensity will spike to infinity. This is obviously an artifact of the ray tracing equations and not based in reality. This leaves an unrealistic solution to the wave equation at the caustic and in an area surrounding the caustic and introduces a phase change to the ray that can cause errors further along the ray [4]. In ray tracing, shadow zones appear as stark contrast to the rays. In reality the differences between the shadow zone and the rays would be more continuous as opposed to areas of zero sound up against the ray zones [7].

In reference [7], a method to smooth the transition at shadow zones and caustics was developed. The idea is to calculate an intensity distribution for the rays using a Gaussian bell curve. This provides a solution closer to the exact solution of the wave equation. This method of using Gaussian beams still requires the use of a high-frequency, first order approximation and adds the challenge of selecting the correct Gaussian profile with the proper beam width [4].

3.2 Additional Solution Methods

3.2.1 Wavenumber Integration

Wavenumber integration can be used in horizontally stratified, range independent media to solve the wave equation. That is the media can only change in depth and not in range. This works by dividing a half space into a number of smaller layers and then solving the equations within each slice by matching conditions at the boundaries.

A Hankel transform is used to solve the Helmholtz equation in order to produce a Green's function as a solution to the Helmholtz equation. The two dimensional, Cartesian coordinate Fourier transform solution can be turned into a one dimensional, cylindrical coordinate Hankel transform and by defining the Hankel transform as Frisk did in reference [3], we can produce equation 3.23. This can then be applied to the inhomogeneous wave equation to produce equation 3.24.

$$g(r) = \int_0^\infty g(k_r) J_0(k_r r) k_r dk_r \tag{3.23}$$

$$I.H.T \left(\frac{1}{r} \frac{\delta}{\delta r} \left[r \frac{\delta p(r; z, z_0)}{\delta r} \right] \right) = k_r^2 g(r; z, z_0) \tag{3.24}$$

Wavenumber integration solves each layer separately. For a complicated sound speed profile, this will require many layers. The more layers that are introduced, the longer the solution will take to obtain. For higher frequencies, higher resolution models will be required which will also take more time [4].

3.2.2 Normal Mode Solution

Normal modes solves the Helmholtz equation through an eigen function expansion. This solution takes the form of a Green's function, and produces a set of modes similar to those of a vibrating string. The frequencies of the vibration are used to develop the horizontal wavenumbers of the modal propagation which are then weighted based on the source depth and summed to produce the complete acoustic field [4]. A representation of the first three modes of a string can be seen in figure 3-3.

Once again assuming cylindrical symmetry and range independence you can put the Green's function into the form of the Sturm-Liouville equation [3], which has a known solution. The Sturm-Liouville equation satisfies the velocity boundary equations but does not satisfy the impedance conditions, because energy is trapped within the half space and no attenuation into the ocean bottom is taken into account. Equation 3.25 shows the general normal mode equation.

$$p(r, z) = \sum a_n(z_0) u_n(z) R_n(r) \tag{3.25}$$

The amplitude of each mode is represented by $a_n(z_0)$, $u_n(z)$ is the vertical mode function, and $R_n(r)$ is the radial function. This equation can then be used to describe a simple situation where a half space has a pressure release surface at the top, a perfectly reflecting

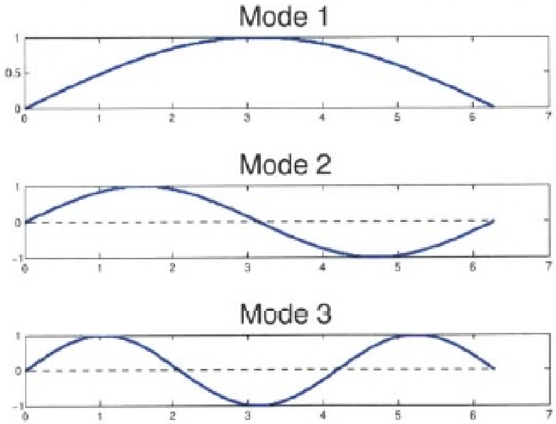

Figure 3-3: First three modes of a vibrating string

bottom with a fixed sound speed and density, and a constant SSP within the waveguide. This would be represented by equation 3.26.

$$p(r,z) = \frac{2i\pi}{h}\sum \sin(k_{zn})z_0 \sin(k_{zn}z)H_0^1(k_n,r) \tag{3.26}$$

Where r is range and z is depth. You can easily see the amplitude, vertical mode function, and radial function represented as in 3.25. The radial function is represented by the Hilbert transform, $H_0^1(k_n,r)$, and can be replaced with $\frac{e^{ik_nr}}{\sqrt{k_nr}}$ by making a farfield approximation. It can be seen that even to solve the simple case requires the entire field to be solved to see the direction of sound propagation within the waveguide [3]. Because we assumed a constant sound speed, any varying sound speed would require some form of finite difference method to find the solution at each sound speed layer. This can quickly become computationally unwieldy. The results will appear to look very much the same as rays especially at higher frequencies where ray tracing excels. This can be seen in figure 3-4. As frequency rises, it can be seen that the normal mode method produces similar results to ray tracing.

Due to the high frequency nature of the sources that will be used in the simulation, offsetting inaccuracies from the high frequency approximation, and the computational complexity of wavenumber integration and normal modes, ray tracing will be used for the simulation. BELLHOP, as discussed in chapter 2, will be used to calculate and produce the ray patterns.

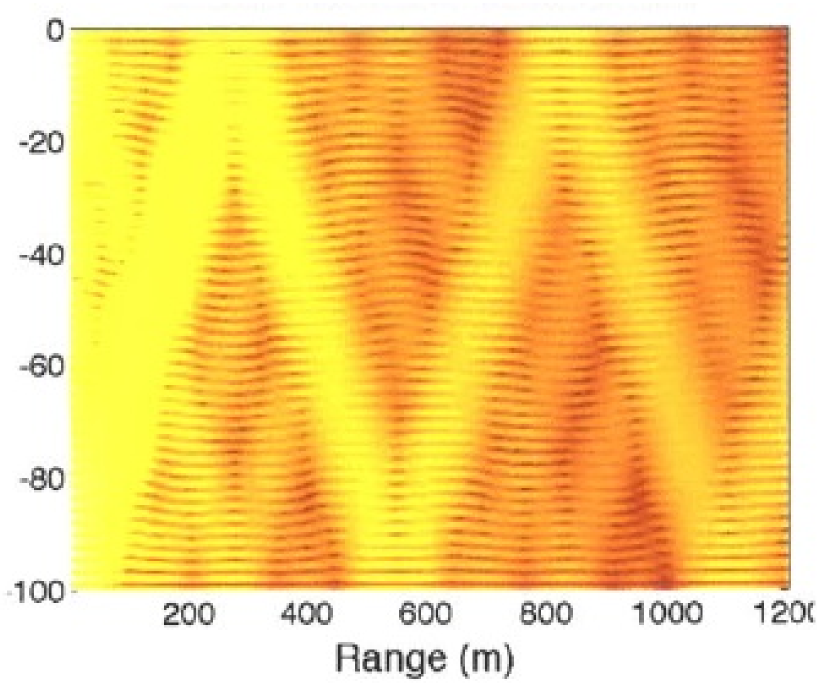

Figure 3-4: Normal mode waveguide at 1000 Hz

Chapter 4

Reverberation

The depth of the ocean varies significantly from a few meters to several thousand meters. An AUV operates anywhere within that space, surface to bottom. In order to correctly track a target, an accurate idea of how sound travels within and interacts with the ocean must be developed. In simulation, the AUV is assumed to be equipped with a mid-frequency (1-5kHz) sonar source and a hydrophone array. This source and the hydrophone array will be simulated by BELLHOP, as will the target.

The goal is to locate the target within the water column by producing a sonar pulse and then waiting for the return at the hydrophones. This will require the acoustic ping to travel from the source, through the water column in a ray pattern, interact with the target, and then travel back through the ocean to arrive at the hydrophone array. During this time, based on the ray pattern, the rays may interact with an ocean surface, whether the top or the bottom. This interaction causes the signal to change and produces a different result

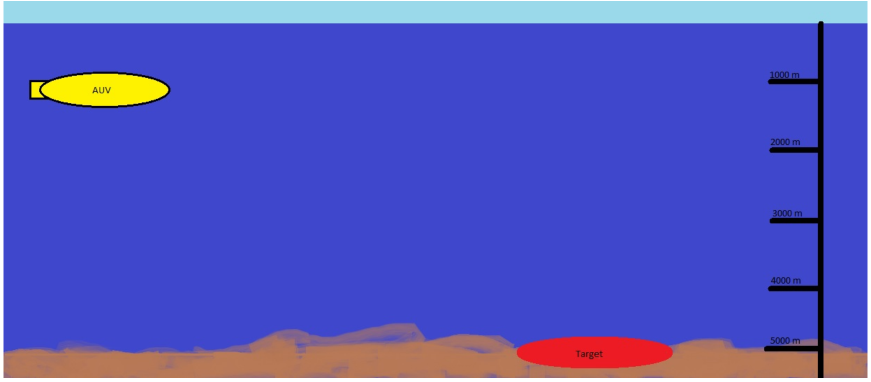

Figure 4-1: Sample Simulation Environment

than one would expect to see at the hydrophone array otherwise. Because of the myriad interactions the sound pulse will undergo in the water, it is necessary to take into account the distortion to the signal due to target strength, environmental noise, and ambient noise.

The majority of environmental noise, when using an active sonar system, is due to interactions with the ocean surfaces and subsequent reverberation. Reverberation comes in three main types: reverberation at the sea surface, reverberation at the bottom surface, and volume reverberation.

Volume reverberation is due to marine life and matter distributed throughout the ocean, as well as, the inhomogeneous structure of the ocean. Surface reverberation is produced by scattering that occurs near the ocean surface and is caused by air bubbles and roughness at the 2D surface. This scattering is dependent on grazing angle, frequency, and roughness. Roughness is affected by wave height and wind at the surface [3]. Bottom reverberation will be discussed in the next section.

4.1 Bottom Reverberation

The reverberation at the ocean bottom is mostly a result of the roughness and makeup of the bottom. The bottom works as an effective scatterer and reflector of sound. Figure 4-2 illustrates the ways in which sound interacts at the ocean bottom. The amount of sound scattered and the direction in which it scatters will be dependent on grazing angle, frequency, bottom density, bottom sound speed, density of water, and sound speed in the water at the interface.

There are multiple ways to determine bottom reverberation. Many different models exist but they all start from the same basis. Equation 4.1 is the basic scattering equation and is a Lambert's law relationship that provides a relationship between grazing angle and backscatter strength [11].

$$S_s = 10 \log \mu + 10 \log \sin^2 \theta \tag{4.1}$$

Where S_s is defined as the scattering strength, $10 \log \mu$ is a proportionality constant based on the amount of energy lost from the interaction that will change based on bottom type, and θ is the grazing angle. This law was developed for light but works well with acoustic scattering, as well, and provides a good base line on which to base more complex models. Through testing, it has been determined that $10 \log \mu$ can be approximated to between -25

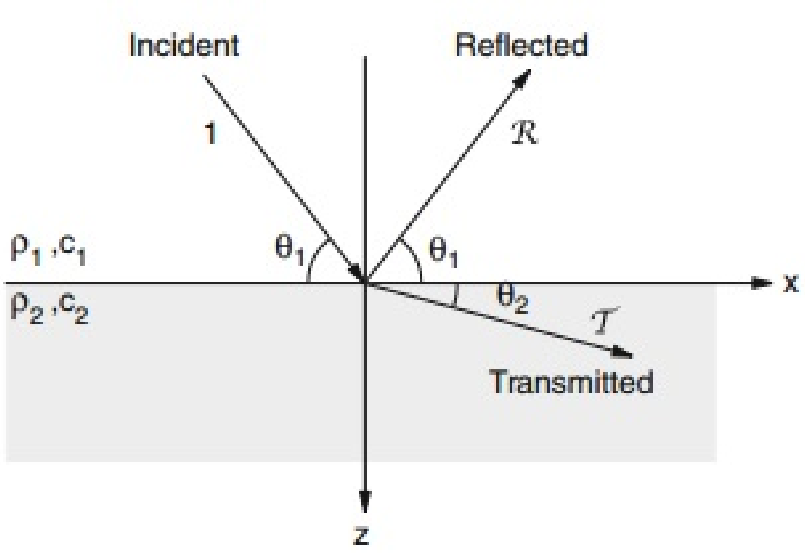

Figure 4-2: Reflection and Transmission of Sound at a Surface Interaction

and -35dB for most unconsolidated sediments ranging from silt to coarse sand. An average of -29dB was used as the baseline value for verifying subsequent models [4]. While the Lambert's law relation provides a generalized value for bottom scattering strength, several physics based models have since been developed.

