NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

ACTIVE STEALTH IN SUBMERGED ENVIRONMENTS: A NUMERICAL INVESTIGATION

by

Dylan E. Hyde

June 2023

Thesis Advisor:	Robert L. Bassett
Co-Advisor:	Anthony Austin
Second Reader:	Lucas C. Wilcox

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE	*Form Approved OMB No. 0704-0188*

1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** June 2023	**3. REPORT TYPE AND DATES COVERED** Master's thesis

4. TITLE AND SUBTITLE
ACTIVE STEALTH IN SUBMERGED ENVIRONMENTS: A NUMERICAL INVESTIGATION

5. FUNDING NUMBERS

6. AUTHOR(S) Dylan E. Hyde

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey, CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
N/A

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release. Distribution is unlimited.

12b. DISTRIBUTION CODE
A

13. ABSTRACT (maximum 200 words)

Enemy acoustic sensors can present issues for U.S. Navy submarines that attempt to operate undetected. One potential way to address this threat is to use active noise cancellation to mask submarine noise. This thesis examines the feasibility of this idea by numerically solving a partial differential equation–constrained optimization problem to compute a masking signal for a given source signal as the signal propagates through a simulated 2D ocean environment. We examine how the location and type of source affect the strength of the cancellation. In particular, we show that it is possible to effectively cancel the sound from a source within a specific region—such as the area in proximity to an acoustic sensor—provided that the source's location and sound profile are known.

14. SUBJECT TERMS sensor, acoustics, PDE constrained optimization	**15. NUMBER OF PAGES** 49
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	**18. SECURITY CLASSIFICATION OF THIS PAGE**	**19. SECURITY CLASSIFICATION OF ABSTRACT**	**20. LIMITATION OF ABSTRACT**
Unclassified	Unclassified	Unclassified	UU

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release. Distribution is unlimited.

ACTIVE STEALTH IN SUBMERGED ENVIRONMENTS: A NUMERICAL INVESTIGATION

Dylan E. Hyde
Captain, United States Army
BS, United States Military Academy, 2014

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN APPLIED MATHEMATICS

from the

NAVAL POSTGRADUATE SCHOOL
June 2023

Approved by: Robert L. Bassett
Advisor

Anthony Austin
Co-Advisor

Lucas C. Wilcox
Second Reader

Francis X. Giraldo
Chair, Department of Applied Mathematics

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Enemy acoustic sensors can present issues for U.S. Navy submarines that attempt to operate undetected. One potential way to address this threat is to use active noise cancellation to mask submarine noise. This thesis examines the feasibility of this idea by numerically solving a partial differential equation–constrained optimization problem to compute a masking signal for a given source signal as the signal propagates through a simulated 2D ocean environment. We examine how the location and type of source affect the strength of the cancellation. In particular, we show that it is possible to effectively cancel the sound from a source within a specific region—such as the area in proximity to an acoustic sensor—provided that the source's location and sound profile are known.

THIS PAGE INTENTIONALLY LEFT BLANK

Table of Contents

THIS PAGE INTENTIONALLY LEFT BLANK

List of Figures

THIS PAGE INTENTIONALLY LEFT BLANK

List of Tables

THIS PAGE INTENTIONALLY LEFT BLANK

List of Acronyms and Abbreviations

ABC	absorbing boundary condition
dB	decibels
PDE	partial differential equation
RMS	root mean square
USN	U.S. Navy

THIS PAGE INTENTIONALLY LEFT BLANK

Acknowledgments

I would not have been able to finish this thesis without the support of my wife, Moriah Hyde. Her efforts allowed me the time required to complete this thesis, and she kept me motivated when I did not want to work.

I also need to thank my two advisors, Dr. Robert Bassett and Dr. Anthony Austin, who were always available to answer my questions and help me along the way. Their assistance was not only invaluable to my thesis but also my overall masters education.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 1:
Introduction

This thesis examines the feasibility of active underwater noise cancellation. Active underwater noise cancellation has both civilian and military relevance but is an under-researched area of acoustics. Underwater acoustics is particularly important to the Navy because of its mission and the requirement for stealth that goes along with its operations. The Chief of Naval Operation in his 2022 navigation plan specifically indicated China as a main competitor that the United States Navy (USN) will need to focus on in the next decade. One of the areas of concern is China's "rapidly increasing military capability" [1]. One of these capabilities is the construction of sophisticated sensors. A key to the USN's submarine force is the ability "to operate inside of the adversary's defensive perimeter and to deliver effects" according to the USN's Commander of Submarine Forces [2]. Sensors allow an adversary to limit that capability. Noise cancellation, and masking signals specifically, may be one way to provide that ability to operate freely despite the presence of acoustic hydrophones. The more flexibility a submarine has to operate undetected, the more effective that submarine can be.

This thesis looks at whether we can mask a known sound profile such as one associated with a submarine in a given region of the ocean. By "masking a sound profile", we mean broadcasting a signal that results in the cancellation of a different signal within the region. We refer to the sound we are trying to mask as the "source" and the mechanism that broadcasts the masking signal as the "mask." In other words, given two signals, one we control, and one we do not, is it possible to choose the one we control to minimize the acoustic energy within a specific region? We evaluate this through the lens of the military application of canceling noise that comes from a submarine, but it has applications to all sound traveling both under and over water, so this research could be applied to any active noise cancellation problem.

1.1 Research Objectives

The main objective of this research is to evaluate the feasibility of active underwater noise cancellation in an environment where the only sound being produced is coming from a known source. We aim to identify the best results that can be realistically achieved given the physical reality of how sound travels. We identify how location and type of signal of the source and mask affect our ability to cancel the source and in what situations the best results can be achieved.

The results of this research demonstrate the feasibility of active underwater noise cancellation and outline how to achieve optimal results. This process includes what sound profile is the most conducive to cancellation as well as the optimal location for the mask. The goal is to achieve a zero effect from the source noise, completely canceling it with the masking noise. In other words, we want to minimize the acoustic energy present in a given region.

1.2 Motivation

The primary motivation for this research is its potential military applications to allow freedom of movement for submarines. That is, we want to be able to cancel a specific noise produced by a submarine to the point where it cannot be picked up by an adversary's sensor. In simple terms, the goal of this research is to allow submarines to operate with less concern for signals that they are emitting being picked up by acoustic sensors because they have confidence that those signals can be actively masked.

In addition to naval applications, this research is also relevant to combating underwater noise pollution. Many underwater creatures rely on "acoustic signals for survival, communication, prey location, predator avoidance and navigational purposes" [3]. Existing noise pollution in waterways has been shown to negatively affect many species of fish to varying degrees [3]. Decreasing this noise pollution would help preserve ecosystems, and simply masking a source noise would allow for current waterborne activities to continue with little to no interruption. This is important because one of the main causes of underwater noise pollution is from vessels in shipping lanes [3]. If the assumption is made that average daily vessel flow on many of these routes is increasing, it follows that the noise pollution along these routes is also increasing. Being able to mask noise underwater could counteract the effects of any future increase in vessels traffic.

1.3 Related Literature

The literature on active noise-cancelling has looked at similar problems to ours but everything that we found has taken a different approach. There are two overarching approaches to acoustic cloaking: active and passive. Andrew Norris discusses passive acoustic cloaking in his paper *Acoustic Cloaking Theory*. His analysis showed that with specific materials the inside of an object can allow a wave to act as if the object was not present [4]. His research shows that specific materials have the property to allow sound waves to pass through and come out the other side as if no object was there. Similarly, in *Design of continuously graded elastic acoustic cloaks* it is discussed how an elastic cylinder, given specific properties will allow sound waves to pass though without reflection [5]. Acoustic diodes are another passive acoustic cloaking technique that have been researched. Acoustic diodes are objects that allow sound to pass through when traveling one direction but not when traveling the other [6].

One common approach to active noise cancelling is to utilize a phase shift. This approach uses a microphone to analyzes an incoming sound and a speaker to generate a 180 degree shift on the incoming sound wave, effectively reducing the sound level [7]. This is an effective approach but only works locally, in practice, limiting its usefulness in settings where the acoustic sensor and mask are not collocated.

A similar active wave control has been researched in the field of vibration control. Tanaka and Kikushima present the use of an active sink in order to control vibrations in a beam, in their paper [8]. They take a different approach than ours, utilizing controls on the boundary conditions to damp vibrations. Despite this difference, their work considers the same governing partial differential equation—the wave equation—as this thesis.

Our research differs from previous work on noise cancellation because we are taking an active approach over a large environment. While an active noise cancelling approach is well researched over a small domain, we could not find any research for taking an active noise cancelling over a large domain.

