Implementing Noise Cancellation by Solving the Wave Equation Using the Finite Difference Method

Ieun Choo, Richeek Dutta, Myles Gong, Parth Shrotri, Evan Yu
University of Illinois Urbana-Champaign, Dept. of Aerospace Engineering

The wave equation in an initial boundary value problem (IBVP) that can be used to explain a multitude of phenomena ranging from music to earthquakes. Acoustic waves, seismic waves, beam bending, and electromagnetic waves can all be solved for using this equation. In this report, we outline how we implemented a novel form of noise cancellation using the wave equation. We discretize the problem in space using the method of lines, convert the second order ODE to 2 first order ODEs using the state-space model, and then use the trapezoid method to forward-propagate our solution in time. The boundary conditions enforce the amplitude of the input wave on the left boundary and zero amplitude in the right boundary, where the left boundary represents the outer shell of the noise cancelling headphone and the right boundary represents the eardrum. The initial condition is zero amplitude and zero derivative of amplitude everywhere in space. Solving this IBVP using these boundary conditions gives us the acoustic wave that always goes to zero amplitude at the eardrum irrespective of entry sound wave. Thus, the difference between this solution sound wave and the undisturbed sound wave is the wave that our headphone will have to produce to create the solution wave from the input wave. Then, we test the spatial and temporal convergence of our solution using the method of manufactured solutions (MMS).

I. Nomenclature

A	=	Matrix to solve the 1D Wave equation
a	=	Position of the outer shell of headphones
B	=	Matrix to solve 1D Wave Equation in standard from
b	=	Position of eardrum of user
c	=	Speed of sound (m/s)
F	=	Forcing function representing background noise (Magnitude)
L	=	Distance from the a to b
t	=	Time (s)
w	=	Vector representing u and $\dot{u}$
x	=	Distance (cm)
u	=	A function of time and distance that is a solution to the 1D Wave Equation
Δt	=	time-step (s)
Δx	=	distance-step (cm)

II. Introduction

This report discusses the use of a numerical solver that has the potential for use in the digital logic of *active noise cancellation* technologies. In the modern era, the demand in urban and suburban areas for noise cancelling headphones has been on the rise. Overexposure to constant load noises poses more than just an inconvenience: the European Environment Agency estimates that daily exposure to loud environmental noise contributes to 48,000 new cases of ischaemic heart disease each year, as well as 12,000 premature deaths [1]. While active research in field has been growing, leading to the development of new materials efficient at sound-absorbing, or passive noise cancellation, there is much improvement needed in the field of active noise cancellation technologies. The current technology used in noise-cancelling headphones relies on reading an external sound wave and playing a wave of the opposite phase into the air, which destructively cancels the incoming sound. This technology is highly effective at reducing noise levels for sustained sounds in the low frequency (50 Hz - 1 kHz) range. However, for higher frequency sounds that change frequently (as is often the case with environmental noise), it is more difficult to line up an opposing waveform, thus causing unwanted feedback and reducing the effectiveness of the noise cancellation effect [2].

As an alternative, we propose a noise cancelling method, which performs active noise cancellation inside the ear canal by generating a specific sound wave to continuously damp incoming external sound waves to zero before they reach the ear drum. We posit that this method is theoretically superior to current active noise cancellation techniques for several reasons. Firstly, this new method does not rely on pure destructive interference, which requires that the incoming wave and generated wave line up exactly at all times. This allows our new method to potentially have significant effects in all frequency ranges. Secondly, noise cancellation within the ear canal rather than externally prevents unwanted interference from wind and temperature gradients, which can significantly alter the way that sound waves propagate [3]. In this project, we introduce a numerical method employing a trapezoid method solution of an initial boundary value problem of the acoustic wave equation to drive this new method for active noise cancellation.