4.2 Fluid-Solid Interface

The first model is based on the physics of a fluid half-space over a solid half-space. I will be discussing the methods from reference [4]. Bottom sediments can be easily modeled as fluids but this means they only support compressional sound waves. This is often a good approximation due to the structure of most sediments. However, when there is no sediment or the bottom is made of rock, the medium will support both compressional and shear waves and must be modeled as an elastic medium. Sound interaction with these bottoms are also lossy. For a true model, information about a specific geographical position would be required, but using empirical values for certain bottom types a general approximation can be made. In this case, the second medium being a solid means that it supports both shear and compressional waves with sound speeds of c_s and c_p. The reflection coefficient,

R, is defined in equation 4.2 and the total effective impedance is defined in equation 4.3.

$$R = \frac{Z_{tot} - Z_1}{Z_{tot} + Z_1} \tag{4.2}$$

$$Z_{tot} = Z_p \cos^2 \theta_s + Z_s \sin^2 2\theta_s \tag{4.3}$$

Using the reflection coefficient from 4.2, the bottom loss can be defined as in equation 4.4.

$$BL = -10 \log |R|^2 \tag{4.4}$$

In order to show how much bottom type, and thus c_p and c_s, can affect the reflectivity conditions, figure 4-3 shows the bottom loss for several different seafloor types based on the values in table 4.1, both from ref. [4]. The values for ocean bottom loss at low angles varies from less than 1dB per bounce, for hard surfaces with good reflectivity, to up to 15dB per bounce for bottoms with poor reflectivity, and the response of each bottom is independent of frequency, in this model.

Bottom type	ρ (percent)	ρ_b/ρ_w	c_p/c_w	c_p (m/s)	c_s (m/s)	α_p (dB/λ_p)	α_s (dB/λ_s)
Clay	70	1.5	1.00	1500	$<$100	0.2	1.0
Silt	55	1.7	1.05	1575	$80z^{0.3}$	1.0	1.5
Sand	45	1.9	1.1	1650	$110z^{0.3}$	0.8	2.5
Gravel	35	2.0	1.2	1800	$180z^{0.3}$	0.6	1.5
Moraine	25	2.1	1.3	1950	600	0.4	1.0
Chalk	-	2.2	1.6	2400	1000	0.2	0.5
Limestone	-	2.4	2.0	3000	1500	0.1	0.2
Basalt	-	2.7	3.5	5250	2500	0.1	0.2

Table 4.1: Bottom parameters for multiple bottom types

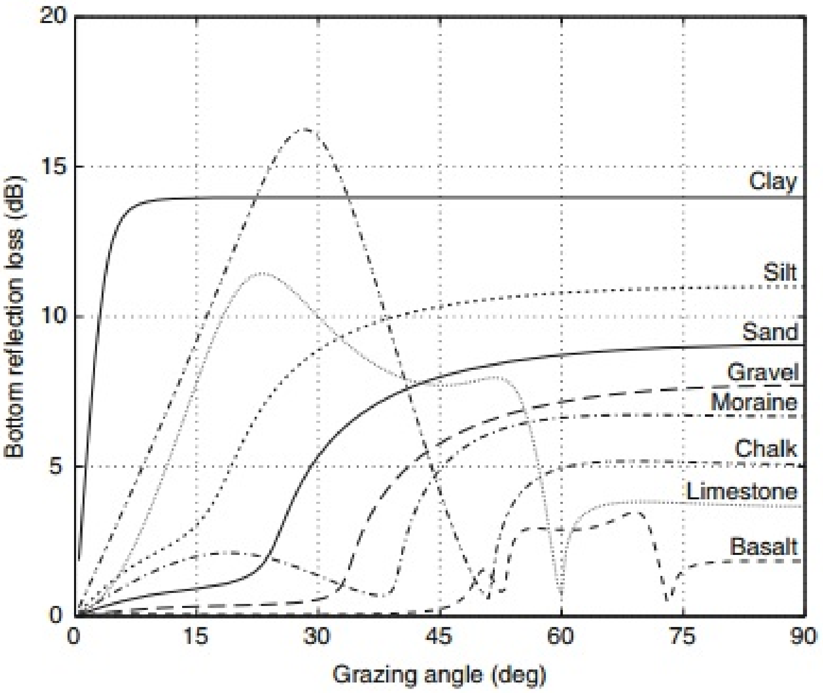

Figure 4-3: Bottom Loss for Various Bottom Types

Figure 4-4 from reference [4], shows the effects of varying parameters on bottom loss, based on values in table 4.2. In (a) it can be seen that compressional wave speed has an effect on bottom loss at all grazing angles and is instrumental in determining the critical angle for bottom loss. A lower c_p results in higher losses. Figure 4-4 (b) shows the effect of α_p. The loss in this graph exhibits a heavy dependence on α_p. Part (c) displays the effect of various bottom densities. Below the critical angle, ρ_{bottom} has little effect and is not a concern in modeling longer range problems. Finally, in figure 4-4 (d), the effects of a varying shear speed can be seen to have an effect below and above the critical angle and, because it takes the role of compressional wave speed in in sedimentary bottoms, can be very important to overall bottom loss.

	c_p (m/s)	c_s (m/s)	α_p (dB/λ_p)	α_s (dB/λ_s)	ρ (kg/m^3)
Case(a)	-	0	0.5	0	2000
Case(b)	1600	0	-	0	2000
Case(c)	1600	0	0.5	0	-
Case(d)	1600	-	0.0	0	2000

Table 4.2: Bottom parameters for figure 4-4

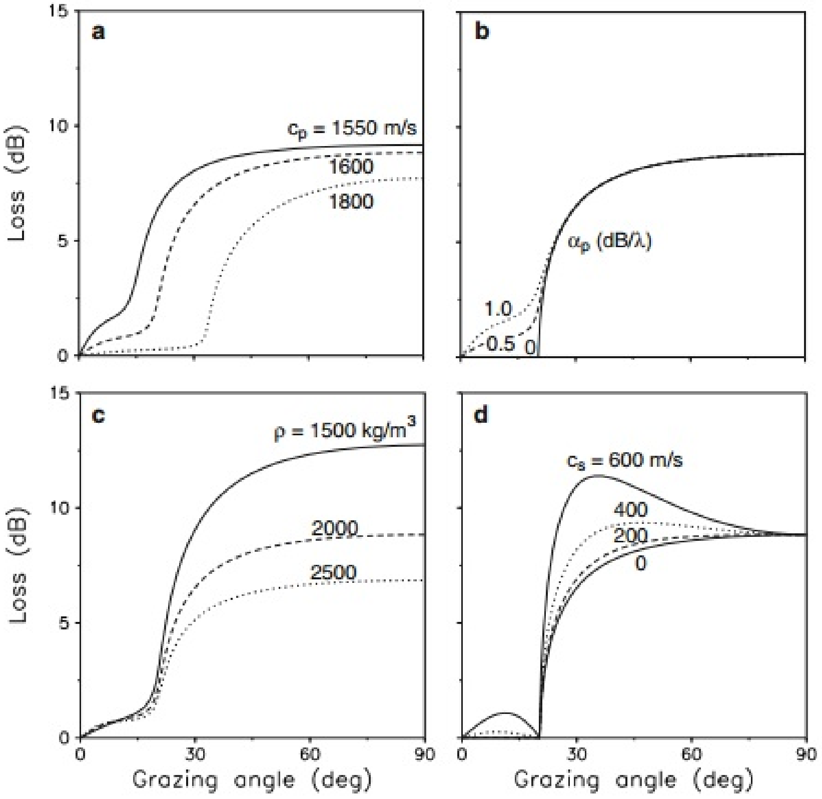

Figure 4-4: Bottom Loss in a Liquid-Solid Half Space

4.3 APL-UW Bottom Reverberation Model

The next model, developed at the University of Washington Applied Physics Laboratory and discussed in reference [5], provides a physics based approach to geoacoustic modeling of bottom reverberation. This model requires the use of six parameters to define the bottom type. They include the density ratio of sediment to water mass density, the ratio of sediment to water sound speed, the ratio of the imaginary wavenumber to the real wavenumber in the sediment, the ratio of sediment volume scattering cross section to the sediment attenuation coefficient, the spectral exponent of the bottom, and the spectral strength of the bottom. The last two parameters were determined through field testing and fitting of the model to real world experimental values. These numbers would be best determined through direct measurement at the experiment site but can otherwise be obtained from a table of average values in ref. [5]. Equation 4.5 defines bottom backscattering strength as a function of two dimensionless quantities: $\sigma_r(\theta)$, the backscattering cross section per unit solid angle per unit area due to roughness of the bottom interface, and $\sigma_v(\theta)$, the backscattering cross section per unit solid angle per unit area due to volume scattering from below the interface.

$$S_b(\theta) = 10\log[\sigma_r(\theta) + \sigma_v(\theta)] \tag{4.5}$$

4.3.1 Roughness Scattering Cross Section, $\sigma_r(\theta)$

For smooth bottoms, $\sigma_r(\theta)$ can be represented by a Kirchoff approximation at for grazing angles near 90 degrees and a composite roughness approximation for other angles. $\sigma_r(\theta)$ is determined empirically for rough bottom types. The calculations for both are exhaustive and described in section IV of ref. [5] and ultimately yields equation 4.6 for $\sigma_r(\theta)$, $\sigma_{int}(\theta)$ is the interpolated cross-section between the two approximations and $\sigma_{lr}(\theta)$ is the large roughness cross-section.

$$\sigma_r(\theta) = f(y)\sigma_{int}(\theta) + [1 - f(y)]\sigma_{lr}(\theta) \tag{4.6}$$

$$y = \frac{\tan^{-1} s - \theta_r}{\Delta\theta} \tag{4.7}$$

where

$$\sigma_r = 7.5^\circ \tag{4.8}$$

$$\Delta\theta = 0.5^\circ \tag{4.9}$$

Both are reference values and determined empirically.

4.3.2 Sediment Volume Scattering Cross Section, $\sigma_v(\theta)$

$\sigma_v(\theta)$ is determined by the bottom material including both refraction and transmission loss at the interface.

$$\sigma_v(\theta) = S(\theta, s)F(\theta, \sigma_{pr}, s) \tag{4.10}$$

Where $S(\theta, s)$ and $F(\theta, \sigma_{pr}, s)$ are used to account for the refraction and transmission loss and include shadowing and bottom slope corrections. Their derivation is described in reference [5].

4.3.3 Limits and Accuracy

A lack of simultaneous acoustic and geoacoustic data with adequate detail to construct a proper representation of the fluid-solid interface constrain the model accuracy to the uppermost layers of any bottom type. Because of the lack of data, the model is based on the physical characteristics of the bottom type and is, therefore, heavily dependent on the quality of the data regarding the bottom type. The physics involved assume the bottom to be a simple fluid that is statistically homogeneous in the horizontal and vertical. This keeps the physics simple enough for implementation.

It is possible to increase the accuracy by taking reverberation measurements in the region of operation and using that data to fit values for model parameters. This limits the negative effects of using estimated values especially grain size, which can quickly induce large errors. As the quality of bottom characterization improves, the model accuracy improves. Errors reach up to 10dB for rough bottoms, and up to 3dB for a well-defined smooth bottom.

The model performs better with larger grazing angles and smoother bottom types. The model incorporates all of the current known mechanisms for backscatter and can be used to determine values for a wide range of frequencies, bottom types, and grazing angles with a reasonable expectation of accuracy. The model is consistent with existing data except the model shows a more rapid falloff of backscatter strength as grazing angle decreases than previous data suggests. This falloff is consistent with more recent measurement data [5].

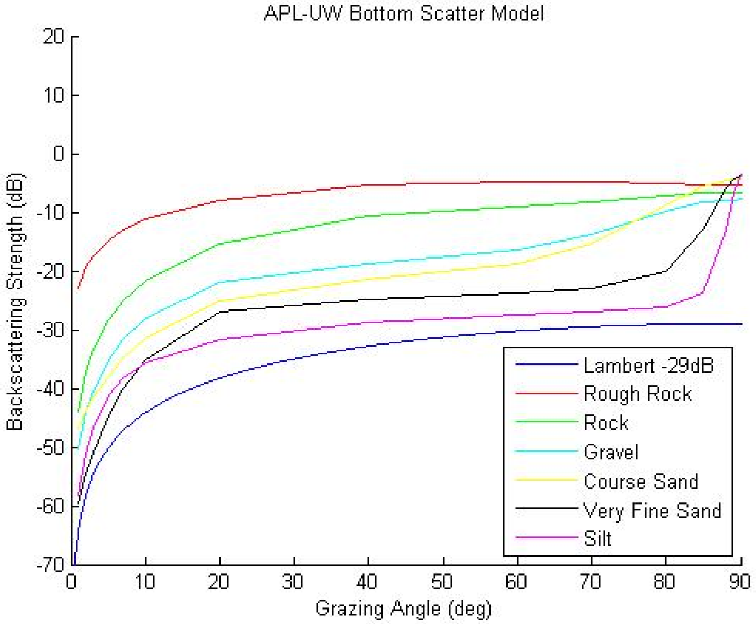

Figure 4-5: APL-UW Backscatter Model

Figure 4-5 provides a look at scattering strength versus grazing angle for a variety of bottom types and was developed using bottom data from reference [5]. The Lambert scattering model with an offset of -29dB is provided for reference.