1.4 Research Design

In order to establish the feasibility of active underwater noise cancellation, we solve an optimization problem constrained by a partial differential equation (PDE). These are of interest to mathematicians because PDEs govern much of the natural environment. We use a PDE constrained optimization model in order to obtain a realistic solution to the problem. Utilizing the wave equation with known speed of underwater sound propagation, we model a known source and optimize for the values that will minimize the disturbance of the source signal on a designated region in our domain. The wave equation governs the physical constraints of sound in water.

In order to solve the wave equation, we use the finite element method, which entails breaking up our domain into subdomains called elements, and approximating the solution to the equation on each element. Once we have discretized our problem in both space and time, we optimize for the value of the masking signal that results in the minimum amount of energy over our chosen region.

The rest of this thesis will be broken down into two additional chapters. Chapter 2 outlines the specific mathematical formulation of our active noise cancellation problem. To do this, we take our problem and discretize it in both space and time. We then take the discretized problem and solve our optimization problem utilizing that discretized framework. Chapter 3 presents the results from our research broken down by type of source and then the location of the source. Finally, we provide our conclusions.

CHAPTER 2:
Mathematical Background

2.1 Problem Statement

First, we need a model for the underwater propagation of sound. We use the linear undamped wave equation,

$$\frac{\partial^2 u}{\partial t^2} = c^2 \Delta u + q(t), \tag{2.1}$$

where u represents the acoustic pressure, c is the speed of sound underwater, and $q(t)$ models the combined sound generated by the source and the mask. The symbol Δ denotes the Laplacian operator.

We use equation (2.1) to simulate the sound emitted by the source and mask within a region of the ocean, modeled here as a square domain $\Omega \subset \mathbb{R}^2$, for a total time of T seconds. Our aim is to select the mask to minimize the amount of sound present in a subregion, $\mathcal{S} \subset \mathbb{R}^2$. Figure 2.1 is a visual representation of Ω with $\mathcal{S}$ represented by the box at the bottom of the region. In terms of our application to submarine stealth, $\mathcal{S}$ is the region around an enemy sensor in which we want to cancel the sound. To model $\mathcal{S}$, we use its characteristic function $\alpha(\vec{x})$, which takes the value 1 if $\vec{x}$ is in $\mathcal{S}$ and 0 otherwise:

$$\alpha(\vec{x}) = \begin{cases} 1 & \text{if } \vec{x} \in \mathcal{S} \\ 0 & \text{otherwise.} \end{cases}$$

We measure the magnitude of sound in $\mathcal{S}$ with the squared 2-norm of u on $\mathcal{S} \times [0, T]$,

$$\|u\|_{2,\mathcal{S}}^2 = \int_0^T \int_\Omega \alpha(\vec{x})\, u(\vec{x}, t)^2 \, dV \, dt. \tag{2.2}$$

Physically, $\|u\|_{2,\mathcal{S}}^2$ is the total amount of acoustic energy imparted upon $\mathcal{S}$ by the wave u within T seconds. The combined source and mask signal $q(t)$ has the form

$$q(t) = g(t)\delta_\varepsilon\big(\vec{x} - \vec{s}(t)\big) + f(t)\delta_\varepsilon\big(\vec{x} - \vec{d}(t)\big), \tag{2.3}$$

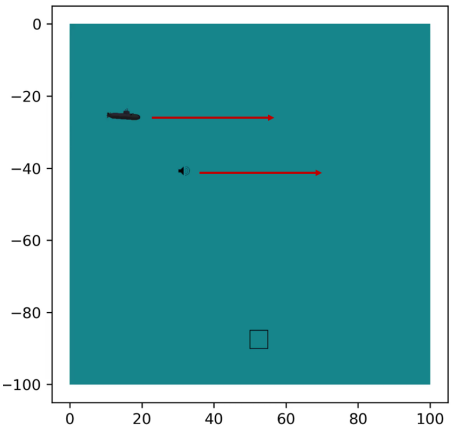

Figure 2.1. Visual representation of the domain of our problem. The image of the submarine represents our source, while the image of the speaker represents our mask. The arrows represent the direction of movement of the source and mask, and the small box at the bottom of the domain is the region in which we seek to cancel the noise.

where $g(t)$ represents the known source signal and $f(t)$ is the mask signal, which we select via optimization. The symbol δ represents the Dirac delta function: we model the source and mask as point sources located at positions $\vec{s}(t)$ and $\vec{d}(t)$ respectively, which may vary with time. The Dirac delta is an idealization; in our simulation, we approximate it by [9, Section 2.5]

$$\delta_\varepsilon(\vec{x}) = \delta_\varepsilon(x, y) = \frac{\varepsilon^2}{\pi^2(x^2 + \varepsilon^2)(y^2 + \varepsilon^2)}, \tag{2.4}$$

where ε is a small positive number.

The problem of finding the most effective way to cancel our source term amounts to selecting f to make u as small as possible. That is, we aim to solve the following optimization problem:

$$\min_f \mathcal{J}(u; f) := \int_0^T \int_\Omega \alpha(\vec{x})\, u(\vec{x}, t)^2 \, dV \, dt \tag{2.5}$$
$$\text{s.t.} \quad \frac{\partial^2 u}{\partial t^2} = c^2 \Delta u + g(t)\delta_\varepsilon\big(\vec{x} - \vec{s}(t)\big) + f(t)\delta_\varepsilon\big(\vec{x} - \vec{d}(t)\big).$$

We assume that there is initially no sound in our domain. Consequently, u satisfies zero

initial conditions, and the only sound that exists in our domain is produced by either the source or the mask.

2.2 Spatial Discretization

We use a Galerkin method to discretize equation (2.1) [10], [11]. This method requires us to convert equation (2.1) into variational form, which we then solve over a mesh of finite elements.

2.2.1 Variational Form

To put equation (2.1) into variational form, we multiply both sides by a test function v and integrate over Ω, giving

$$\int_\Omega \frac{\partial^2 u}{\partial t^2}\, v\, dV \;=\; \int_\Omega c^2\, \Delta u\, v\, dV + \int_\Omega q(\vec{x}, t)\, v\, dV. \tag{2.6}$$

We can simplify equation (2.6) using Green's identity,

$$\int_\Omega (\Delta u)\; v\, dV = -\int_\Omega \nabla u \cdot \nabla v\, dV + \int_{\partial\Omega} v\, (\nabla u \cdot \vec{n})\; dS, \tag{2.7}$$

where $\vec{n}$ is the outward pointing normal vector to $\partial\Omega$. This yields

$$\frac{d^2}{dt^2} \int_\Omega u\, v\, dV \;= -c^2 \int_\Omega \nabla u \cdot \nabla v\, dV + c^2 \int_{\partial\Omega} (\nabla u \cdot \vec{n})\, v\, dS + \int_\Omega q(\vec{x}, t)\, v\, dV. \tag{2.8}$$

We must select u so that equation (2.8) is satisfied for all test functions v.

2.2.2 Boundary Condition

All PDEs require a rule to be defined for how the solution behaves at the boundaries of the domain. This provides the required information to accurately model the physical realities at the boundaries. In our case a simple Dirichlet or Neumann condition has some potential benefits, primarily that they both zero out the $\int_{\partial\Omega} (\nabla u \cdot \vec{n})\, v\, dS$ term; however, the reflections caused by these boundary conditions are not realistic for an environment as large as the ocean. As a work-around, the simulation can be run over a large environment and

stopped before any reflected waves reach the region of interest, but increasing the size of the environment is computationally very expensive.

Instead, we utilize an *absorbing boundary condition* (ABC). An ABC is a condition that directs wave energy that reaches the boundaries of a domain to leave the domain instead of reflecting back inwards. The specific ABC we use in our simulation is

$$\frac{\partial u}{\partial t} + c\nabla u \cdot \vec{n} = 0 \text{ on } \partial\Omega. \tag{2.9}$$

This is a first-order approximation to the exact absorbing boundary condition; it demands that all waves approaching the boundary parallel to the normal vector will pass through the boundary [12], [13]. Despite this not being a perfect ABC, through multiple experiments we observe that it was good enough to prevent any significant reflections. We never test our source close to the boundary as this may cause some issues.

By rearranging equation (2.9) to $\nabla u \cdot \vec{n} = -\frac{1}{c}\frac{\partial u}{\partial t}$, we can apply this to the integral over $\partial\Omega$ in equation (2.8), which yields

$$\frac{d^2}{dt^2}\int_\Omega u\, v\, dV = -c^2 \int_\Omega \nabla u \cdot \nabla v\, dV - c\frac{d}{dt}\int_{\partial\Omega} u\, v\, dS + \int_\Omega q(x,t)\, v\, dV. \tag{2.10}$$

Equation (2.10) defines the variational form of equation (2.1) with the ABC (2.9).