Section III discusses the development and implementation of the numerical method used to solve the noise cancelling problem. Section III.A presents details of the noise cancelling problem and its set-up and section III.B introduces the development of the finite difference method we would be using to solve the wave equation. Comparison with the manufactured solution and the convergence test for the validation of the method are discussed in Section III.C. Section III.D addresses the application of the numerical method to the noise cancelling problem with various random source noises.

Section IV analyzes quantified graphical and numerical results of the study and discusses about the noise cancelling effect of our method.

Section IV finally closes by giving the summary of the methods and results with potential future improvements.

III. Methods

A. Problem Statement

The goal of this project is to model a noise cancelling headphone that uses the wave interference by generating a new wave that propagates from speaker to ear. In this problem, a numerical method is used to find the waveform of a wave, hence termed the desired wave, that is equal to an input wave (external noise) at one boundary and equal to zero at the other boundary (no noise). This desired should be an oscillating function, which goes to zero at one end. Physically, the sum of the external sound wave and the sound wave generated by the noise cancelling speaker should yield the desired wave in order for noise cancellation to occur. A digital logic circuit would be able to take the desired wave computed from the numerical solver and subtract it from readings of the external sound wave in order to reverse-engineer the necessary generated sound wave.

To find the waveform of the desired wave, we will solve a one-dimensional wave equation. A one-dimensional wave that propagates along the horizontal(x) direction from a to b while also changing in time from $t = 0$ to $t = T$ can be expressed as a solution to the wave equation (Eq.(1). The wave equation is a linear second order partial differential equation that traces a property u of the wave. We consider u as a displacement of the wave motion that occurs along the vertical direction (i.e. the loudness of the sound wave) and will be dealing all waves under the standard condition where the speed of sound $c = 343m/s$.

$$\frac{\partial^2 u}{\partial t^2} = c^2 \frac{\partial^2 u}{\partial x^2}, \quad (a < x < b) \tag{1}$$

To find a new wave generated by the headphone, a and b in the wave equation should each be defined as the position of the speaker at the opening of the ear and the position of the eardrum, respectively. We assume that the distance from speaker to eardrum is $x = 2.5$ cm and set the speaker position as $x = 0$ cm. That is, $a = 0$ m and $b = 0.025$ m.

In order to find a unique solution of Eq.(1), which is a second order differential equation in both temporial and spacial coordinates, two linearly independent initial conditions and two linearly independent boundary conditions are required. The initial and the boundary conditions are selected based on the requirements of the waveform inside the ear.

1. The Boundary Conditions

In order to perform noise cancelling, we enforce that the amplitude u of the wave should be 0 at the ear, physically meaning that there is no sound. At the speaker on the other boundary, the amplitude should follow the waveform of the external sound, or input wave function which we want to cancel. This suggests that the boundary conditions should be:

$$u(t,0) = F(t) \tag{2}$$
$$u(t,b) = 0 \tag{3}$$

Where $F(t)$ is the input wave function that represents the sound wave profile and b is the distance between the speaker and the ear, which in this case is assumed to be 2.5 cm, which is approximately the distance between the opening of the ear canal to the eardrum.

2. The Initial Conditions

The initial conditions of the region between the speaker and the ear is simply 0 for both the state and time derivative components. In mathematical terms,

$$u(0,x) = 0 \tag{4}$$
$$\dot{u}(0,x) = 0. \tag{5}$$

This is because there is no forcing function before $t = 0$. This means that every point in the domain starts with no acoustic magnitude. As every point is at $u = 0$, $\dot{u}$ is also equal to zero.

B. Development of the Finite Difference Method

As the wave equation depends both on space and time, requiring a problem-solving of IBVP (initial boundary value problem), we will develop a finite difference (FD) method, combining the solving method for a IVP(initial value problem) and a BVP (boundary value problem). Our first approach to the problem is to discretize the wave equation in space to result in an IVP. That is, to solve a space-dependent problem. Then once we arrive at the IVP, we will apply a trapezoid method to solve it since it has a high stability. Further, the trapezoid method is has a fast computation time over higher-order time stepping methods such as the commonly-used Runge-Kutta 4 method. Fast computation is desired for this application, as active noise cancelling requires near instant response from the digital logic circuit in order to accommodate rapid changes in external sounds.