4.4 NRL Model

More recently, the Naval Research Laboratory (NRL) developed a semi-empirical model for determining scattering strength off of the ocean bottom using a small slope approximation. The equations in the model are developed from data sets collected over years of sea trials and experimentation. Using a variety of systems and varying waveforms, NRL was able to make direct measurements of bottom scattering for a wide range of bottom conditions, water depth, and frequencies under the Space and Naval Warfare Systems Command's Critical Sea Test program and the Office of Naval Research's Littoral Warfare Advanced Development project. These measurements were then correlated to the archived bottom data [9]. Because of the overwhelming amount of data provided as to it's accuracy and completeness from the direct measurements archived data, this model was ultimately chosen for implementation.

4.4.1 General Derivation

Scattering strength is defined in equation 4.11. The scattering geometry is shown in figure 4-6. These angles are used throughout the derivation of scattering strength to explain the acoustic interaction with the ocean surface. In the equation below, k and q are the incoming and outgoing wavevectors used to define θ_{inc} and θ_{scat}.

$$SS = 10 \log \sigma \tag{4.11}$$

$$[\mathbf{k_h} \bullet \mathbf{q_h}] = k_0^2 \cos\theta_{inc} \cos\theta_{scat} \cos\phi_{bi} \tag{4.12}$$

Later parts of the derivation will depend on the difference between $\mathbf{k}$ and $\mathbf{q}$, where $\mathbf{k} - \mathbf{q} = (\mathbf{Q}_h, \mathbf{Q}_z)$. The magnitude of $\mathbf{Q}_h$ can then be expressed in terms of the geometry.

$$|\mathbf{Q}_h| = k_0 \sqrt{\cos^2\theta_{inc} + \cos^2\theta_{scat} - 2\cos\theta_{inc}\cos\theta_{scat}\cos\phi_{bi}} \tag{4.13}$$

Many models have been used to describe scattering at rough ocean boundaries. These include first-order approximations, composite roughness models, small slope approximations, and high order perturbation theories. This model uses a small slope approximation in conjunction with a first-order perturbation approximation in order to increase accuracy of

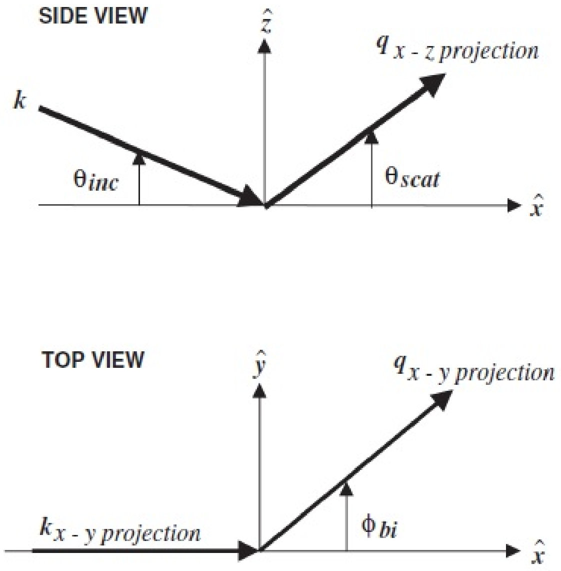

Figure 4-6: Angle Geometry

Variable Definitions	
σ	Scattering cross-section/unit area/unit solid angle
$\lvert\mathbf{Q}_h\rvert$	Difference in horizontal wave vector magnitude
$W(\mathbf{Q}_h)$	2D roughness spectral density
σ_{int}	Noncoherent component of the scattering cross section for a rough interface
c_p	Compressional wave speed and attenuation in the bottom
c_s	Shear wave speed and attenuation in the bottom
β	"Algebraic form" dependent on boundary conditions
θ_{inc}	Grazing angle at the bottom
θ_{scat}	Angle of the scattered ray
ϕ_{bi}	Difference in azimuth between θ_{inc} and θ_{scat}
ρ_{bot}	Ratio of solid mass density to fluid mass density
c_0	Sound speed in water at the interface
w_2	Spectral Strength
γ_2	Parameter describing frequency dependence
$\mathbf{I}$	Spatial spectrum integral dependent on $\mathbf{Q}_h$ and Q_z

Table 4.3: Variables used in the NRL bottom scattering model

the model output. Using this process, the bottom roughness is modeled using $W(\mathbf{Q}_h)$.

$$W(\mathbf{Q}_h) = \frac{w_2}{(h_0|\mathbf{Q}_h|)^{\gamma_2}} \tag{4.14}$$

This model assumes all rough surface scattering to be incoherent, so the coherent portion is removed. If the coherent portion was included, conservation of energy would be violated due to an artifact in the spectrum of equation 4.14. This spectra also causes an issue with scattering near the specular direction by introducing a sharp peak in σ_{int}.

$$\sigma_{int} = \frac{1}{8\pi}\left|\frac{\beta}{|\mathbf{Q}_h|Q_z}\right|^2 \bullet \mathbf{I} \tag{4.15}$$

The coefficient in front of the integral, **I**, provides the small slope approximation. The integral, **I**, is easily defined but hard to evaluate due to intense oscillation.

$$\mathbf{I}(\alpha) = \frac{2}{\pi}\sum(-)^{n+1}\sin(\pi n v)\frac{\Gamma^2(nv+1)}{\Gamma(n+1)}\left(4^v\alpha\right)^n \tag{4.16}$$

From here, a first-order approximation is then made from **I** yielding:

$$\sigma_{int} = \left|\frac{\beta^2}{2}\right| W(\mathbf{Q}_h) \tag{4.17}$$

These equations have been general in nature. To apply them to the ocean bottom, specific boundary conditions must be determined and applied.

4.4.2 Ocean Bottom Specifics

Using c_p and c_s, the acoustic compressional and shear wavenumbers are defined as:

$$\begin{aligned} k_p &= \frac{2\pi f}{c_p}, \\ k_s &= \frac{2\pi f}{c_s} \end{aligned} \tag{4.18}$$

Three general functions are then defined. These expressions are used to construct β further on.

$$\begin{aligned} a_j(\theta_j;\rho_{bot},k_0,k_p,k_s) &= \rho_{bot}\frac{k_0\sin\theta_j}{P_j}\left[1-\frac{4k_0^2\cos^2\theta_j S_j(S_j-P_j)}{k_s^4}\right], \\ b_j(\theta_j;\rho_{bot},k_0,k_p,k_s) &= \rho_{bot}\frac{k_0\sin\theta_j}{P_j}\left[1-\frac{2k_0^2\cos^2\theta_j}{k_s^2}\right], \\ \xi(\theta_j;k_0,k_p,k_s) &= \frac{k_0\sin\theta_j}{k_s^2 P_j}\left[(S_j-P_j)^2-k_p^2\right] \end{aligned} \tag{4.19}$$

where

$$\begin{aligned} P_j &= \sqrt{k_p^2-k_0^2\cos^2\theta_j}, \\ S_j &= \sqrt{k_s^2-k_0^2\cos^2\theta_j}, \\ \theta_1 &= \theta_{inc}, \\ \theta_2 &= \theta_{scat} \end{aligned} \tag{4.20}$$

With all of the preceding definitions, an expression for β can be developed.

$$\begin{aligned} &\beta(\theta_{inc},\theta_{scat},\phi_{bi};rho_{bot},k_0,k_s,k_p) = \\ &\frac{1}{(a_1+1)(a_2+1)}[4(1-\rho_{bot})\left([\mathbf{k}_h\bullet\mathbf{q}_h]\,\xi_1\xi_2-k_0^2\sin\theta_{inc}\sin\theta_{scat}\right) \\ &+\frac{8\rho_{bot}}{k_s^2}\left[\mathbf{k}_h\bullet\mathbf{q}_h\right]^2\xi_1\xi_2-4\left[\mathbf{k}_h\bullet\mathbf{q}_h\right](a_1-\xi_1)(a_2-\xi_2)+4k_0^2a_1a_2 \\ &+\frac{2k_s^2}{\rho_{bot}}(a_1-b_1)(a_2-b_2)-\frac{4k_p^2}{\rho_{bot}}b_1b_2] \end{aligned} \tag{4.21}$$

Now that the expression for β is established, it can be plugged into equation 4.17 and scattering strength, SS, can be solved for with equation 4.11. These equation apply specifically to a fluid-solid interface, but can handle any interface involving loose sediments [9].

Figure 4-7 shows the backscattering strength versus grazing angle using the NRL model. Table 4.4 contains the values used for the geoacoustic parameters.

	ρ_{bot}	c_p/c_0	c_s/c_0	γ_2	w_2 (m^4)
Mud	1.4	0.99-0.002i	0.1-0.0004i	3.3	0.000518
Sand	2.0	1.2-0.005i	0.3-0.077i	3.3	0.006957
Limestone	2.4	2.7-0.004i	1.5-0.003i	3.3	0.004
Basalt	2.7	3.4-0.006i	1.8-0.006i	3.3	0.01862

Table 4.4: Geoacoustic parameters for figure 4-7

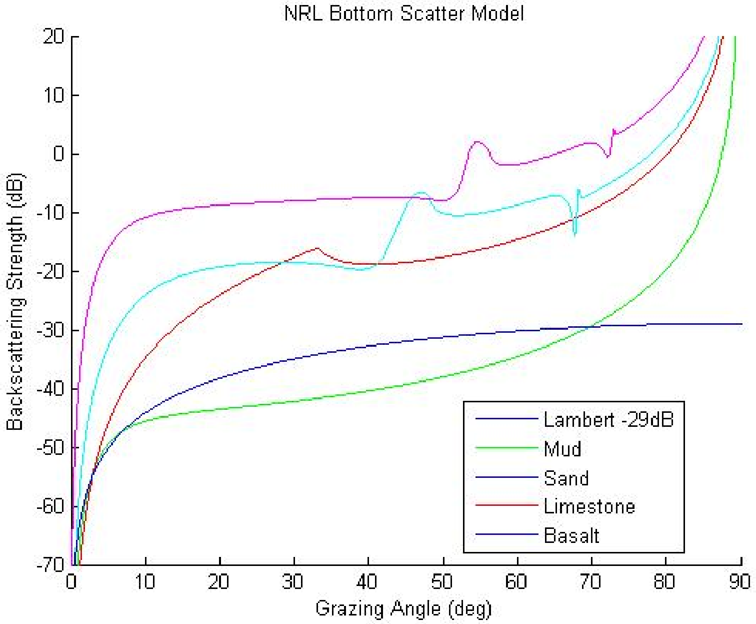

Figure 4-7: NRL Backscatter Strength versus Grazing Angle

Chapter 5

Results and Conclusion

5.1 Testing and Results

In chapter 4, the NRL method for calculating bottom scattering was fully described. This model was then implemented and it's results verified against reference [9]. The next step on the way to results was to use this model to calculate reverberation level at the source/receiver. This was initially done by establishing a simulation environment with a single source/receiver at the ocean surface. Using an isovelocity profile, with a sound speed in water, c_w of 1500 m/s, a fan of rays with incident angles of 0 to 90 degrees were modeled to travel to the ocean bottom at a depth of 5000 m. Transmission loss was calculated assuming spherical spreading using equation 5.1 where D is the distance traveled by the ray. The rays travel from the source to the bottom along a straight line path and then the reverberation signal returns to the source with different rays arriving at different times based on the grazing angle and overall distance traveled.

$$\begin{aligned} TL &= 20\log(D) \\ D &= \frac{depth}{\sin\theta_{inc}} \end{aligned} \tag{5.1}$$

Scattering strength, SS, at the ocean bottom was calculated using the NRL method outlined in chapter 4.