2.2.3 Finite Elements

Now that we have put equation (2.1) into variational form, we can discretize in space. In order to do this we use a finite element method. As shown in Figure 2.2 we utilize a triangular mesh that was generated utilizing the Gmsh software [14]. We refer to each one of the triangles in the mesh as an element and each point of the triangle as a node. There are a total of n nodes in our mesh.

We use this mesh to approximate functions on Ω by piecewise-linear polynomials that are continuous across elements. We represent these piecewise polynomials in the Lagrange basis $\phi_1, ..., \phi_n$, where ϕ_i equals 1 at the i^{th} node and 0 at all others. This same basis will be used for both our test and trial spaces, making this a Galerkin method. We expand our test

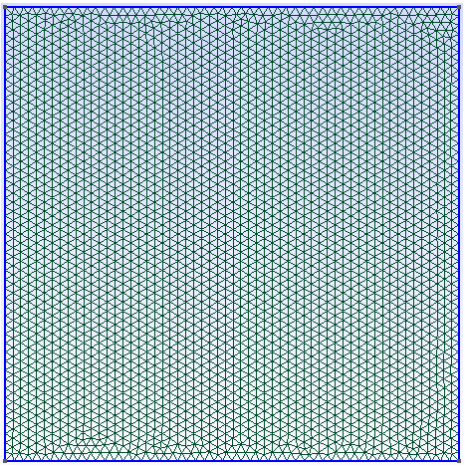

Figure 2.2. A coarse visual representation of the triangular mesh that was utilized for the finite element method. The actual mesh utilized was much finer. See Chapter 3 for the specifics.

and trial functions in this basis as follows, where u_i and v_i are the expansion coefficients [10]:

$$u(\vec{x},t) \approx \sum_{i=1}^{n} u_i(t)\phi_i(\vec{x}), \qquad v(\vec{x}) \approx \sum_{i=1}^{n} v_i\phi_i(\vec{x}), \qquad q(\vec{x},t) \approx \sum_{i=1}^{n} q_i(t)\phi_i(\vec{x}).$$

Next, we form the mass, surface, and stiffness matrices $\mathbf{M}$, $\mathbf{S}$, and $\mathbf{K}$. These are defined as follows:

$$\mathbf{M}_{ij} = \int_{\Omega} \phi_i\,\phi_j\,dV, \qquad \mathbf{S}_{ij} = \int_{\delta\Omega} \phi_i\,\phi_j\,dS, \qquad \mathbf{K}_{ij} = \int_{\Omega} \nabla\phi_i^{\top}\,\nabla\phi_j\,dV.$$

Since the functions involved in these integrals are simple piecewise linear functions, they can be evaluated exactly.

We take the coefficients of the expansions and combine them into the following column

vectors:

$$\mathbf{u} = \begin{bmatrix} u_1(t) \\ u_2(t) \\ \vdots \\ u_n(t) \end{bmatrix}, \qquad \mathbf{v} = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix}, \qquad \mathbf{q} = \begin{bmatrix} q_1(t) \\ q_2(t) \\ \vdots \\ q_n(t) \end{bmatrix}. \tag{2.11}$$

We further decompose $\mathbf{q}$ into its components due to the source and mask as follows. First, we expand the (approximate) Dirac functions in our finite element basis:

$$\delta_\varepsilon\big(\vec{x} - \vec{s}(t)\big) \approx \sum_{i=1}^{n} \delta_i^{\mathrm{s}}(t)\phi_i(\vec{x}), \qquad \delta_\varepsilon\big(\vec{x} - \vec{d}(t)\big) \approx \sum_{i=1}^{n} \delta_i^{\mathrm{d}}(t)\phi_i(\vec{x}).$$

Next, we define

$$\boldsymbol{\delta}^{\mathbf{s}} = \begin{bmatrix} \delta_1^{\mathrm{s}}(t) \\ \delta_2^{\mathrm{s}}(t) \\ \vdots \\ \delta_n^{\mathrm{s}}(t) \end{bmatrix}, \qquad \boldsymbol{\delta}^{\mathbf{d}} = \begin{bmatrix} \delta_1^{\mathrm{d}}(t) \\ \delta_2^{\mathrm{d}}(t) \\ \vdots \\ \delta_n^{\mathrm{d}}(t) \end{bmatrix},$$

giving

$$\mathbf{q} = g(t)\boldsymbol{\delta}^{\mathbf{s}} + f(t)\boldsymbol{\delta}^{\mathbf{d}}. \tag{2.12}$$

Inserting these matrices into the variational form (2.10), we obtain

$$\frac{d^2}{dt^2}\mathbf{v}^\top\mathbf{M}\mathbf{u} = -c^2\,\mathbf{v}^\top\mathbf{K}\mathbf{u} - c\frac{d}{dt}\mathbf{v}^\top\mathbf{S}\mathbf{u} + \mathbf{v}^\top\mathbf{M}\mathbf{q}. \tag{2.13}$$

Since equation (2.13) must hold for all $\mathbf{v}$, it follows that

$$\frac{d^2}{dt^2}\mathbf{M}\mathbf{u} = -c^2\,\mathbf{K}\mathbf{u} - c\frac{d}{dt}\mathbf{S}\mathbf{u} + \mathbf{M}\mathbf{q}. \tag{2.14}$$

Equation (2.14) is (2.1) discretized in space in matrix form.

2.3 Time Discretization

Now that we have discretized our equation in space we can discretize it in time. We divide the time interval $[0, T]$ into K time steps of size Δt and define

$$t_k = k(\Delta t), \qquad u^k = u(t_k).$$

Consider a general second-order ordinary differential equation,

$$\frac{d^2u}{dt^2} = f(u, t). \tag{2.15}$$

According to the central difference method [15, Chapter 1],

$$\frac{d^2u}{dt^2}\Big|_{t=t_k} \approx \frac{u^{k+1} - 2u^k + u^{k-1}}{(\Delta t)^2} \tag{2.16}$$

and

$$\frac{du}{dt}\Big|_{t=t_k} \approx \frac{u^{k+1} - u^{k-1}}{2(\Delta t)}.$$

We consider three different time-stepping schemes for equation (2.15): forward Euler, backward Euler, and leapfrog [16, Chapter 3]. These schemes all use equation (2.16) to approximate the second derivative of u; they differ according to the time at which they evaluate the right-hand side. Forward Euler evaluates the right-hand side at time step $k - 1$,

$$\frac{u^{k+1} - 2u^k + u^{k-1}}{(\Delta t)^2} = f(u^{k-1}, t_{k-1}). \tag{2.17}$$

Backward Euler evaluates the right-hand side at time step $k + 1$,

$$\frac{u^{k+1} - 2u^k + u^{k-1}}{(\Delta t)^2} = f(u^{k+1}, t_{k+1}). \tag{2.18}$$

Leapfrog evaluates the right-hand side at time step k,

$$\frac{u^{k+1} - 2u^k + u^{k-1}}{(\Delta t)^2} = f(u^k, t_k). \tag{2.19}$$

The effectiveness of each of these methods depends on the equation that is being solved. For our problem, forward Euler is unstable, and backward Euler dissipates energy, which is physically unrealistic for our undamped wave equation. This leaves the leapfrog method which is what we use. Unlike backward Euler, the leapfrog method is energy-preserving; moreover, it is second-order accurate, while both Euler methods are only first-order accurate [16, Chapter 4].

Returning to the problem of discretizing equation (2.14) in time, we define,

$$\mathbf{u}^{k+1} = \begin{bmatrix} u_1(t_{k+1}) \\ u_2(t_{k+1}) \\ \vdots \\ u_n(t_{k+1}) \end{bmatrix}, \qquad \mathbf{u}^{k} = \begin{bmatrix} u_1(t_{k}) \\ u_2(t_{k}) \\ \vdots \\ u_n(t_{k}) \end{bmatrix}, \qquad \mathbf{u}^{k-1} = \begin{bmatrix} u_1(t_{k-1}) \\ u_2(t_{k-1}) \\ \vdots \\ u_n(t_{k-1}) \end{bmatrix}.$$

We use these to discretize equation (2.14) using the leapfrog form found in equation (2.19):

$$\mathbf{M}\mathbf{u}^{k+1} - 2\mathbf{M}\mathbf{u}^{k} + \mathbf{M}\mathbf{u}^{k-1} = -c^2 (\Delta t)^2 \mathbf{K}\mathbf{u}^{k} - \frac{c\,(\Delta t)}{2}\mathbf{S}\mathbf{u}^{k+1} + \frac{c\,(\Delta t)}{2}\mathbf{S}\mathbf{u}^{k-1} + (\Delta t)^2 \mathbf{M}\mathbf{q}. \quad (2.20)$$