The reason a finite difference method was selected over a finite element method was also to improve speed of computation. Although this sacrifices some accuracy, it will be shown later that this decrease does not affect the absolute accuracy too much.

1. Method of Lines

We can first discretize the spatial domain to have $n + 1$ equally distributed points (Eq.(6):

$$x_j = a + \frac{(b-a)(j-1)}{n} \quad (j = 1, 2, \cdots, n+1) \tag{6}$$

Then, we can use a Lagrange polynomial to approximate the spatial dependence of u in terms of finite number of locally defined basis functions. The second-order spatial representation would be:

$$u(x,t) \approx \sum_{i=j-1}^{j+1} b_i(t) L_i^j(x) \tag{7}$$

where $L_i^j(x)$ is the i^{th} Lagrange basis and $b_i(t)$ is the unknown coefficient. Moreover, since the basis functions are locally defined, we can plug $u_i(t)$ into $b_i(t)$ (derived from Eq.(8)) to rewrite Eq.(7) as Eq.(9) where $u_i(t) \approx u(x_i, t)$.

$$u(x_j,t) \approx \sum_{i=j-1}^{j+1} b_i(t) L_i^j(x) = b_j(t) \tag{8}$$

$$u(x,t) \approx \sum_{i=j-1}^{j+1} u_i(t) L_i^j(x) \tag{9}$$

Substituting Eq.(9) to the wave equation (Eq.(1)), we get

$$\sum_{i=j-1}^{j+1} \dot{u}_i(t) L_i^j(x_j) = c^2 \sum_{i=j-1}^{j+1} u_i(t) \frac{d^2 L_i^j}{dx^2}|_{x=x_j}. \tag{10}$$

Finally, we use the definition of the Lagrange polynomial

$$L_{j-1}^j(x) = \frac{(x - x_j)(x - x_{j+1})}{(x_{j-1} - x_j)(x_{j-1} - x_{j+1})} = \frac{1}{2\Delta x^2}(x - x_j)(x - x_{j+1}) \tag{11}$$

to derive

$$\sum_{i=j-1}^{j+1} \dot{u}_i(t) L_i^j(x_j) = \dot{u}_j(t), \quad \sum_{i=j-1}^{j+1} u_i(t) \frac{d^2 L_i^j}{dx^2}|_{x=x_j} = \frac{1}{\Delta x^2}(u_{j-1} - 2u_j + u_{j+1}) \tag{12}$$

which will be substituted to Eq.(10) to result in a IVP for advancing approximate solution at x_j and u_j:

$$\dot{u}_j(t) = \frac{c^2}{\Delta x^2}(u_{j-1} - 2u_j + u_{j+1}) \quad (j = 2, \cdots, n) \tag{13}$$

Since there are no initial value problem for u_1 and u_{n+1}, we can write the Eq.(13) in the matrix form as

$$\overset{\cdot}{\begin{bmatrix} u_2 \\ u_3 \\ \vdots \\ u_{n-1} \\ u_n \end{bmatrix}} = \frac{c^2}{\Delta x^2} \begin{bmatrix} 2 & -1 & \cdots & 0 & 0 \\ -1 & 2 & -1 & \cdots & 0 \\ \vdots & \vdots & \ldots & \vdots & \vdots \\ 0 & 0 & \cdots & 2 & -1 \\ 0 & 0 & \cdots & -1 & 2 \end{bmatrix} \begin{bmatrix} u_2 \\ u_3 \\ \vdots \\ u_{n-1} \\ u_n \end{bmatrix}, \tag{14}$$

where

$$A = \frac{c^2}{\Delta x^2} \begin{bmatrix} 2 & -1 & \cdots & 0 & 0 \\ -1 & 2 & -1 & \cdots & 0 \\ \vdots & \vdots & \ldots & \vdots & \vdots \\ 0 & 0 & \cdots & 2 & -1 \\ 0 & 0 & \cdots & -1 & 2 \end{bmatrix}. \tag{15}$$

That is, $\dot{u} = Au$ with the initial conditions associated as discussed in Section III.A.2.