Reverberation level transmitted back to the source/receiver is determined using equation 5.2.

$$RL_s = SL - 2TL + SS + 10\log A \tag{5.2}$$

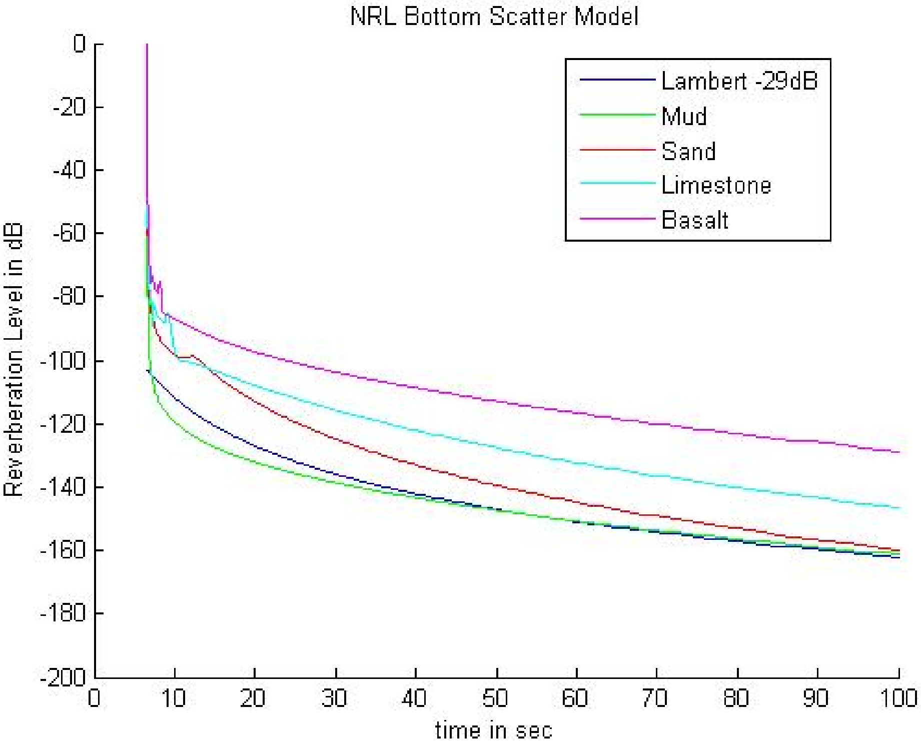

Figure 5-1: Reverberation Level versus Time

where

$$A = \frac{c_w \tau}{2} \phi \frac{r}{\sin \theta_{inc}} \qquad \phi = 2\pi \tag{5.3}$$

$10 \log A$ is a term representing the area the ray covers at the boundary due to its spreading from a single point as it travels. Transmission Loss is multiplied by a factor of two to account for the loss from source to bottom and back to the source. SL is the original source strength and is set to 0 for the purposes of testing. τ is the pulse length of the source signal [1]. A value of 1 sec was used. Figure 5-1 shows the results. Included in the model were several bottom types with geoacoustic properties as defined in table 4.4 with an additional set of values for the Lambert's Law -29dB approximation.

Lower grazing angles mean more time traveled to reach the bottom and return. It can be seen in figure 5-1 that with large grazing angles, and subsequently shorter time traveled, the bottom type has a large effect and variations in shape from the Lambert's Law -29dB are in evidence. As time continues on and grazing angles get shallower, there is a smaller effect from bottom reverberation and transmission loss over distance becomes dominant with the curve continuing downward at a decreasing rate. All of this matches with expected behavior.

5.1.1 Integration with BELLHOP

The next step in implementing the model included integration with the BELLHOP beam tracing model. BELLHOP, as discussed in chapter 2, takes an environmental file, .env, and outputs one of several different file types based on selections made in the .env file. The data runs used produced both a .ray file and a .arr file. The .ray file can then be used with the BELLHOP MATLAB command plotray. This produces a plot of the rays with depth and range. The curvature of each ray is based on the sound speed profile (SSP) described in the .env file. Utilizing the .ray file and it's plot, the ray paths were analyzed to determine the footprint, in range, over which the rays made their first bottom interaction.

The second run of BELLHOP was used to produce a .arr file by setting the maximum range to that of the footprint determined from the .ray file. The .arr file determines the eigenrays that connect the source to the receiver array and describes each ray as it interacts with the receivers. The data provided by the .arr includes the amplitude of the ray, the phase as compared to the source, the travel time between the source and the receiver, the take-off angle at the source, the take-off angle at the receiver, the number of surface interactions, and the number of bottom interactions [8].

Several environments were tested with the NRL bottom scattering model in order to ensure it worked over multiple ocean, source, and target profiles. For each of the data runs, the basics of the ocean environment were set up in a similar way. A single source was placed near the surface with a downward facing beam. The beamwidths used were between 20 and 30 degrees and steered between the horizontal and the ocean bottom. This ensured that the rays would interact with the bottom before the surface in most environments. The frequency used in each model was 3000 Hz. These values for beamwidth and frequency are commensurate with the sources used by LAMSS at MIT for at sea experiments. To calculate the reverberation, the receiver was modeled as a long string of 200 separate receivers evenly spaced along the ocean bottom. The ray plot from the .ray file was used to determine the range over which the first bottom bounce for each ray occurred. This range was then used to set the range over which the receiver array was spread. A large number of beams, <500, was used to ensure that a sufficient number of beams existed to guarantee interaction with the receiver array.

Once all of the above selections were made, BELLHOP was then used to develop the .arr file. This file included the information for each receiver as discussed above. Using the script bottomreverb.m, included as Appendix B, each .arr file was then parsed for the relevant information. It was first necessary to know if a ray interacted with a given element on the receiver array. If so, then the data for that receiver was parsed to determine the number of bottom interactions as the ray entered the receiver. If that value was zero, meaning this was the first bottom interaction for that ray, then the amplitude, travel time, and take-off angle at the receiver were stored for use in the reverberation level calculation.

For the cases tested, only one ray interaction occurred within each discrete time bin. This meant that none of the rays needed to be added incoherently. For future implementation of the model in C++, it will be necessary to include a way to perform the addition of the incoherent amplitudes in order to get an accurate representation of the total amplitude at a given time. This can easily be done by converting the amplitude into power and from power to dB level. But since no addition was required, the amplitude of the ray could be directly converted to transmission loss, TL, as in equation 5.4.

$$TL = 20\log(Amplitude) \tag{5.4}$$

From here, the remaining step is to use the take-off angle at the receiver to calculate scattering strength. θ_{inc} was set equal to the take-off angle, as was θ_{scat}, assuming that the backscatter is in the same direction as the incident angle. ϕ_{bi} was set to 180°. The bottom of each environment was assumed to be limestone and the values in table 4.4 were used as the applicable geoacoustic properties. This left differences in results independent of the bottom type. The reverberation level was calculated as described in equation 5.2 and plotted versus the travel time. The travel time was calculated using the travel time for the each ray from the .arr file and multiplying by two to account for the time it took the ray to travel from the source to the receiver and then back to the source.

The figures that follow will show the SSP, the ray trace, and the reverberation level for each environment.

5.1.2 Munk

The Munk profile is a common theoretical environment. It is a deep ocean problem with a depth of 5000 m. Due to the nature of the SSP, as seen in figure 5-2, sound curves down toward the bottom and then turns back up as it approaches the bottom. In order to ensure that sound would reach the bottom, the source was placed at a depth of 1000 m. This prevented the development of a convergence zone propagation path. It did, however, still allow for the sound channel between about 1000 m and 2000 m to form for lower take-off angles at the source. This sound channel can be seen in figure 5-3. In figure 5-4, it can be seen that at higher take-off angles, quicker return times, that the effect of bottom backscatter is low but increases with shallower take-off angles at the receiver, longer return times.

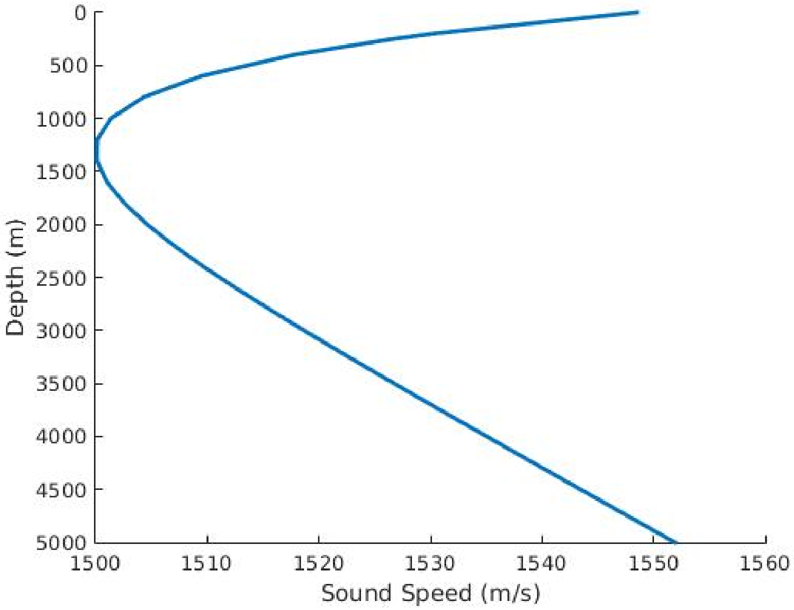

Figure 5-2: Munk SSP

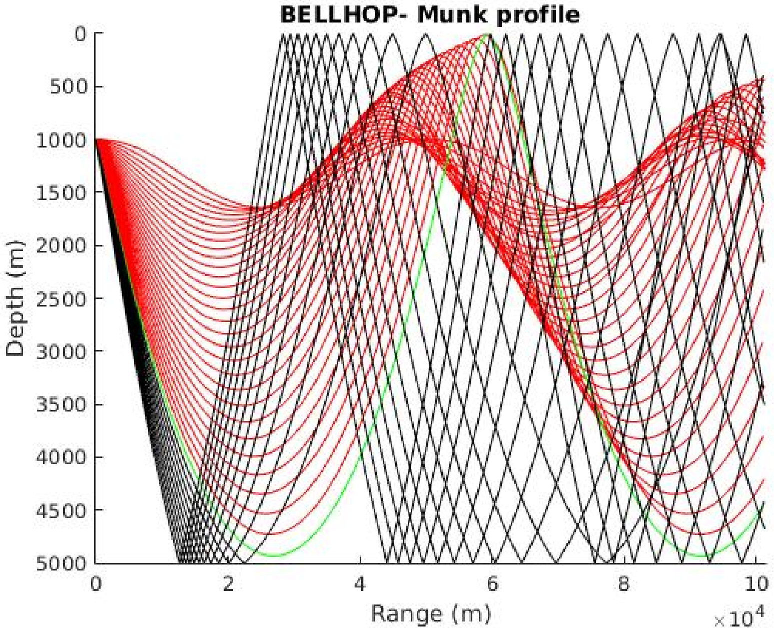

Figure 5-3: Munk Ray Diagram

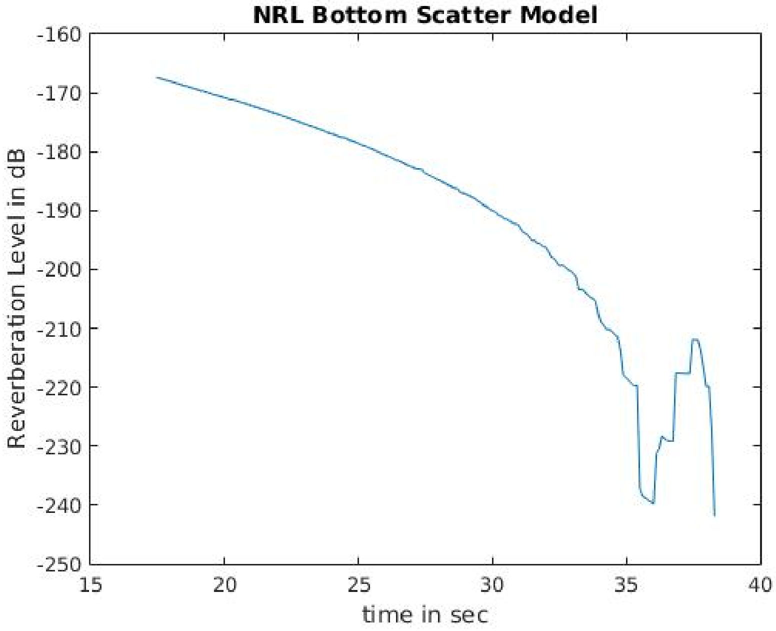

Figure 5-4: Munk Reverberation Level

5.1.3 Isovelocity

A deep ocean, 5000 m, isovelocity profile is not realistic but is good for testing purposes to compare against other models. In this deep, isovelocity environment, the effects of backscatter are very low, forming only very small perturbations in the reverberation level. This continues until the take-off angle at the receiver approaches zero causing the intensity to go to infinity and a spike in reverberation level.