Rearranging equation (2.20), we obtain the following equation for the solution at t_{k+1}, given the solution at the prior steps t_k and t_{k-1}:

$$\left(\mathbf{M} + \frac{c\,(\Delta t)}{2}\mathbf{S}\right)\mathbf{u}^{k+1} = \left(2\mathbf{M} - c^2 (\Delta t)^2 \mathbf{K}\right)\mathbf{u}^{k} + \left(\frac{c\,(\Delta t)}{2}\mathbf{S} - \mathbf{M}\right)\mathbf{u}^{k-1} + (\Delta t)^2 \mathbf{M}\mathbf{q}. \quad (2.21)$$

Finally, we expand $\mathbf{q}$ back out to give the final form of the variational equation. In order to do this we need to evaluate $\boldsymbol{\delta}^{\mathbf{s}}$ and $\boldsymbol{\delta}^{\mathbf{d}}$ at each time step. The notation we will use for this is $\boldsymbol{\delta}^{\mathbf{s},k}$ which represents $\boldsymbol{\delta}^{\mathbf{s}}$ at time step k. Similarly, we will use $\boldsymbol{\delta}^{\mathbf{d},k}$, which represents $\boldsymbol{\delta}^{\mathbf{d}}$ at time step k This gives us the following equation:

$$\left(\mathbf{M} + \frac{c\,(\Delta t)}{2}\mathbf{S}\right)\mathbf{u}^{k+1} = \left(2\mathbf{M} - c^2 (\Delta t)^2 \mathbf{K}\right)\mathbf{u}^{k} + \left(\frac{c\,(\Delta t)}{2}\mathbf{S} - \mathbf{M}\right)\mathbf{u}^{k-1} + (\Delta t)^2 g(t_k)\mathbf{M}\,\boldsymbol{\delta}^{\mathbf{s},k} + (\Delta t)^2 f(t_k)\mathbf{M}\,\boldsymbol{\delta}^{\mathbf{d},k}. \quad (2.22)$$

Equation (2.22) is the matrix form of (2.1) discretized in both space and time.

2.4 Discretization of the Optimization Problem

Now that we have discretized our continuous problem, we discuss how to find the value for our mask that gives us the optimal minimum value of our objective function.

2.4.1 Objective Function

In Section 2.1 we introduced our objective function $\mathcal{J}(u; f)$, defined in equation (2.5). We can approximate the integral that arises in the definition of this function using our numeric solution to equation (2.1) obtained by solving equation (2.22). For the integration in space, we use the same integration rule that we used for integrating over our finite element mesh. For integration in time we use the trapezoid rule. To describe this, we abuse notation and redefine $\mathbf{u}$ according to

$$\mathbf{u} = \begin{pmatrix} \mathbf{u}^0 \\ \mathbf{u}^1 \\ \vdots \\ \mathbf{u}^K \end{pmatrix}.$$

Additionally, we define an "α-weighted mass matrix" $\hat{\mathbf{M}}$ by

$$\hat{\mathbf{M}}_{ij} = \int_\Omega \alpha \, \phi_i \, \phi_j \, dV.$$

The trapezoid rule approximation to $\mathcal{J}(u; f)$ may be written as,

$$J(\mathbf{u}) = \Delta t \left(\frac{(\mathbf{u}^0)^\top \hat{\mathbf{M}} \mathbf{u}^0}{2} + \sum_{k=1}^{K-1} (\mathbf{u}^k)^\top \hat{\mathbf{M}} \mathbf{u}^k + \frac{(\mathbf{u}^K)^\top \hat{\mathbf{M}} \mathbf{u}^K}{2} \right). \tag{2.23}$$

In order to simplify this equation, we define

$$\mathbf{P} = (\Delta t) \begin{pmatrix} \frac{\hat{\mathbf{M}}}{2} & 0 & \dots & 0 & 0 \\ 0 & \hat{\mathbf{M}} & & & 0 \\ \vdots & & \ddots & & \vdots \\ 0 & & & \hat{\mathbf{M}} & 0 \\ 0 & 0 & \dots & 0 & \frac{\hat{\mathbf{M}}}{2} \end{pmatrix}. \tag{2.24}$$

Our discretized objective is just the quadratic form generated by P:

$$J(\mathbf{u}) = \mathbf{u}^\top \mathbf{P}\mathbf{u}. \tag{2.25}$$

2.4.2 Constraints

Our goal is to minimize $J(\mathbf{u})$ as defined by equation (2.25), but this must be done within the constraints of the wave equation. These constraints are defined by the $K-1$ equations (2.22). We can condense these equations into a single equation by utilizing matrices. Let

$$\mathbf{A} = \begin{pmatrix} \mathbf{I} & 0 & 0 & 0 & \cdots & 0 \\ 0 & \mathbf{I} & 0 & 0 & \cdots & 0 \\ \mathbf{A}^- & \mathbf{A}^0 & \mathbf{A}^+ & 0 & \cdots & 0 \\ 0 & \mathbf{A}^- & \mathbf{A}^0 & \mathbf{A}^+ & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathbf{A}^- & \mathbf{A}^0 & \mathbf{A}^+ \end{pmatrix}, \tag{2.26}$$

where $\mathbf{I}$ denotes the identity matrix and

$$\mathbf{A}^- = \mathbf{M} - \frac{c(\Delta t)}{2}\mathbf{S}, \qquad \mathbf{A}^0 = c^2(\Delta t)^2\mathbf{K} - 2\mathbf{M}, \qquad \mathbf{A}^+ = \mathbf{M} + \frac{c(\Delta t)}{2}\mathbf{S}.$$

Further, let

$$\mathbf{B} = \begin{pmatrix} 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & (\Delta t)^2\mathbf{M}\boldsymbol{\delta}^{\mathbf{d},2} & 0 & \cdots & 0 \\ 0 & 0 & 0 & (\Delta t)^2\mathbf{M}\boldsymbol{\delta}^{\mathbf{d},3} & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & (\Delta t)^2\mathbf{M}\boldsymbol{\delta}^{\mathbf{d},K-1} \end{pmatrix} \tag{2.27}$$

and

$$\mathbf{C} = \begin{pmatrix} 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & (\Delta t)^2 \mathbf{M}\boldsymbol{\delta}^{\mathbf{s},2} & 0 & \cdots & 0 \\ 0 & 0 & 0 & (\Delta t)^2 \mathbf{M}\boldsymbol{\delta}^{\mathbf{s},3} & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & (\Delta t)^2 \mathbf{M}\boldsymbol{\delta}^{\mathbf{s},K-1} \end{pmatrix}. \tag{2.28}$$

The matrix $\mathbf{A}$ has dimensions Kn x Kn, while $\mathbf{B}$ and $\mathbf{C}$ both have dimensions Kn x K. The first two rows correspond to the initial conditions, which in our case are zero. Finally, we need to define our source and mask signals as vectors:

$$\mathbf{f} = \begin{pmatrix} f(t_1) \\ f(t_2) \\ \vdots \\ f(t_K) \end{pmatrix}, \qquad \mathbf{g} = \begin{pmatrix} g(t_1) \\ g(t_2) \\ \vdots \\ g(t_K) \end{pmatrix}.$$

Using these matrices and vectors we can write the constraint (2.22) to our optimization problem as

$$\mathbf{Au} = \mathbf{Bf} + \mathbf{Cg}. \tag{2.29}$$

Combining our objective function and this constraint now gives us our fully discretized optimization problem:

$$\min_{\mathbf{f}} J(\mathbf{u}) := \mathbf{u}^\top \mathbf{Pu} \tag{2.30}$$

$$\text{s.t.} \quad \mathbf{Au} = \mathbf{Bf} + \mathbf{Cg}. \tag{2.31}$$

Note that due to our zero initial conditions we can eliminate the variables $f(t_1)$ and $f(t_2)$, if we like.