2. The State-Space Model

The wave equation is a second-order PDE, which we can put into a state-space model. We newly define a vector w that contains both u and its time derivative:

$$w = \begin{bmatrix} u \\ \dot{u} \end{bmatrix}. \tag{16}$$

Then, by defining a new matrix B, we can finalize our system for w to $\dot{w} = Bw$:

$$\dot{w} = \begin{bmatrix} \dot{u} \\ \ddot{u} \end{bmatrix} = \begin{bmatrix} \dot{u} \\ Au \end{bmatrix} = \begin{bmatrix} 0_{n-1} & I_{n-1} \\ A & 0_{n-1} \end{bmatrix} \begin{bmatrix} u \\ \dot{u} \end{bmatrix} = Bw, \tag{17}$$

where

$$B = \begin{bmatrix} 0_{n-1} & I_{n-1} \\ A & 0_{n-1} \end{bmatrix}. \tag{18}$$

By expanding the matrix equation in Eq. 17, we return the system $\ddot{u} = \frac{c^2}{\Delta x^2} Au$ and $\dot{u} = I\dot{u}$, which we expect from Eq. 14

3. Solving the IVP using the Trapezoid Method

We have arrived at the initial value problem $f(w_k, t_k) = \dot{w} = Bw$. To solve the IVP, we chose to apply the trapezoid method to find a solution with high stability:

$$w_{k+1} = w_k + \frac{\Delta t}{2}(f(w_k, t_k) + f(w_{k+1}, t_{k+1})) = w_k + \frac{\Delta t}{2}(Bw_k + Bw_{k+1}) \tag{19}$$

Then, we can finally recast the equation as

$$\begin{aligned} (I - \frac{\Delta t}{2}B)w_{k+1} &= w_k + \frac{\Delta t}{2}Bw_k \\ w_{k+1} &= (I - \frac{\Delta t}{2}B)^{-1}(w_k + \frac{\Delta t}{2}Bw_k) \end{aligned} \tag{20}$$

We will advance this system (Eq. 20) by using the initial conditions discussed in Section III.A.2.

C. Validation of the Finite Difference Method

1. Validation Method

To validate that our finite difference method is appropriate for the development of a noise-cancelling device, we conducted convergence tests using the method of manufactured solutions. We first compared our numerical solution to a manufactured solution and checked if the error converged at the expected rate.

For the validation tests, we used a sine wave for the source noise we want to perform noise-cancelling on for 5 periods (10π) at various Δt values. The manufactured solution for the wave displacement to be 0 at ear and $\sin(880\pi t)$ at the speaker is

$$\begin{aligned} u_{\text{manufactured}} &= \frac{(L-x)^2}{L^2} sin(t) \\ \dot{u}_{\text{manufactured}} &= \frac{(L-x)^2}{L^2} cos(t). \end{aligned} \tag{21}$$

This fully defines the desired state-space vector the models the system. By comparing our numerical solution to the manufactured solution defined above, and by observing that our spatial and temporal errors converge at the expected rate, we can be confident that our solution works as intended.

2. Validation Results

The plots shown in Fig. 1 and 2 illustrate the spatial and temporal errors are a function of Δx^2 and Δt^2 respectively. The order of spatial discretization used is 2. Thus the expected spatial convergence rate is $O(\Delta x^2)$. Trapezoidal method is used to forward propagate the discretized IBVP. Because the trapezoidal method has a convergence rate of $O(\Delta t^2)$, the temporal convergence rate is also expected to be $O(\Delta t^2)$. Therefore, these plots show expected behaviour and prove that our solution to the IBVP is accurate.