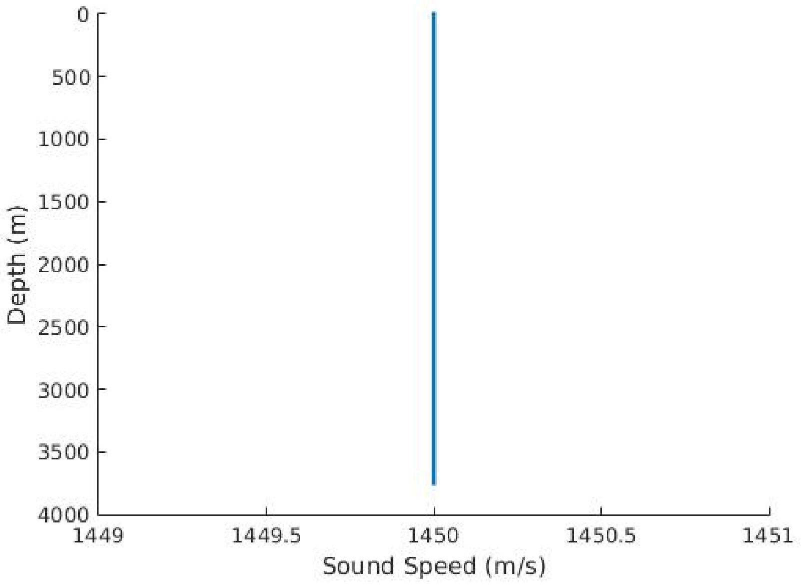

Figure 5-5: Isovelocity SSP

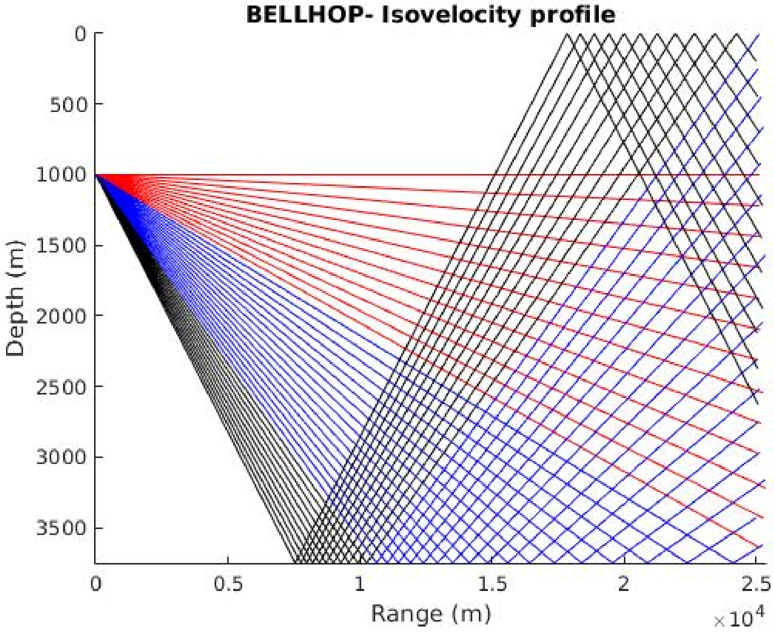

Figure 5-6: Isovelocity Ray Diagram

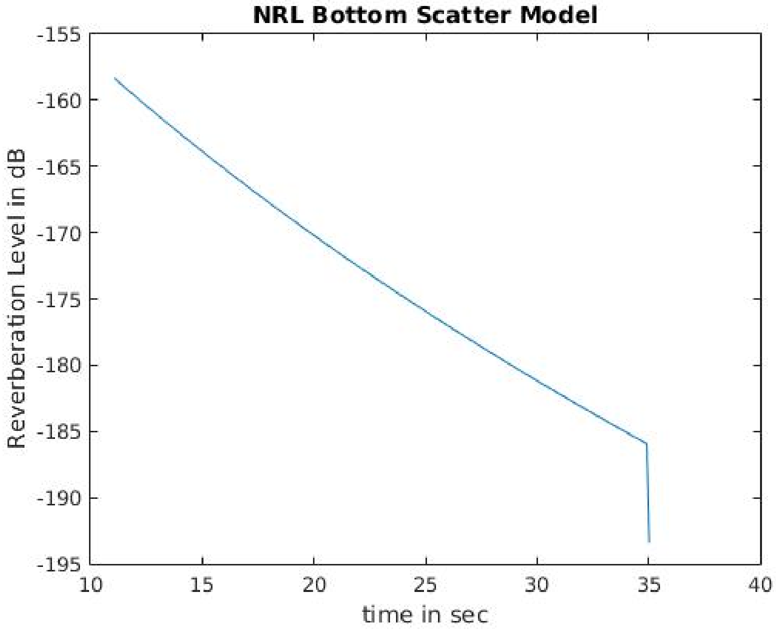

Figure 5-7: Isovelocity Reverberation Level

5.1.4 Deep Arctic

The deep arctic profile is a nearly linear SSP that goes to a depth of 3750 m. This profile behaves in a way similar to the Munk profile, with regard to reverberation. As the take-off angle gets more shallow, the reverberation builds until the take-off angle passes through 0 degrees.

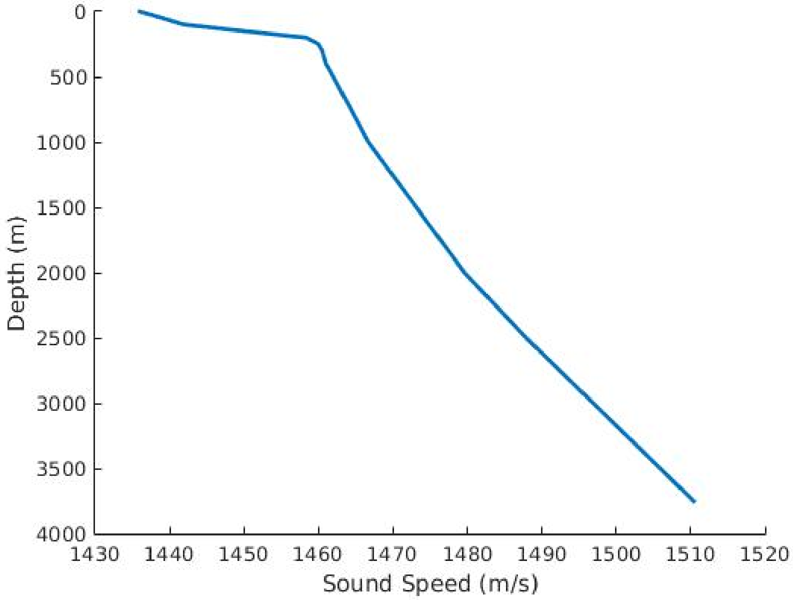

Figure 5-8: Arctic SSP

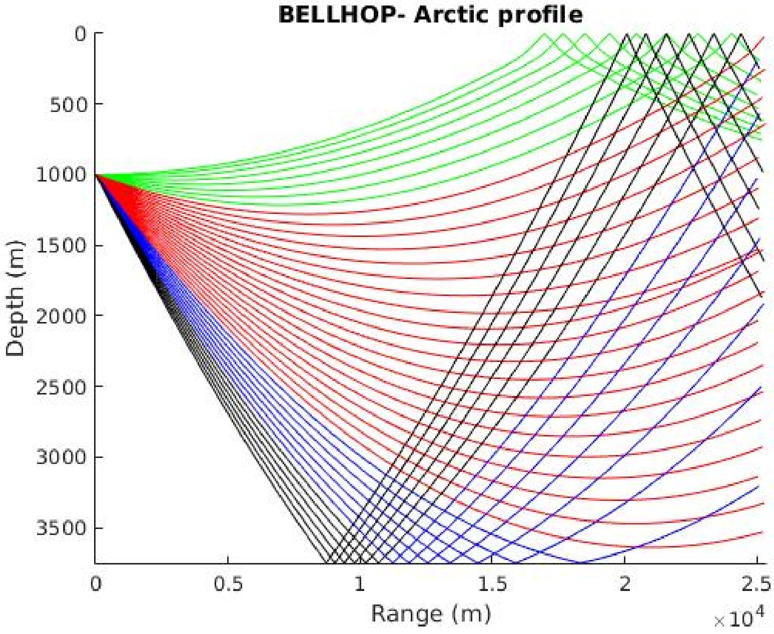

Figure 5-9: Arctic Ray Diagram

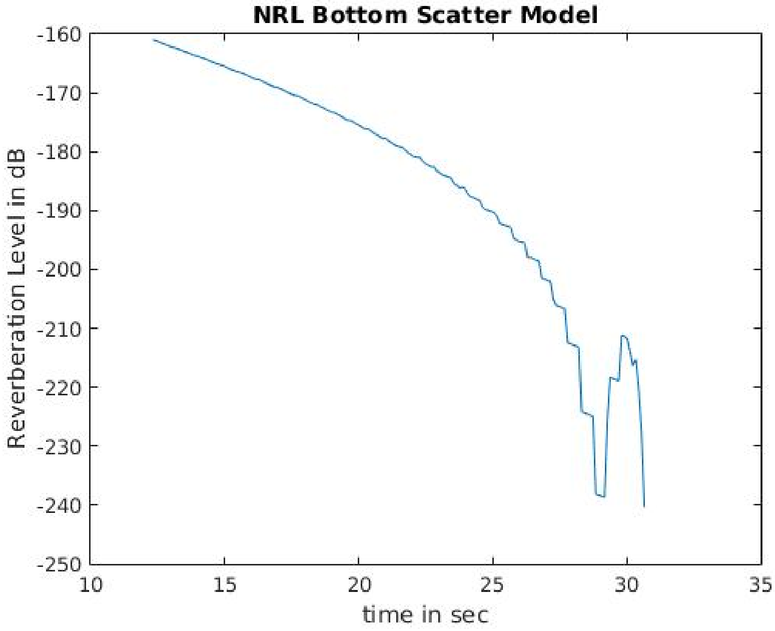

Figure 5-10: Arctic Reverberation Level

5.1.5 Shallow Summer

The shallow summer profile is warmer at the top due to heating by the sun, thus a higher sound speed. This sound speed decreases to a minimum 30 m from the bottom and stays at that value to the final depth of 100 m. The shallow summer profile exhibits the most odd behavior, with regard to reverberation. Due to the multipath propagation, several surface bounces occur that redirect sound back at the bottom in the opposite direction causing many instances of take-off angle passing through 0 and causing spikes in reverberation level. Overall, the reverberation behaves as expected.

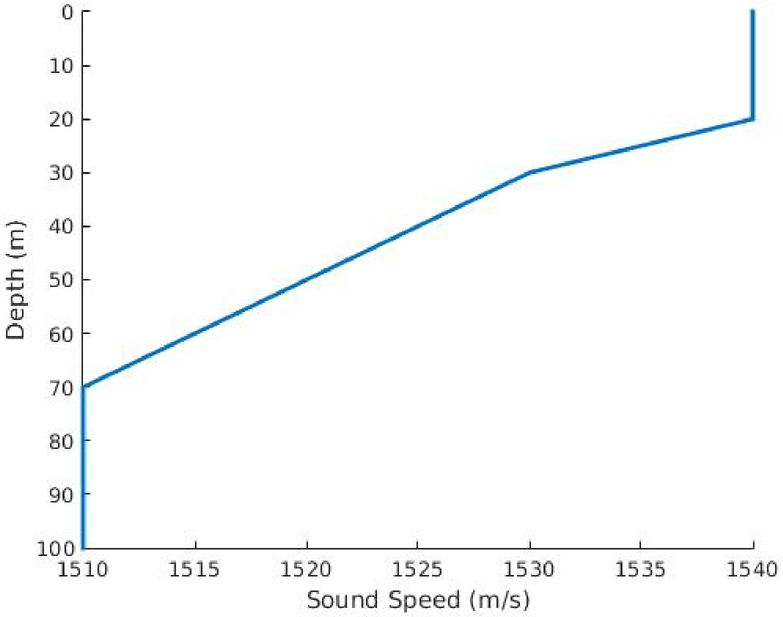

Figure 5-11: Shallow Summer SSP

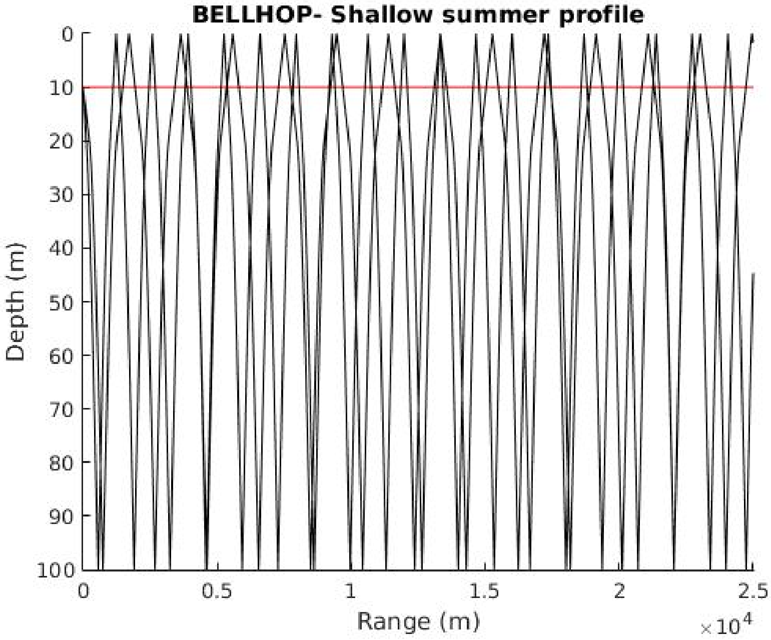

Figure 5-12: Shallow Summer Ray Diagram

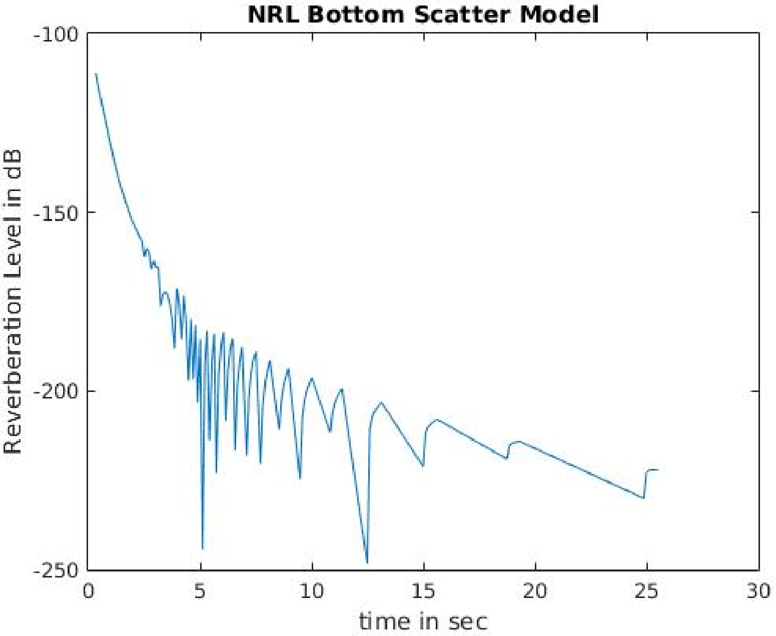

Figure 5-13: Shallow Summer Reverberation Level

5.1.6 Shallow Winter

The shallow winter profile is a 100 m depth, isovelocity profile. Though dominated by transmission loss, we see the same jumps in reverberation as the take-off angle at the receiver approaches 0.