2.4.3 Gradient Computation

In order to solve this problem for the minimum value we used an L-BFGS solver [17], [18, Section 7.2]. This is a quasi-Newton method that uses information from the gradient of the objective to approximate the Hessian. From there it uses this approximate Hessian to step closer to the minimum value. In order to use this method we need to find the gradient of

objective function (2.30) with respect to $\mathbf{f}$. We find the gradient by utilizing the reduced gradient method [19, Chapter 12]. To do this we solve our constraint function (2.31) for $\mathbf{u}$:

$$\mathbf{u} = \mathbf{A}^{-1}(\mathbf{B}\mathbf{f} + \mathbf{C}\mathbf{g}). \tag{2.32}$$

This new equation expresses $\mathbf{u}$ as a function of $\mathbf{f}$. Next, we take our objective function $J(\mathbf{u}) = \mathbf{u}^\top \mathbf{P}\mathbf{u}$ and plug in $\mathbf{u} = \mathbf{A}^{-1}(\mathbf{B}\mathbf{f} + \mathbf{C}\mathbf{g})$ to get the reduced functional $\hat{J}$,

$$\hat{J}(\mathbf{f}) = \left(\mathbf{A}^{-1}(\mathbf{B}\mathbf{f} + \mathbf{C}\mathbf{g})\right)^\top \mathbf{P}\mathbf{A}^{-1}(\mathbf{B}\mathbf{f} + \mathbf{C}\mathbf{g}). \tag{2.33}$$

We can now take the gradient of $\hat{J}$ in the direction of $\mathbf{f}$ by differentiating equation (2.33):

$$\nabla \hat{J}_{\mathbf{f}} = 2\mathbf{B}^\top \mathbf{A}^{-\top} \mathbf{P}\mathbf{A}^{-1}(\mathbf{B}\mathbf{f} + \mathbf{C}\mathbf{g}). \tag{2.34}$$

Now that we have the gradient we have everything required to solve our problem. Note that $\mathbf{A}^{-1}(\mathbf{B}\mathbf{f} + \mathbf{C}\mathbf{g})$ is found during the forward solve of our constraint equation, so the only significant additional work required to evaluate the gradient is a linear solve with $\mathbf{A}^\top$. This can be done by time stepping in the same way as the forward solve due to the block triangular structure of the $\mathbf{A}$ matrix. The only difference is that since $\mathbf{A}^\top$ is block upper triangular instead of lower triangular, the steps move backward in time.

CHAPTER 3:
Results

3.1 Implementation

Across all experiments, some implementation pieces did not change. As mentioned in Chapter 2, we use the Gmsh software package to generate a finite element mesh for Ω consisting of 23,270 triangular elements inside a 100m by 100m square, where the y-axis represents the depth and the x-axis represents the distance. We use the FEniCS Python package [20] to process the mesh and assemble the finite element matrices. When we optimize over a subregion of the domain, our mesh does not align with the subregion.

For the wave equation (2.1), we must specify the speed of sound underwater, which in principle varies with pressure, temperature, and salinity [21]. At the depth at which submarines generally operate, the variation in the speed is extremely small. Therefore, for simplicity, we use a constant wave speed of 1525 m/s [7, Chapter 5]. For our Dirac approximation (2.4), we set $\varepsilon = 0.1$. Our experiments are run over the time interval $[0, 1]$ seconds, with 6,000 time steps. For our optimization solver, we utilize the SciPy Python package [22]. The tolerance for our optimize is 1e−6, and the maximum number of iterations is 100.

In order to determine the effectiveness of our masking term, we define the reduction factor as the objective value without a mask divided by the objective value with a mask. In chapter 2 we discuss how the objective value is the total acoustic energy, therefore, the reduction factor can be interpreted as the decrease of total acoustic energy from one experiment to another.

We also report the *root mean square* (RMS) value of u, which is equal to

$$\|u\|_{\text{RMS}} = \sqrt{\int_0^T \int_\Omega \frac{1}{T \times \text{vol}\,(\mathcal{S})}\, \alpha(\vec{x})\, u(\vec{x}, t)^2 \, dV \, dt}.$$

This gives us the average acoustic pressure at any point in $\mathcal{S}$ at a specific time. The reduction factor for this term can be measured in *decibels* (dB).

3.2 Optimization by Type of Source

For this thesis, we examine two types of sources: a constant source and a sinusoidal source. For the constant source, we set $g(t) = 1$, and for the sinusoidal source we set $g(t) = 1000 \sin(200\pi t)$. The results from our experiments are listed in Table 3.1 for the constant source and Table 3.2 for the sinusoidal source.

3.2.1 Stationary Constant Source over Entire Region

At first, this research examined cancelling a stationary constant source over the entire domain, Ω. Our source is located at $(20, -25)$, and our mask is located at $(30, -45)$. By doing this we successfully cancel this source over the entire region, with a reduction factor of 3.7×10^2.

3.2.2 Stationary Constant Source over Subregion

For this research, it is not necessary to cancel the source over the entire region. In order to mask the source from a sensor, it is only required to cancel the source in the subregion around the sensor. To do this, we cancel over the square subregion with the lower left corner at $(50, -90)$ and the upper right corner at $(55, -85)$. This was more successful with a reduction factor of 1.4×10^3.

3.2.3 Moving Constant Source over Subregion

Finally, we look at a moving constant source with a stationary mask and a moving constant source with a moving mask. For both of these experiments, the source and mask start at the same location as in the stationary experiments and move in the positive x direction at a rate of 10 m/s. These tests have a reduction factor of 1.4×10^3 and 8.2×10^2, respectively.

3.2.4 Stationary Sinusoidal Source over Entire Region

Most real world activities do not emit a constant source. For the next part of our experiment, we test the effectiveness of cancelling a sinusoidal source. For all of our tests with sinusoidal source terms, the source began at location $(20, -25)$, and our mask began at location $(30, -45)$, similar to the constant source. Unlike the constant source with a sinusoidal source, we were not able to cancel over the entire region, attaining a reduction factor of

Table 3.1. Objective values with constant source

Source	Mask	Min.Region	Objective	Reduction	RMS (u)	dB
Stationary	None	Entire Region	3.9×10^{1}	N/A	6.2×10^{-2}	N/A
Stationary	Stationary	Entire Region	1.1×10^{-1}	3.7×10^{2}	3.2×10^{-3}	1.3×10^{1}
Stationary	None	10×10 square	8.1×10^{-2}	N/A	2.8×10^{-2}	N/A
Stationary	Stationary	10×10 square	5.8×10^{-5}	1.4×10^{3}	7.6×10^{-4}	1.6×10^{1}
Moving	None	10×10 square	6.0×10^{-2}	N/A	2.4×10^{-2}	N/A
Moving	Stationary	10×10 square	4.3×10^{-5}	1.4×10^{3}	6.5×10^{-4}	1.6×10^{1}
Moving	Moving	10×10 square	7.3×10^{-5}	8.2×10^{2}	8.5×10^{-4}	1.5×10^{1}

only 1.1×10^{0}. Fortunately, this is not an essential task because to mask the source from a sensor, we are only required to mask it over a subregion.

3.2.5 Stationary Sinusoidal Source over Subregion

For our sinusoidal source, we use the same subregion as our constant source, a square with the lower left corner at $(50, -90)$ and the upper right corner at $(55, -85)$. Unlike attempting to cancel over the entire region, canceling the sinusoidal source over this subregion is possible, with a reduction factor of 8.6×10^{2} for a stationary source and mask.

3.2.6 Moving Sinusoidal Source over Subregion

Similar to the moving constant source, we examine two moving sinusoidal source experiments, one with a moving source and stationary mask; the other with a moving source and mask. For both of these experiments any moving terms move in the positive x direction at a rate of 10 m/s. These two experiments result in reduction factors of 1.3×10^{2} and 3.0×10^{2}, respectively.

Figures 3.1 and 3.2 show what a simulation of a stationary constant source looks like without a mask and then with a mask. The moving simulations look almost identical to the stationary simulations when examining still images; therefore, they are not pictured. Figure 3.1 shows that by 0.18 seconds, the acoustic pressure has emanated from the source to the entire domain, while we see in Figure 3.2, when we add the mask, the acoustic pressure is contained to the general area of the source. Figures 3.3 and 3.4 show what a simulation of a

Table 3.2. Objective values with sinusoidal source

Source	Mask	Min. Region	Objective	Reduction	RMS (u)	dB
Stationary	None	Entire Region	1.0×10^{4}	N/A	1.0×10^{0}	N/A
Stationary	Stationary	Entire Region	9.8×10^{3}	1.1×10^{0}	9.9×10^{-1}	1.1×10^{-1}
Stationary	None	10×10 square	1.3×10^{1}	N/A	3.6×10^{-1}	N/A
Stationary	Stationary	10×10 square	1.5×10^{-2}	8.6×10^{2}	1.2×10^{-2}	1.5×10^{1}
Moving	None	10×10 square	8.2×10^{0}	N/A	2.9×10^{-1}	N/A
Moving	Stationary	10×10 square	6.2×10^{-2}	1.3×10^{2}	2.5×10^{-2}	1.1×10^{1}
Moving	Moving	10×10 square	2.7×10^{-2}	3.0×10^{2}	1.7×10^{-1}	1.2×10^{1}

stationary sinusoidal source looks like with and without a mask. Figure 3.3 shows us how a sinusoidal source differs considerably from a constant source. Figure 3.4 shows how we are able to cancel the acoustic pressure within our region of interest but not in the entire domain.