D. Implementation of the Method to Generate the Noise Cancellation

1. Source Modeling

As a sample input to test our methods, we decided to use a simple sine function to avoid the additional computational cost of calculating the time derivative of a sound file numerically at each point. The input wave function at the boundary was chosen to be $u(t) = \sin 880\pi t$, which models the a sinusoidal sound wave of an A440 musical note. We will use this boundary condition, along with the initial conditions and the trapezoidal method described in Eq. 20 to visualize a wave solution between points a and b at each time step.

When being used in noise cancellation, the input function otherwise is an unknown function that represents the background noise. For these situations, we can calculate the derivative using numerical differentiation of last two data points. The solution process in this case is unchanged using the trapezoidal method, though physically the sensor would initially have to read a single extra data point, which the digital logic could use alongside the prescribed initial conditions to compute a time derivative of the input function.

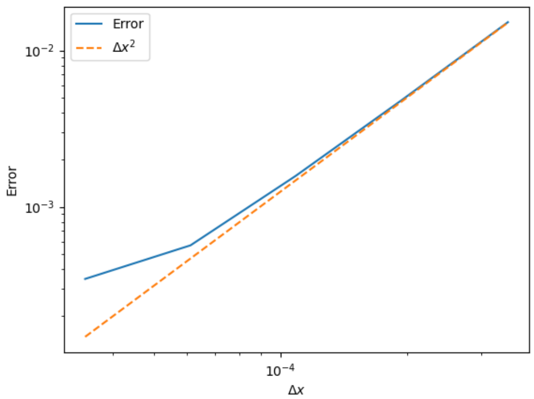

Fig. 1 Spatial convergence (Error vs Δx)

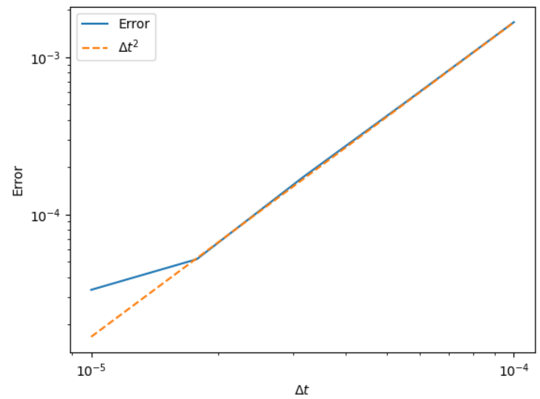

Fig. 2 Temporal convergence (Error vs Δt)

2. Code Structure

The code is structured into 3 primary files: One for the main runs that will actually solve the problem in practice, one that contains the code for the numerical method, and one that runs the convergence tests. The numerical method file contains a `Simulator` class that will run the simulation for a given Δx, Δt, initial condition, etc. This is then called from both the convergence test file and the main file to perform the simulations. Finally, we have a short plotting script to facilitate plotting the results of the main file.

IV. Results

All results in this section are from the dummy sine input with $\Delta t = 0.01s$ and $\Delta x = 3.89E - 3cm$. We see in Fig. 3 that the amplitude at $x = a$ (the position of the speaker) for all time steps matches the input wave. From this figure, we can clearly see that the particle at the headphone's outer shell moves in a sine wave. In Fig. 4 the amplitude at $x = b$ at every time step. As clearly demonstrated, the amplitude at the eardrum is zero at every time step. From this, it is clear that our numerical solution accurately produces the boundary conditions that we had specified in our problem statement.