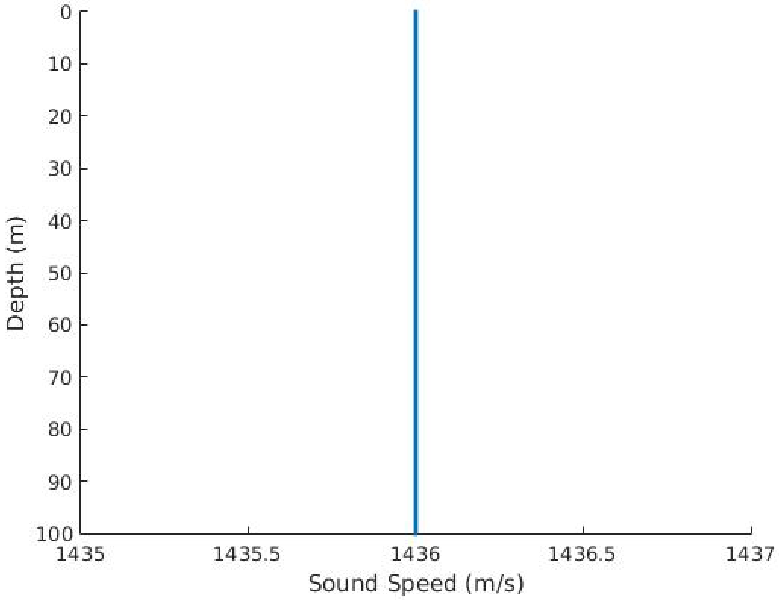

Figure 5-14: Shallow Winter SSP

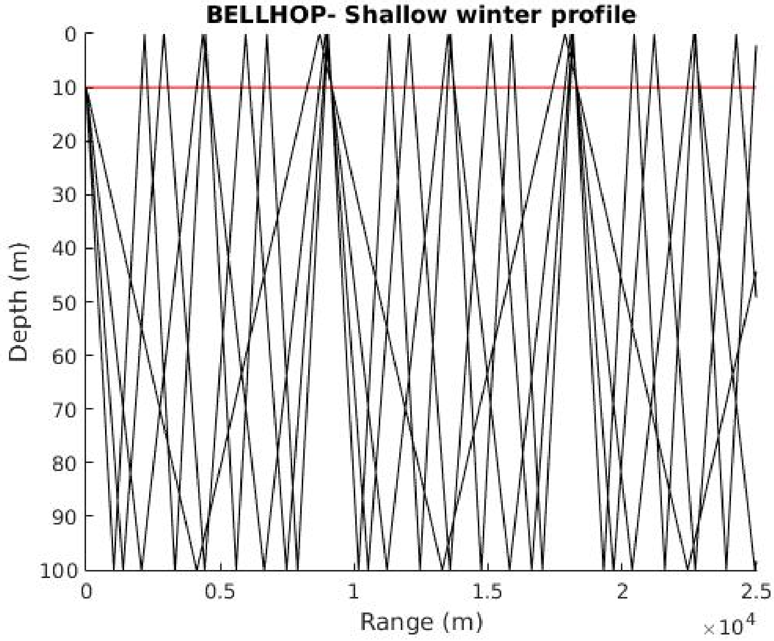

Figure 5-15: Shallow Winter Ray Diagram

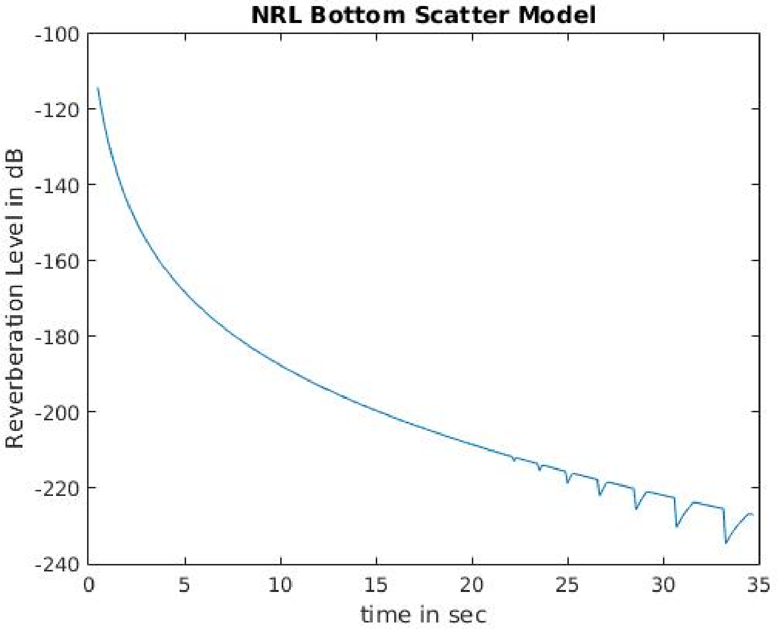

Figure 5-16: Shallow Winter Reverberation Level

5.1.7 Pekeris

The Pekeris profile is a 100 m depth, isovelocity profile that operates in a way similar to the shallow winter profile. Both profiles are dominated by the transmission loss.

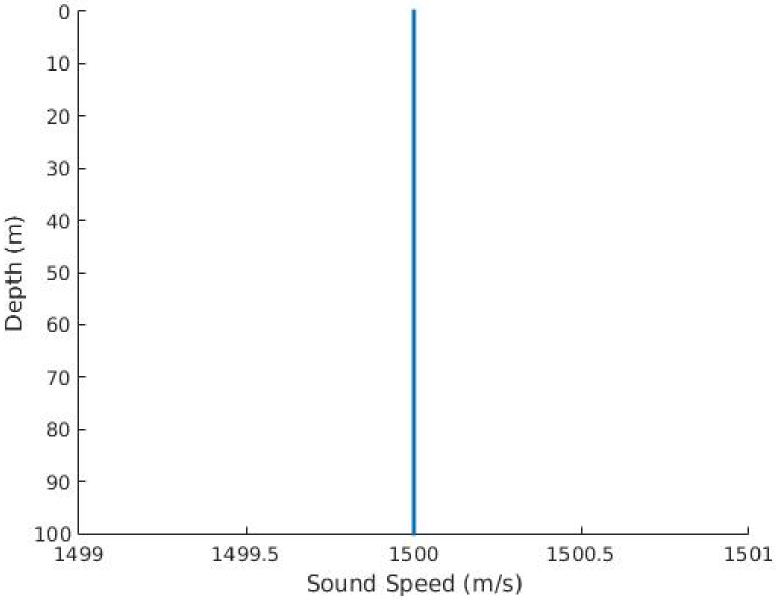

Figure 5-17: Pekeris SSP

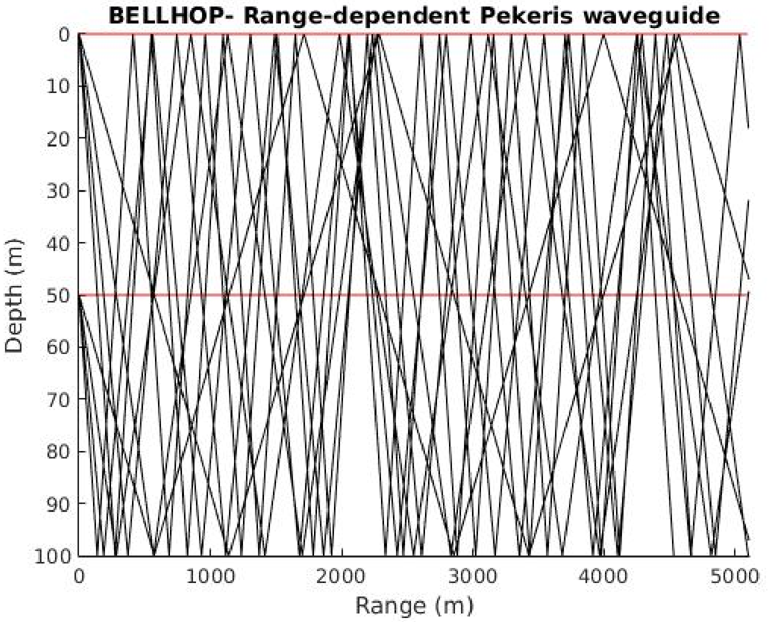

Figure 5-18: Pekeris Ray Diagram

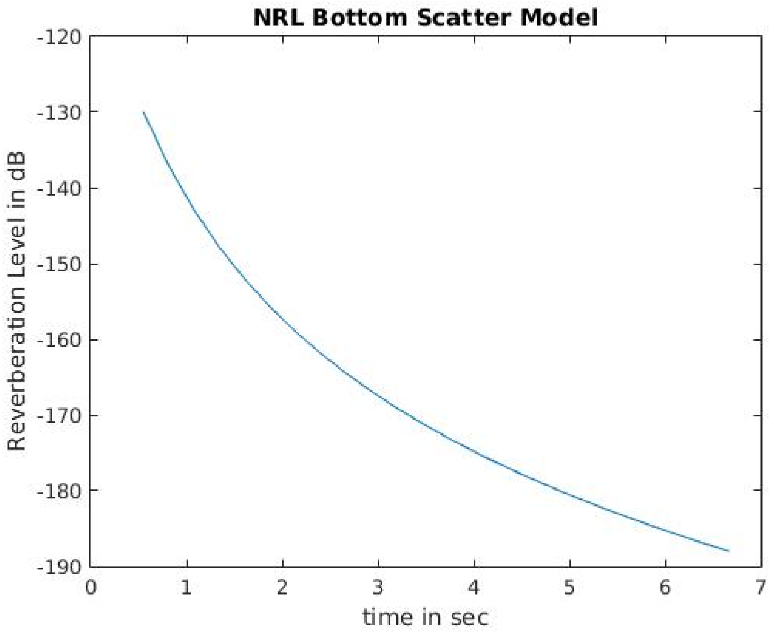

Figure 5-19: Pekeris Reverberation Level

5.1.8 Combined Deep Profiles

The deep water, 3750 m (arctic and isovelocity) and 5000 m (Munk), profiles are shown here combined. It can be seen that the Munk and Arctic profiles have similar reverberation levels with a different return time, due to the shorter return for the different SSP in the arctic and a shorter return time due to a shallower bottom depth.

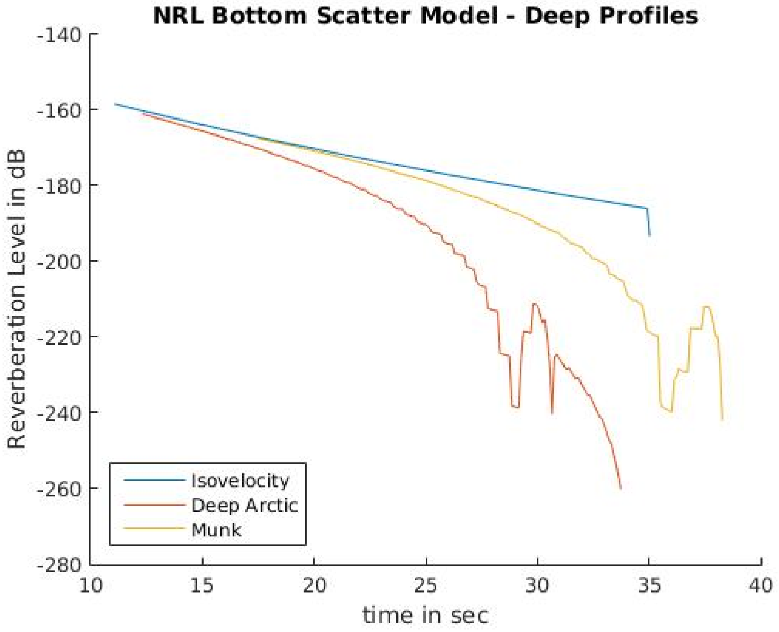

Figure 5-20: Combined Deep Reverberation Level

5.1.9 Combined Shallow Profiles

The shallow water profiles are shown combined here. While shorter in time scale, the Pekeris return is very similar to the shallow winter profile, as expected. The shallow summer has many more jumps than either due to more multipath arrival and the differing SSP but maintains a similar curvature.