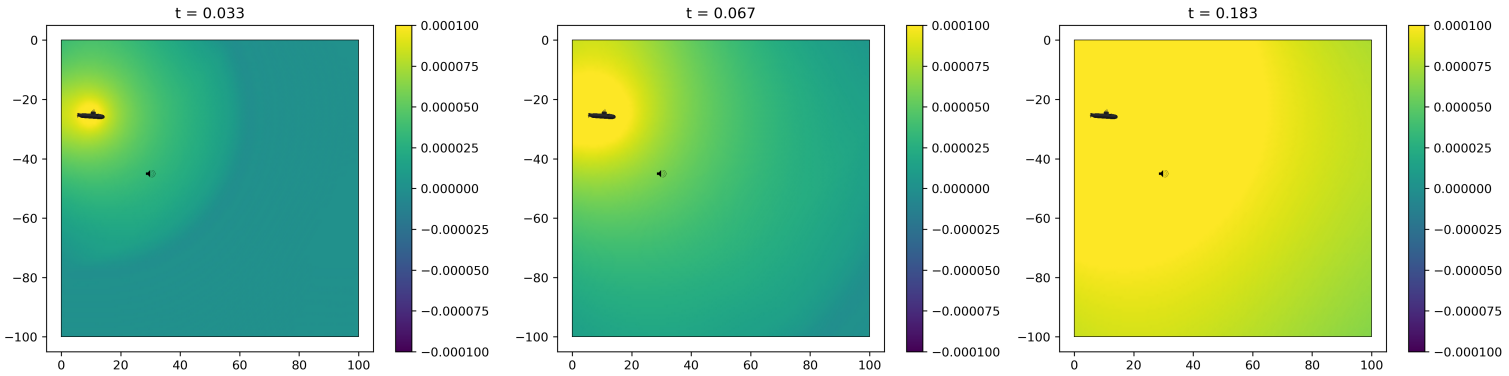

Figure 3.1. Constant source no mask

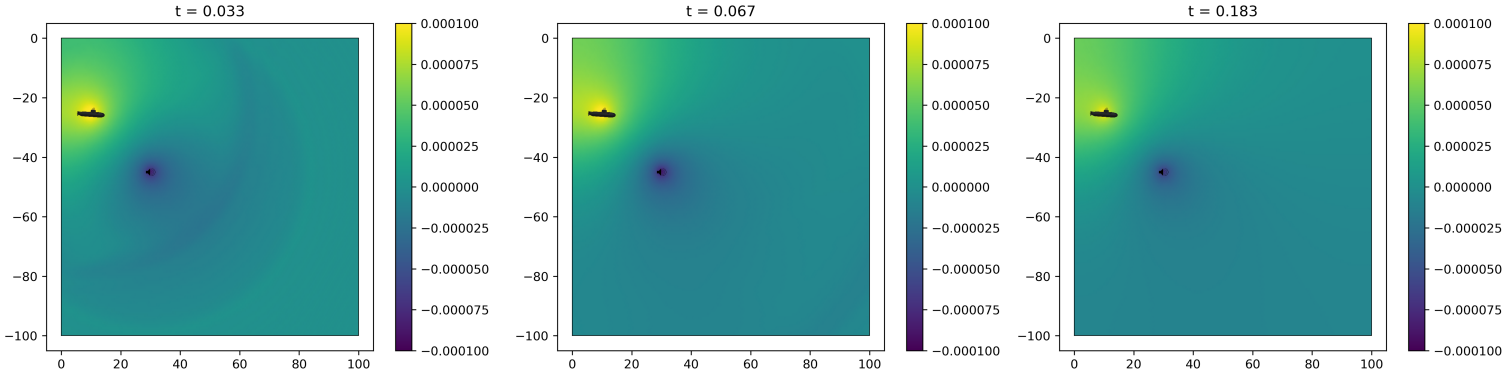

Figure 3.2. Constant source with mask

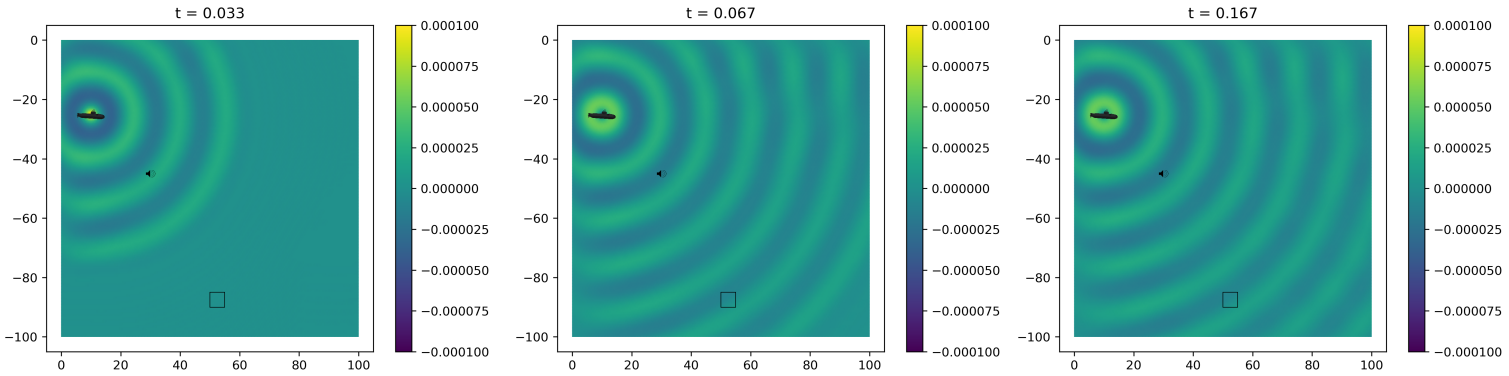

Figure 3.3. Sine source no mask

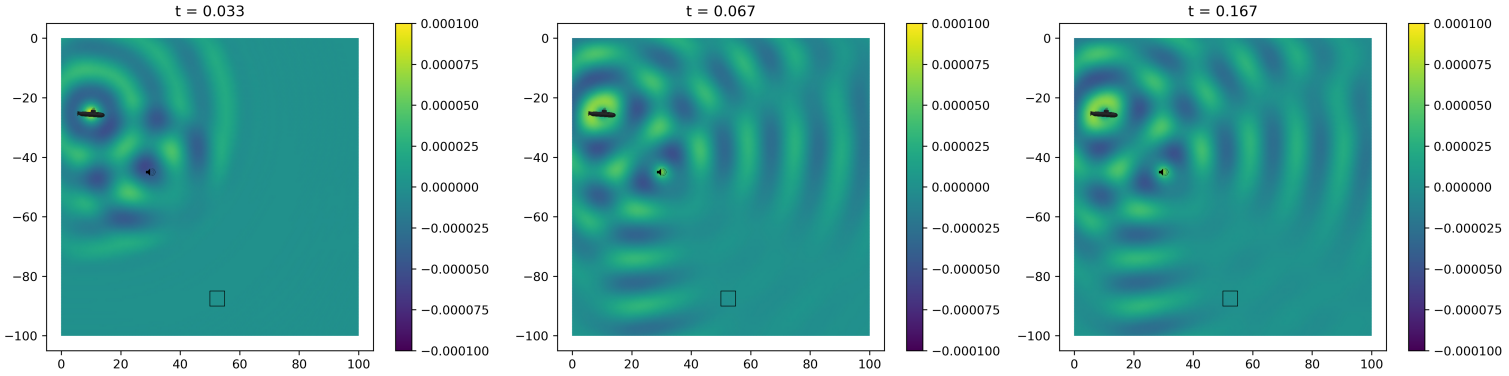

Figure 3.4. Sine source with mask

3.3 Optimization by Location of the Masking Agent

We also found that the location for the mask plays a significant role in how well we are able to cancel our source term. In order to cancel the source signal, we utilize a discrete optimization approach. Figure 3.5 shows the results from this method where the red dot is the source, the black box is our cancellation region, and each box represents the objective value if the mask is put there. In general, we find that the mask should be as close to directly between the source and the optimization region as possible. The results are slightly better when the mask is closer to the source than the optimization region.