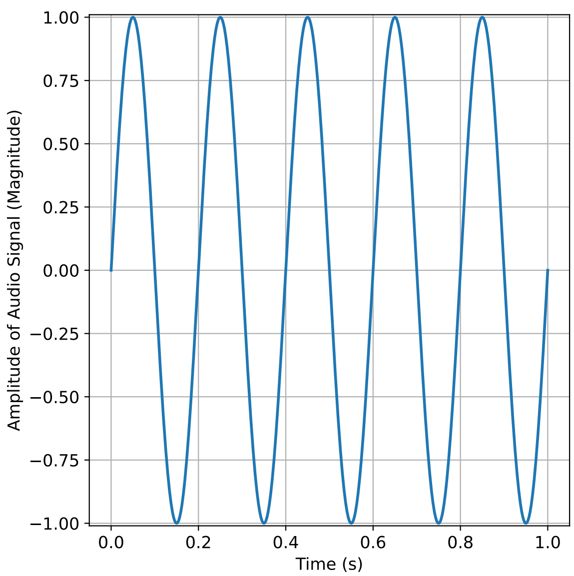

Fig. 3 u vs t for $x = a$

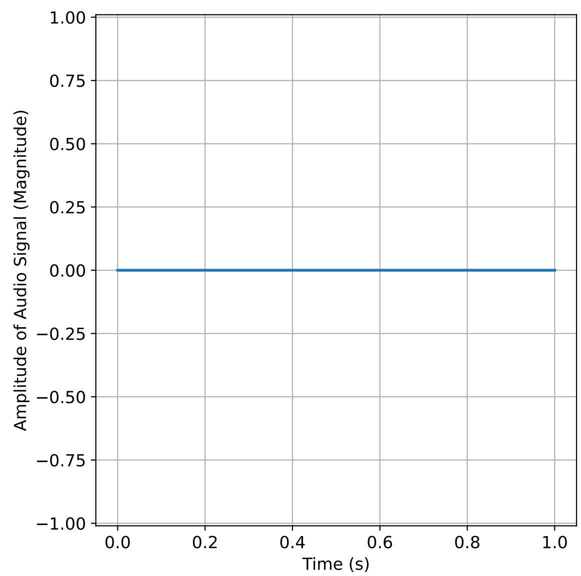

Fig. 4 u vs t for $x = b$

Next, we plot the amplitude of real wave u for $a \leq x \leq b$ at some random time steps in Fig. 5 to Fig. 8.