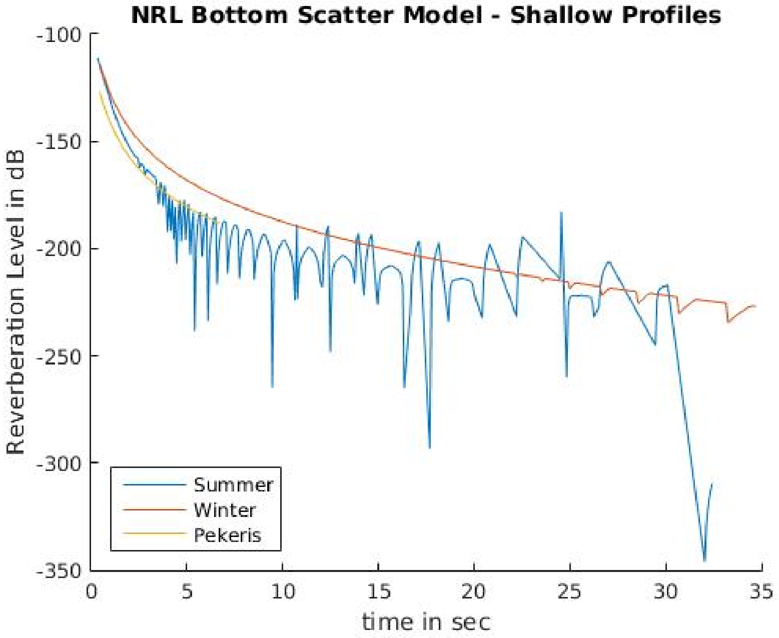

Figure 5-21: Combined Shallow Reverberation Level

5.2 Conclusion and Future Improvements

The current implementation of the NRL bottom model is functional. However, to make the model more useful for LAMSS at MIT, the model will be moved to the LAMSS C++ code base. More specifically, the model will be added to the program uSimActiveSonar. This program takes a source waveform, source strength, and elevation angle as input and outputs a simulated time series of what the hydrophone receiver hears back from the ocean. uSimActiveSonar already includes a set of code for modeling surface reverberation of an acoustic signal. Once implemented in uSimActiveSonar, it will be possible to test the reverberation model with a sonar processor, target tracker, and beamforming program [2].

Another improvement to the simulator would be to add bistatic capabilities to the model. The NRL model is already bistatic and, because the NRL surface reverberation model is already functional in uSimActiveSonar, would not be difficult to implement. This would improve the overall quality of simulated reverberation.

With the reverberation techniques fully implemented in the simulated environment, the processes developed for use on AUVs can be more completely tested before being used in at sea experiments where ship time is expensive and short. By building in the ability to test for all expected environmental impacts before going to sea, research time will be better utilized and the work of finding and fixing code glitches will be reduced, thereby making field work more efficient and more rewarding.

Appendix A

Matlab Code: Testing the NRL Model

```
%Thomas Miller
%Master's Thesis
%Comparing various bottom reverberation models with an isovelocity SSP

%This code will be used to simulate a single plane wave that interacts with
%the ocean floor at a specific angle. Transmission Loss will be calculated
%along the distance traveled, backscatter will be calculated using one of
%several models, and then transmission loss back to the receiver will be
%determined. These values will be added together and used to compare the
%various models over several incident angles. The SSP will be isovelocity
%and isodensity.

clear all
close all
format compact
clc

%%
depth = 5000;
theta_inc = [0:0.1:90]*pi/180;
c_0 = 1500; %isovelocity sound speed in water
```

```
%Lambert's Law and -29dB for bottom

S_b_lambert = -29 + 10*log10((sin(theta_inc)).^2);

%NRL Model

f = 3000; %frequency in Hz
rho_b = [1.4 2.0 2.4 2.7]; %bottom densities for [mud;sand;mudstone;limestone;basalt]
cp = [0.99-0.002*1i 1.2-0.005*1i 2.7-0.004*1i 3.4-0.006*1i]*c_0;
cs = [0.1-0.004*1i 0.3-0.07*1i 1.5-0.003*1i 1.8-0.006*1i]*c_0;
gamma_2 = [3.3 3.3 3.3 3.3];
w_2 = [0.000518 0.006957 0.004 0.01862];
h_0 = 1;

color_bar = ['g' 'r' 'c' 'm'];
legend_bar = ['Mud' 'Sand' 'Limestone' 'Basalt'];

theta_scat = theta_inc;
sigma_bi = pi;

k_0 = 2*pi*f/c_0;

for(n=1:length(rho_b))

    for(m=1:length(theta_inc))

        theta1 = theta_inc(m);
        theta2 = theta_scat(m);

        kp = 2*pi*f/cp(n);
        ks = 2*pi*f/cs(n);

        kq = k_0^2*cos(theta1)*cos(theta2)*cos(sigma_bi);

        p1 = sqrt(kp^2-k_0^2*(cos(theta1))^2);
        p2 = sqrt(kp^2-k_0^2*(cos(theta2))^2);
        s1 = sqrt(ks^2-k_0^2*(cos(theta1))^2);
        s2 = sqrt(ks^2-k_0^2*(cos(theta2))^2);
        a1 = rho_b(n)*k_0*sin(theta1)/p1*(1-(4*k_0^2*...
            (cos(theta1))^2*s1*(s1-p1)/ks^4));
        a2 = rho_b(n)*k_0*sin(theta2)/p2*(1-(4*k_0^2*...
            (cos(theta2))^2*s2*(s2-p2)/ks^4));

        b1 = rho_b(n)*k_0*sin(theta1)/p1*(1-(2*k_0^2*...
            (cos(theta1))^2/ks^2));
        b2 = rho_b(n)*k_0*sin(theta2)/p2*(1-(2*k_0^2*...
            (cos(theta2))^2/ks^2));

        zeta1 = (k_0*sin(theta1)/(ks^2*p1))*((s1-p1)^2-kp^2);
        zeta2 = (k_0*sin(theta2)/(ks^2*p2))*((s2-p2)^2-kp^2);

        Q_h = k_0*sqrt((cos(theta1))^2+(cos(theta2))^2-2*...
            cos(theta1)*cos(theta2)*cos(sigma_bi));
        W_Qh = w_2(n)/((h_0*Q_h)^gamma_2(n));

        beta = (1/((a1+1)*(a2+1)))*(4*(1-rho_b(n))*(kq*zeta1*zeta2-k_0^2*...
            sin(theta1)*sin(theta2))+(8*rho_b(n)/ks^2)*kq^2*zeta1...
            *zeta2-4*kq*(a1-zeta1)*(a2-zeta2)+4*k_0^2*a1*a2+...
            (2*ks^2/rho_b(n))*(a1-b1)*(a2-b2)-4*kp^2*b1*b2/rho_b(n));
        %beta = -4*k_0^2*sin(theta1)*sin(theta2);
        sigma(m) = abs(beta/2)^2*W_Qh;

    end

    figure(1)
    hold on
    plot(theta_inc*180/pi,S_b_lambert)
    S_b_nrl = 10*log10(sigma);
    plot(theta_inc*180/pi,S_b_nrl,color_bar(n));

end
ylabel('Backscattering Strength (dB)')
xlabel('Grazing Angle (deg)')
title('NRL Bottom Scatter Model')
axis([0 90 -70 20])
legend('Lambert -29dB','Mud','Sand','Limestone','Basalt','Location','Best')
```

```
hold off

%%

%APL-UW Model

S_b_lambert = -29 + 10*log10((sin(theta_inc)).^2);
figure(2)
hold on
plot(theta_inc*180/pi,S_b_lambert)

theta_inc_apl = [1 2 3 5 7 10 20 40 60 70 80 85 88 89 90];
S_b_apl_rr = -[23 19.4 17.4 14.8 13 11.3 7.9 5.4 4.9 5.0 5.1 5.3 5.4 5.4 5.4];
S_b_apl_r = -[44 37.3 33.3 28.4 25.2 21.7 15.4 10.6 9.1 8.3 7.3 6.8 6.6 6.6 6.6];
S_b_apl_g = -[50.3 44.2 40.3 35.1 31.7 28.1 22.1 18.9 16.5 13.7 9.9 8.4 8.0 7.9 7.8];
S_b_apl_cs = -[47.1 43.8 41.5 37.9 34.9 31.5 25.1 21.4 18.8 15.3 8.9 5.6 4.6 4.4 4.4];
S_b_apl_vfs = -[59.7 54.7 50.8 44.6 40.1 35.2 27.1 24.9 23.7 23.1 20.2 13.1 5.9 4.3 3.8];
S_b_apl_silt = -[58.2 51.1 46.6 41.3 38.3 35.7 31.8 28.8 27.4 27 26.3 23.9 13.1 6.8 3.6];

plot(theta_inc_apl,S_b_apl_rr,'r')
plot(theta_inc_apl,S_b_apl_r,'g')
plot(theta_inc_apl,S_b_apl_g,'c')
plot(theta_inc_apl,S_b_apl_cs,'y')
plot(theta_inc_apl,S_b_apl_vfs,'k')
plot(theta_inc_apl,S_b_apl_silt,'m')

axis([0 90 -70 20])
ylabel('Backscattering Strength (dB)')
xlabel('Grazing Angle (deg)')
title('APL-UW Bottom Scatter Model')
legend('Lambert -29dB','Rough Rock','Rock','Gravel','Course Sand',...
    'Very Fine Sand','Silt','Location','SouthEast')
hold off

%%

% Fluid-Solid Interface
figure(3)
hold on
```

```
f = 100;
omega = 2*pi*f;
cw = 1500;
rhow = 1000;
k1 = omega/cw;

theta_inc = [0.1:0.1:90]*pi/180;
cp = [1500 1950 2400 3000 5250];
cs = [100 600 1000 1500 2500];
rho = [1500 2100 2200 2400 2700];

color_bar = ['c' 'g' 'b' 'r' 'm'];

for(m=1:length(cp))
    for(n=1:length(theta_inc))

        kp = omega/cp(m);
        ks = omega/cs(m);
        theta_p = acos(k1*cos(theta_inc(n))/kp);
        theta_s = acos(k1*cos(theta_inc(n))/ks);

        zp = rho(m)*cp./sin(theta_p);
        zs = rho(m)*cs./sin(theta_s);
        ztot = zp*(cos(2*theta_s))^2+zs*(sin(2*theta_s))^2;
        z1 = rhow*cw/sin(theta_inc(n));

        r_coeff(n) = (ztot - z1)/(ztot + z1);

    end

    BL = -20*log10(abs(r_coeff));
    plot(theta_inc*180/pi,BL,color_bar(m));
end
ylabel('Backscattering Strength (dB)')
xlabel('Grazing Angle (deg)')
title('Liquid-Solid Bottom Loss Model')
legend('Clay','Moraine','Chalk','Limestone','Basalt'...
    ,'Location','Best')
hold off
```

```
%% Look at TL for a given grazing angle for both models

theta_inc = [0:0.1:90]*pi/180;
theta_scat = -theta_inc;

c0 = 1500;
depth = 5000;
range = depth./sin(theta_inc);
time = range*2/cw;

TL = 20*log10(range);

% Lambert's Law -29dB

S_b_lambert = -29 + 10*log10((sin(theta_inc)).^2);

RL_lambert = -2*TL+S_b_lambert+10*log10(c0*2*pi*range/2);

figure(4)
hold on
plot(time,RL_lambert)

% NRL Model

theta_inc = [0:0.1:90]*pi/180;
theta_scat = theta_inc;

f = 3000; %frequency in Hz
rho_b = [1.4 2.0 2.4 2.7]; %bottom densities for [mud;sand;mudstone;limestone;basalt]
cp = [0.99-0.002*1i 1.2-0.005*1i 2.7-0.004*1i 3.4-0.006*1i]*c0;
cs = [0.1-0.004*1i 0.3-0.07*1i 1.5-0.003*1i 1.8-0.006*1i]*c0;
gamma_2 = [3.3 3.3 3.3 3.3];
w_2 = [0.000518 0.006957 0.004 0.01862];
h_0 = 1;

color_bar = ['g' 'r' 'c' 'm'];
legend_bar = ['Mud' 'Sand' 'Limestone' 'Basalt'];

theta_scat = theta_inc;
```

```
sigma_bi = pi;

c0 = 1500;
depth = 5000;

k_0 = 2*pi*f/c0;

for(n=1:length(cp))
    for(m=1:length(theta_inc))
        theta1 = theta_inc(m);
        theta2 = theta_scat(m);
        kp = 2*pi*f/cp(n);
        ks = 2*pi*f/cs(n);

        kq = k_0^2*cos(theta1)*cos(theta2)*cos(sigma_bi);

        p1 = sqrt(kp^2-k_0^2*(cos(theta1))^2);
        p2 = sqrt(kp^2-k_0^2*(cos(theta2))^2);
        s1 = sqrt(ks^2-k_0^2*(cos(theta1))^2);
        s2 = sqrt(ks^2-k_0^2*(cos(theta2))^2);

        a1 = rho_b(n)*k_0*sin(theta1)/p1*(1-(4*k_0^2)*...
            (cos(theta1))^2*s1*(s1-p1)/ks^4);
        a2 = rho_b(n)*k_0*sin(theta2)/p2*(1-(4*k_0^2)*...
            (cos(theta2))^2*s2*(s2-p2)/ks^4);

        b1 = rho_b(n)*k_0*sin(theta1)/p1*(1-2*k_0^2*...
            (cos(theta1))^2/ks^2);
        b2 = rho_b(n)*k_0*sin(theta2)/p2*(1-2*k_0^2*...
            (cos(theta2))^2/ks^2);

        zeta1 = k_0*sin(theta1)/(ks^2*p1)*((s1-p1)^2-kp^2);
        zeta2 = k_0*sin(theta2)/(ks^2*p2)*((s2-p2)^2-kp^2);

        Q_h = k_0*sqrt((cos(theta1))^2+(cos(theta2)^2)-2*...
            cos(theta1)*cos(theta2)*cos(sigma_bi));
        W_Qh = w_2(n)/((h_0*Q_h)^gamma_2(n));

        beta = (1/((a1+1)*(a2+1)))*(4*(1-rho_b(n))*(kq*zeta1*zeta2-k_0^2*...
            sin(theta1)*sin(theta2))+(8*rho_b(n)/ks^2)*kq^2*zeta1...
            *zeta2-4*kq*(a1-zeta1)*(a2-zeta2)+4*k_0^2*a1*a2+...
            (2*ks^2/rho_b(n))*(a1-b1)*(a2-b2)-4*kp^2*b1*b2/rho_b(n));

        sigma(m) = abs(beta/2)^2*W_Qh;
        range(m) = depth/sin(theta_inc(m));
        time(m) = range(m)*2/cw;
        TL(m) = 20*log10(range(m));
        S_b_nrl(m) = 10*log10(sigma(m));
        RL_nrl(m) = -2*TL(m)+S_b_nrl(m)+10*log10(c0*2*pi*range(m)/(2*...
            sin(theta_inc(m))));
    end

    plot(time,RL_nrl,color_bar(n))
end
ylabel('Reverberation Level in dB')
xlabel('time in sec')
title('NRL Bottom Scatter Model')
axis([0 100 -200 0])
legend('Lambert -29dB','Mud','Sand','Limestone','Basalt','Location','Best')
hold off
```