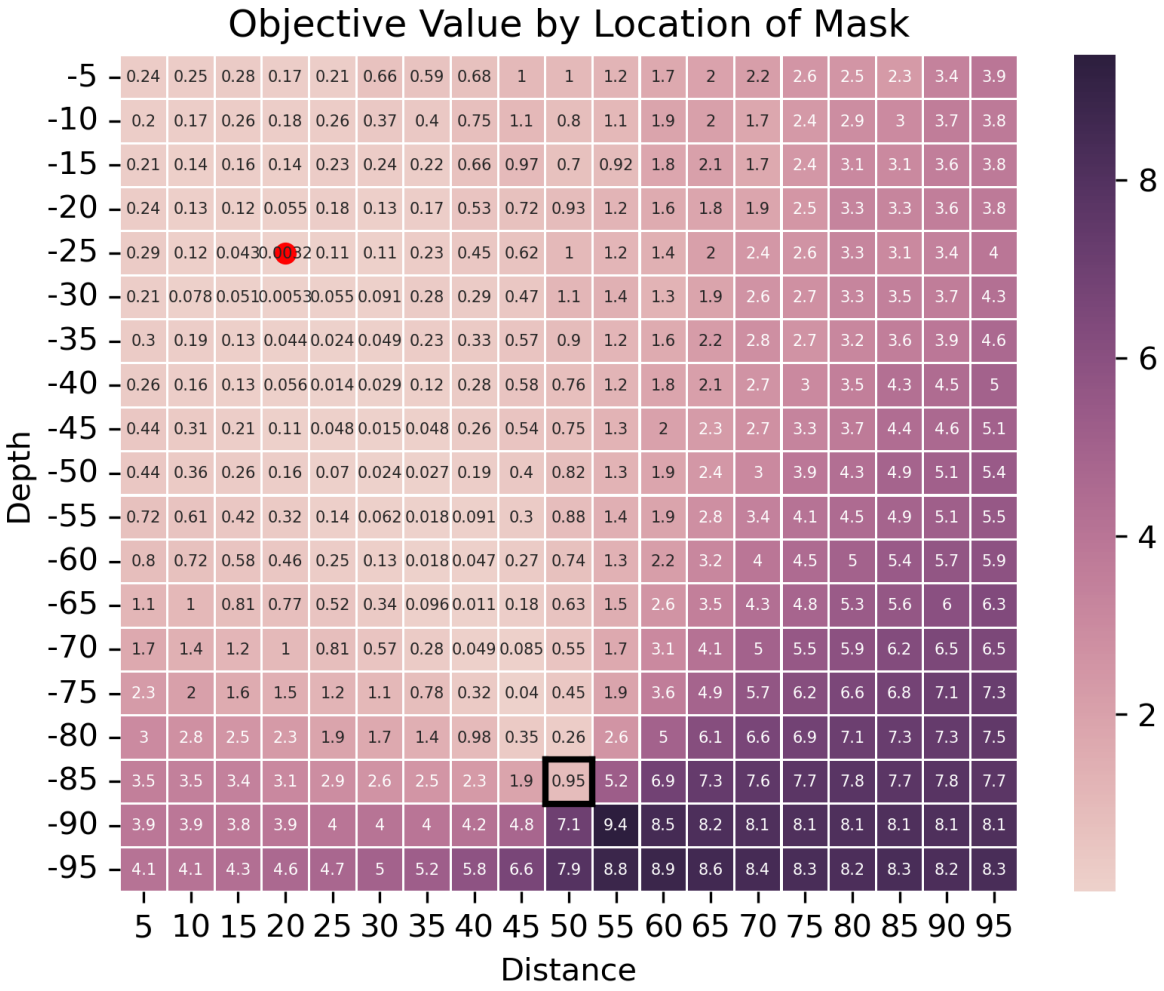

Figure 3.5. Objective value by location of the mask

3.4 Conclusion

Our research looked into the feasibility of masking a known signal underwater. We were able to successfully model both a constant sound source and a sinusoidal sound source given the physical constraints of how sound propagates underwater, using the wave equation and discretizing our domain in both time and space. Once we had this model, we were able to optimize our mask to minimize the 2-norm over the subdomain in which we wanted to eliminate the sound.

Our findings show that a constant source can be effectively minimized over the entire domain, but this is not a very common sound profile. A sinusoidal sound profile is more accurate for how a broader range of sounds generally behave, and we effectively minimized it, but only over a smaller subdomain. This is still a great result for our problem since we only sought to minimize the signal around specific sensors. It would be impractical to have sensors at all locations of a domain; therefore, if the locations of those sensors are known, that would be the region that needed to be minimized over.

Additionally, we identified the importance of the location of the masking agent. As a general rule it is best to have the mask as close to the source as possible and in a direct line between the source and masking region. The worst results are obtained when the masking region is between the mask and the source.

In conclusion, we have found that if specific data is known—the source sound profile, location of the sensor, and location of the mask—we are able to successfully mask the source. While we found these known values affect the overall performance of the mask, as long as we are masking over a relatively small region, we are able to achieve a result with a 100 to 1,000 times decrease in acoustic energy.

APPENDIX: Python Code

```
from dolfin import *
from multiprocessing import Process
import numpy as np
import scipy.sparse
import scipy.sparse.linalg
import matplotlib.pyplot as plt
import matplotlib.colors
from matplotlib.offsetbox import OffsetImage, AnnotationBbox
from matplotlib.cbook import get_sample_data
import math
import os
import sksparse.cholmod
import scipy.optimize
import time

# Wave speed.
c = Constant(1525.0)

# Mesh.
mesh = Mesh("wave2dRealisticUnitsMesh.xml")

# Set the time step and final time.
T = 1
Nsteps = 6000
dt = Constant(T/Nsteps)
times = np.linspace(0, T, Nsteps + 1)
# Use P1 elements.
U = FunctionSpace(mesh, "Lagrange", 1)
obj_scaling = 1e6
c_  = float(c)
dt_ = float(dt)

#region to mute over
lower_left = (50, -90)
upper_right = (55, -85)
alpha = Expression(f"0.0 + 1.0*(x[0] > {lower_left[0]} && x[0] <
    {upper_right[0]})*(x[1] > {lower_left[1]} && x[1] < {upper_right[1]})", degree=2)
g = 1000*np.sin(2*np.pi*100*times)
Nprocs = 8
NstepsPerProc = math.ceil(Nsteps/Nprocs)


# Set up the variational form.
u = TrialFunction(U)
v = TestFunction(U)
M_var = u*v*dx #mass matrix
```

```
Q_var = alpha*u*v*dx
K_var = inner(grad(u), grad(v))*dx #stiffness matrix
S_var = u*v*ds #'surface' mass matrix

def convert_to_scipy_mat(expr):
    '''this function converts from a petsc sparse matrix to
    a scipy sparse matrix without densifying in between.'''
    indptr,indices,vals = as_backend_type(assemble(expr)).mat().getValuesCSR()
    return scipy.sparse.csr_matrix((vals,indices,indptr)).tocsc()


[M, Q, S, K] = [convert_to_scipy_mat(expr) for expr in [M_var, Q_var, S_var, K_var]]


del0_x = np.repeat(20.0,Nsteps)
del0_y = np.repeat(-25.0,Nsteps)

deltaexpr0 =  f"eps*eps/(M_PI*M_PI*((x[0] - x0)*(x[0] - x0) + eps*eps)*
    ((x[1] - y0)*(x[1] - y0) + eps*eps))"
delta0 = Expression(deltaexpr0, eps = 0.1, degree = 4, x0 = del0_x[0], y0 = del0_y[0])
Delta0 = np.zeros((Nsteps, M.shape[0]))
del0_var = delta0*v*dx
for t in range(1, Nsteps-1):
    delta0.x0=del0_x[t]
    delta0.y0=del0_y[t]
    Delta0[t,:] = assemble(del0_var).get_local()
print("Finished Delta0")

def objfun(f, Delta1):
    global W
    global Delta0
    factorized_solver = sksparse.cholmod.cholesky(M + (c_*dt_/2)*S)
    W = np.zeros((Nsteps, M.shape[0])) #solutions. Time x space
    total = 0.0
    r = 1.0
    for t in range(1, Nsteps-1):
        rhs = 2.0*(M @ W[t, :]) - (c_**2)*(dt_**2)*(K @ W[t, :])
        rhs = rhs + (c_*dt_/2.0)*(S @ W[t - 1, :]) - M @ W[t - 1, :]
        rhs = rhs + (dt_**2)*f[t]*(M @ Delta1[t,:])
        rhs = rhs + (dt_**2)*g[t]*(M @ Delta0[t,:])
        W[t + 1, :] = factorized_solver(rhs)

        if t == Nsteps-1:
            r = .5 #this reweighting comes from the trapezoid rule
        total += r*W[t+1,:].T @ Q @ W[t+1,:]
    return obj_scaling*total

def gradfun(f, Delta1):
    global Delta0
    W = np.zeros((Nsteps, M.shape[0])) #solutions. Time x space
```

```
    factorized_solver = sksparse.cholmod.cholesky(M + (c_*dt_/2)*S)
    for t in range(1, Nsteps-1):
        rhs = 2.0*(M @ W[t, :]) - (c_**2)*(dt_**2)*(K @ W[t, :])
        rhs = rhs + (c_*dt_/2.0)*(S @ W[t - 1, :]) - M @ W[t - 1, :]
        rhs = rhs + (dt_**2)*f[t]*(M @ Delta1[t,:])
        rhs = rhs + (dt_**2)*g[t]*(M @ Delta0[t,:])
        W[t + 1, :] = factorized_solver(rhs)
    Lam = np.zeros((Nsteps, M.shape[0])) #adjoint solutions. Time x space
    Lam[-1,:] = (-Q @ W[-1,:])*obj_scaling
    Lam[-2,:] = (-2*Q @ W[-2,:])*obj_scaling
    r = 1.0 #this weighting comes from the trapezoid rule
    for t in range(Nsteps-2, 0, -1):
        if t == 1:
            r = .5 #this weighting comes from the trapezoid rule
        #same solver as above
        Lam[t-1,:] = factorized_solver((2*M - c_**2*dt_**2*K) @ Lam[t,:]
            + ((c_*dt_/2)*S - M) @ Lam[t+1,:] - 2*r*obj_scaling*(Q @ W[t-1,:]))

    grad_vec = -dt_**2*(Delta1.T * (M @ Lam.T)).sum(axis=0)
    return grad_vec


def AssembleFunc(x,y):
    #del1 is the masking signal
    del1_x = np.repeat(x,Nsteps)
    del1_y = np.repeat(y,Nsteps)

    #Forcing
    deltaexpr1 = f"eps*eps/(M_PI*M_PI*((x[0] - x1)*(x[0] - x1)
        + eps*eps)*((x[1] - y1)*(x[1] - y1) + eps*eps))"
    delta1 = Expression(deltaexpr1, eps = 0.1, degree = 4, x1 = del1_x[0], y1 = del1_y[0])
    u0 = Constant(0.0)
    un = interpolate(u0, U)

    del0_var = delta0*v*dx
    del1_var = delta1*v*dx

    factorized_solver = sksparse.cholmod.cholesky(M + (c_*dt_/2)*S)
    Delta1 = np.zeros((Nsteps, M.shape[0]))

    for t in range(1, Nsteps-1):
        delta1.x1=del1_x[t]
        delta1.y1=del1_y[t]
        Delta1[t,:] = assemble(del1_var).get_local()
        if t % 1000 == 0:
            print("Finished assemble step %d (t = %.3e)." % (t, times[t]))
    OptimizeFunc(Delta1)


def OptimizeFunc(Delta1):
```

```
    global x0
    global y0
    global Delta0
    W = np.zeros((Nsteps, M.shape[0])) #solutions. Time x space
    f = np.zeros(Nsteps)
    t = 0
    factorized_solver = sksparse.cholmod.cholesky(M + (c_*dt_/2)*S)
    for t in range(1, Nsteps-1):
        rhs = 2.0*(M @ W[t, :]) - (c_**2)*(dt_**2)*(K @ W[t, :])
        rhs = rhs + (c_*dt_/2.0)*(S @ W[t - 1, :]) - M @ W[t - 1, :]
        rhs = rhs + (dt_**2)*f[t]*(M @ Delta1[t,:])
        rhs = rhs + (dt_**2)*g[t]*(M @ Delta0[t,:])
        W[t + 1, :] = factorized_solver(rhs)
        if t % 1000 == 0:
            print("Finished step %d (t = %.3e)." % (t, times[t]))
    obj_scaling = 1e6
    Lam = np.zeros((Nsteps, Delta0[1,:].shape[0])) #solutions. Time x space
    Lam[-1,:] = (-Q @ W[-1,:])*obj_scaling
    Lam[-2,:] = (-2*Q @ W[-2,:])*obj_scaling
    factorized_solver = scipy.sparse.linalg.factorized(M + (c_*dt_/2)*S) #lu
    r = 1.0
    for t in range(Nsteps-2, 0, -1):
        if t == 1:
            r = .5
        Lam[t-1,:] = factorized_solver((2*M - c_**2*dt_**2*K) @ Lam[t,:]
            + ((c_*dt_/2)*S - M) @ Lam[t+1,:] - 2*r*obj_scaling*(Q @ W[t-1,:]))


    factorized_solver = sksparse.cholmod.cholesky(M + (c_*dt_/2)*S)
    print("Starting Optimization")
    result = scipy.optimize.minimize(objfun, np.zeros(Nsteps), args =(Delta1),
        jac=gradfun, method='L-BFGS-B', tol=1e-6, options={'maxiter':100, 'disp':True})
    np.save(f'./results/({x0}, {y0})ObjValue.npy', result['fun'])
    np.save(f'./results/({x0}, {y0})OptSol.npy', result['x'])
    np.save(f'./results/({x0}, {y0})Overall.npy', result)

x0=5
y0=-10
start = time.time()
while x0 < 6:
    while y0 > -96:
        AssembleFunc(x0,y0)
        y0+=-10
    y0= -5
    x0 +=5
print(f"Time to complete images: {(time.time() - start)/60} minutes")
```

List of References

[1] *Chief of Naval Operations Navigation Plan*, Department of the Navy, Chief of Naval Operations, Washington, DC, USA, 2022 [Online]. Available: https://www.navy.mil/Press-Office/Press-Releases/display-pressreleases/Article/3105576/cno-releases-navigation-plan-2022/

[2] *Commander's Intent 4.0 U.S. Submarine Force and Supporting Organizations*, Department of the Navy, Submarine Force Atlantic, Norfolk, VA, USA, 2022 [Online]. Available: https://www.sublant.usff.navy.mil/

[3] Z.-T. Wang, P.-X. Duan, T. Akamatsu, Y.-W. Chen, X. An, J. Yuan, P.-Y. Lei, J. Li, L. Zhou, M.-C. Liu *et al.*, "Riverside underwater noise pollution threaten porpoises and fish along the middle and lower reaches of the Yangtze River, China," *Ecotox. Environ. Safety*, vol. 226, p. 112860, 2021.

[4] A. N. Norris, "Acoustic cloaking theory," *Proc. R. Soc. A*, vol. 464, no. 2097, pp. 2411–2434, 2008.

[5] C. Sanders, W. Aquino, and T. Walsh, "Design of continuously graded elastic acoustic cloaks," *J. Acoust. Soc. Amer.*, vol. 143, no. 1, pp. EL31–EL36, 2018.

[6] A. H. Bokhari, A. Mousavi, B. Niu, and E. Wadbro, "Topology optimization of an acoustic diode?" *Struct. Multidisciplinary Opt.*, vol. 63, pp. 2739–2749, 2021.

[7] T. D. Rossing, Ed., *Springer Handbook of Acoustics*, 2nd ed. Berlin: Springer, 2014.

[8] N. Tanaka and Y. Kikushima, "Active wave control of a flexible beam: proposition of the active sink method," *JSME Int. J. Ser. 3*, vol. 34, no. 2, pp. 159–167, 1991.

[9] I. M. Gel'fand and G. E. Shilov, *Generalized Functions, Volume I: Properties and Operations*. New York: Academic Press, 1964.

[10] F. X. Giraldo, *An Introduction to Element-Based Galerkin Methods on Tensor-Product Bases: Analysis, Algorithms, and Applications*. Cham: Springer Nature, 2020.

[11] T. J. R. Hughes, *The Finite Element Method: Linear Static and Dynamic Finite Element Analysis*. Mineola: Dover Publications, 2000.

[12] G. Cohen and S. Pernet, *Finite Element and Discontinuous Galerkin Methods for Transient Wave Equations*. Dordrecht: Springer, 2017.

[13] B. Engquist and A. Majda, "Absorbing boundary conditions for the numerical simulation of waves," *Math. Comp.*, vol. 31, pp. 629–651, 1977.

[14] C. Geuzaine and J.-F. Remacle, "Gmsh: A 3-d finite element mesh generator with built-in pre-and post-processing facilities," *Int. J. Numer. Meth. Eng.*, vol. 79, no. 11, pp. 1309–1331, 2009.

[15] R. J. LeVeque, *Finite Difference Methods for Ordinary and Partial Differential Equations*. Philadelphia: SIAM, 2007.

[16] L. N. Trefethen, *Finite Difference and Spectral Methods for Ordinary and Partial Differential Equations*, 1996, unpublished text. Available: https://people.maths.ox.ac.uk/trefethen/pdetext.html

[17] D. C. Liu and J. Nocedal, "On the limited memory BFGS method for large scale optimization," *Math. Program.*, vol. 45, pp. 503–528, 1989.

[18] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed. New York: Springer, 2006.

[19] D. G. Luenberger and Y. Ye, *Linear and Nonlinear Programming*, 3rd ed. New York: Springer Science, 2008.

[20] A. Logg and G. N. Wells, "DOLFIN: Automated finite element computing," *ACM Trans. Math. Softw.*, vol. 37, no. 2, 2010.

[21] M. Greenspan and C. E. Tschiegg, "Tables of the speed of sound in water," *J. Acoust. Soc. Amer.*, vol. 31, no. 1, pp. 75–76, 1959.

[22] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020.

Initial Distribution List

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California