Appendix B

MATLAB Code: NRL Model and BELLHOP

```
%Bottom Reverberation Simulator
%Tom Miller
%August 2015

function bottomreverb( ARRFIL, ird, isd, rho_b, c_0, cp, cs, w_2, depth)

close all

% plot the arrivals calculated by BELLHOP
%
% usage:
% bottomreverb( filename, irr, ird, isd )
% where:
% irr = index of receiver range
% ird = index of receiver depth
% isd = index of source   depth
% rho_b = density of bottom
% c_0 = sound speed in water at the bottom
% cp = compressional sound speed in bottom as a factor of c_0
% cs = shear sound speed in bottom as a factor of c_0
% w_2 = from table
% mbp, April 2009
% read

Narrmx = 5000;

% Read the arrival time/amplitude data computed by BELLHOP
%
% usage:
%[ Arr, Pos ] = read_arrivals_asc( ARRFile, Narrmx );
%
% Arr is a structure containing all the arrivals information
% Pos is a structure containing the positions of source and receivers
%
% ARRFile is the name of the Arrivals File
% Narrmx is the maximum number of arrivals allowed
% mbp 9/96

if nargin == 1
   Narrmx = 200;
end

fid = fopen( ARRFIL, 'r');  % open the file
if ( fid == -1 )
   error( 'read_arrivals_asc: Arrivals file cannot be opened' )
end

% read the header info

freq = fscanf( fid, '%f',  1  );

Nsd  = fscanf( fid, '%i',  1  );    % number of source   depths
Nrd  = fscanf( fid, '%i',  1  );    % number of receiver depths
Nrr  = fscanf( fid, '%i',  1  );    % number of receiver ranges

Pos.s.depth   = fscanf( fid, '%f', Nsd );    % source   depths
Pos.r.depth   = fscanf( fid, '%f', Nrd );    % receiver depths
Pos.r.range   = fscanf( fid, '%f', Nrr );    % receiver ranges

% loop to read all the arrival info (delay and amplitude)
```

```
Arr.A         = zeros( Nrr, Narrmx, Nrd, Nsd );
Arr.delay     = zeros( Nrr, Narrmx, Nrd, Nsd );
Arr.SrcAngle  = zeros( Nrr, Narrmx, Nrd, Nsd );
Arr.RcvrAngle = zeros( Nrr, Narrmx, Nrd, Nsd );
Arr.NumTopBnc = zeros( Nrr, Narrmx, Nrd, Nsd );
Arr.NumBotBnc = zeros( Nrr, Narrmx, Nrd, Nsd );

iii=0;

for isd = 1:Nsd
   Narrmx2 = fscanf( fid, '%i', 1 );  % max. number of arrivals to follow
   disp( [ 'Max. number of arrivals for source index ', num2str( isd ), ' is ', num2str( Narrmx2
   for ird = 1:Nrd
      for ir = 1:Nrr
         Narr = fscanf( fid, '%i', 1 ); % number of arrivals
         Arr.Narr( ir, ird, isd ) = Narr;

         if Narr > 0   % do we have any arrivals?
            da = fscanf( fid, '%f', [ 7, Narr ] );
            Narr = min( Narr, Narrmx ); % we'll keep no more than Narrmx values
            Arr.Narr( ir, ird, isd ) = Narr;

            Arr.A(         ir, 1:Narr, ird, isd ) = da( 1, 1:Narr ) .* exp( 1i * da( 2, 1:Narr )
            Arr.delay(     ir, 1:Narr, ird, isd ) = da( 3, 1:Narr );
            Arr.SrcAngle(  ir, 1:Narr, ird, isd ) = da( 4, 1:Narr );
            Arr.RcvrAngle( ir, 1:Narr, ird, isd ) = da( 5, 1:Narr );
            Arr.NumTopBnc( ir, 1:Narr, ird, isd ) = da( 6, 1:Narr );
            Arr.NumBotBnc( ir, 1:Narr, ird, isd ) = da( 7, 1:Narr );

            A(ir,1:Narr,ird,isd) = da(1,1:Narr).*exp(1i*da(2,1:Narr));
            delay(ir,1:Narr, ird, isd ) = da( 3, 1:Narr );
            SrcAngle(ir,1:Narr, ird, isd ) = da( 4, 1:Narr );
            RcvrAngle(ir,1:Narr, ird, isd ) = da( 5, 1:Narr );
            NumTopBnc(ir,1:Narr, ird, isd ) = da( 6, 1:Narr );
            NumBotBnc(ir,1:Narr, ird, isd ) = da( 7, 1:Narr );

            if (NumBotBnc(ir,1,ird,isd)==0 | NumBotBnc(ir,1,ird,isd)==1)
               iii = iii+1;
```

```
gamma_2 = 3.3;
h_0 = 1;

t_delay(iii) = 2*delay(ir,1,ird,isd);
Amp(iii) = A(ir,1,ird,isd);

theta_inc = RcvrAngle(ir,1,ird,isd)*pi/180;
theta(iii) = theta_inc;
if theta(iii) < 0
   theta(iii) = -theta(iii);
end
theta_scat = theta_inc;
sigma_bi = pi;
theta1 = theta_inc;
theta2 = theta_scat;
k_0 = 2*pi*freq/c_0;
kp = 2*pi*freq/(cp*c_0);
ks = 2*pi*freq/(cs*c_0);

kq = k_0^2*cos(theta1)*cos(theta2)*cos(sigma_bi);

p1 = sqrt(kp^2-k_0^2*(cos(theta1))^2);
p2 = sqrt(kp^2-k_0^2*(cos(theta2))^2);
s1 = sqrt(ks^2-k_0^2*(cos(theta1))^2);
s2 = sqrt(ks^2-k_0^2*(cos(theta2))^2);

a1 = rho_b*k_0*sin(theta1)/p1*(1-(4*k_0^2)*...
    (cos(theta1))^2*s1*(s1-p1)/ks^4);
a2 = rho_b*k_0*sin(theta2)/p2*(1-(4*k_0^2)*...
    (cos(theta2))^2*s2*(s2-p2)/ks^4);

b1 = rho_b*k_0*sin(theta1)/p1*(1-2*k_0^2*...
    (cos(theta1))^2/ks^2);
b2 = rho_b*k_0*sin(theta2)/p2*(1-2*k_0^2*...
    (cos(theta2))^2/ks^2);

zeta1 = k_0*sin(theta1)/(ks^2*p1)*((s1-p1)^2-kp^2);
zeta2 = k_0*sin(theta2)/(ks^2*p2)*((s2-p2)^2-kp^2);
```

```
               Q_h = k_0*sqrt((cos(theta1))^2+(cos(theta2)^2)-2*...
                   cos(theta1)*cos(theta2)*cos(sigma_bi));
               W_Qh = w_2/((h_0*Q_h)^gamma_2);

               beta = (1/((a1+1)*(a2+1)))*(4*(1-rho_b)*(kq*zeta1*zeta2-k_0^2*...
                   sin(theta1)*sin(theta2))+(8*rho_b/ks^2)*kq^2*zeta1...
                   *zeta2-4*kq*(a1-zeta1)*(a2-zeta2)+4*k_0^2*a1*a2+...
                   (2*ks^2/rho_b)*(a1-b1)*(a2-b2)-4*kp^2*b1*b2/rho_b);

               sigma = abs(beta/2)^2*W_Qh;

               Sb_nrl = 10*log10(sigma);

               Sb(iii) = Sb_nrl;

               range = t_delay(iii)/c_0;

               TLT = -20*log10(range);

               TL(iii) = -20*log10(Amp(iii));

               RL_nrl(iii) = -2*TL(iii)+Sb_nrl+10*log10(pi*t_delay(iii)/...
                   sin(theta1));



            end
         end
      end       % next receiver range
   end      % next receiver depth
end % next source depth

figure(1)
plot(t_delay,RL_nrl)
ylabel('Reverberation Level in dB')
xlabel('time in sec')
title('NRL Bottom Scatter Model')

% figure(2)
% plot(theta*180/pi,Sb)
% ylabel('BackScatter')
% xlabel('theta_inc')
% %axis([0 100 -300 0])


fclose( fid );
```

Bibliography

[1] L.M. Brekhovskikh and Y.P. Lysanov. *Fundamentals of Ocean Acoustics*. Springer Science and Business Media New York, 3rd edition, 2003.

[2] S. Danesh. Real time active sonar simulation in a deep ocean environment. Master's thesis, Massachusetts Institute of Technology, February 2013.

[3] G. Frisk. *Ocean and Seabed Acoustics: A Theory of Wave Propagation*. Prentice Hall, 1st edition, 1994.

[4] F.B. Jensen, W.A. Kuperman, M.B. Porter, and H. Schmidt. *Computational Ocean Acoustics*. Springer Science and Business Media New York, 2nd edition, 2011.

[5] Applied Physics Laboratory. Apl-uw high-frequency ocean environmental acoustic models handbook. Technical report, University of Washington, 1994.

[6] P. Newman. *MOOS - Mission Oriented Operating Suite*. Massachusetts Institute of Technology, 2001.

[7] M. Porter and H. Bucker. Gaussian beam tracing for computing ocean acoustic fields. *JASA*, 82(4):1349–1359, 1987.

[8] M.B. Porter. *Bellhop Manual and User's Guide:PRELIMINARY DRAFT*. Heat, Light, and Sound Research, Inc., 2011.

[9] D. Wurmser R.C. Gauss, R.F. Gragg and J.M. Fialkowski. Broadband models for predicting bottom, surface, and volume scattering strengths. Technical report, Naval Research Laboratory, 2002.

[10] H. Schmidt and M. Benjamin. *2.680 - Unmanned Marine Vehicle Autonomy, Sensing, and Communications Course Notes*. Massachusetts Institute of Technology, 2015.

[11] R.J. Urick. *Principles of Underwater Sound*. McGraw-Hill, 1983.