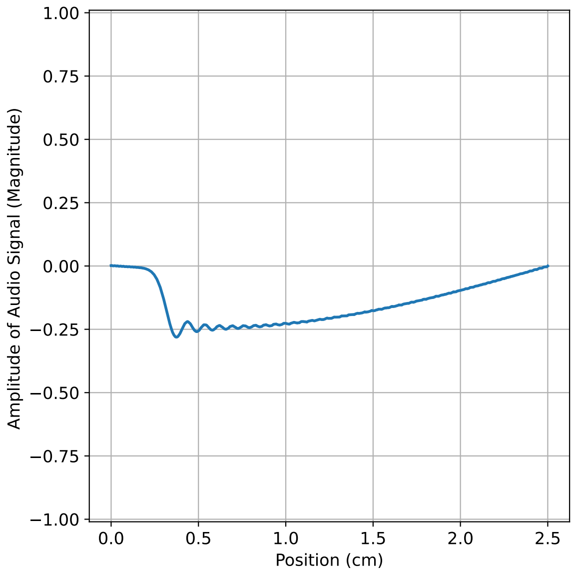

Fig. 5 u **vs** x **for** $t = 0.25s$

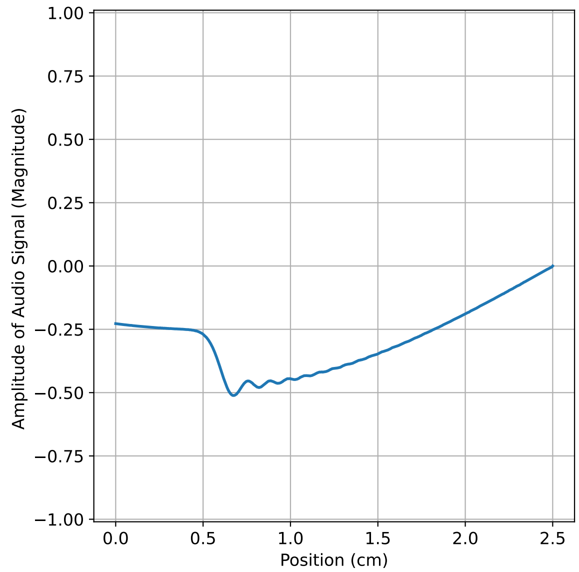

Fig. 6 u **vs** x **for** $t = 0.5s$

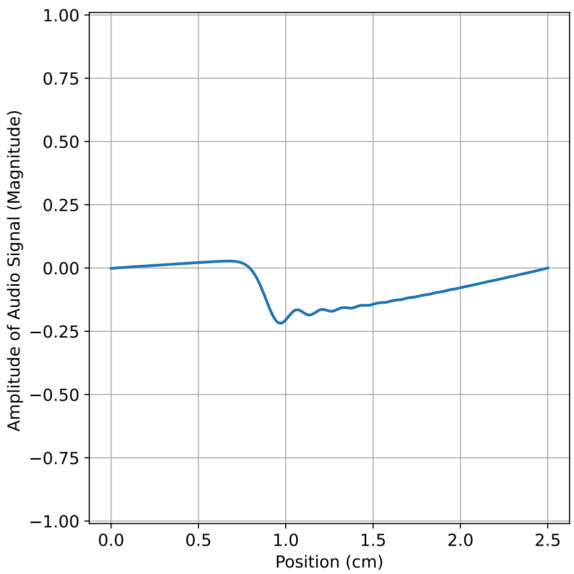

Fig. 7 u **vs** x **for** $t = 0.75s$

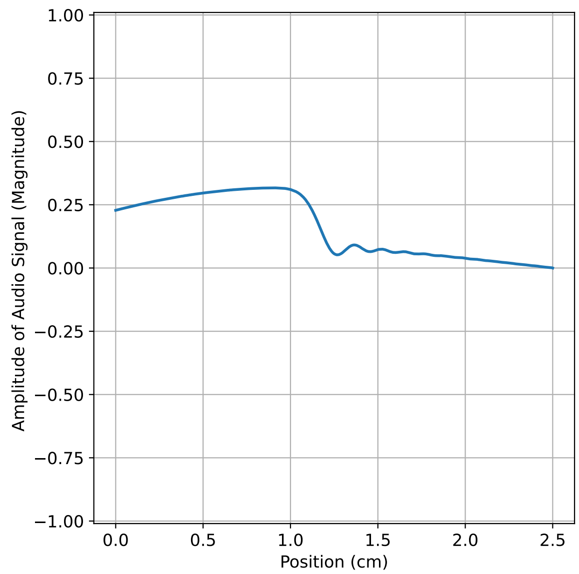

Fig. 8 u **vs** x **for** $t = 1s$

V. Conclusion

As shown from our results, irrespective of the initial amplitude at $x = a$, our solution to the IBVP leads to an amplitude of zero at $x = b$. Thus by subtracting the unchanged waveform from the waveform that is our solution, we can find the waveform that the headphone has to apply to make sure that the amplitude at $x = b$ is zero.

The next step in this project would be further testing and further proving the capabilities of this system. To further test the system in "real world" scenarios, we can use random wave files to validate our solution. To do this, we can download a random wave file from the internet and read it into the program as a list of amplitudes as a function of time. The solver should be able to apply a waveform to this random input wave such that the amplitude at the eardrum is still zero. Traditional noise cancellation works a bit differently than what we have demonstrated here. There, the headphone reads a packet of acoustic waves and generate a waveform which is exactly the negative of this wave packet, making the amplitude of the resultant sound wave equal to zero, effectively cancelling out the sound packet. Here, instead of discretizing sound into packets, the headphones are providing continuous sound waves that only enforce the amplitude at the last time step to be zero. This method will help in dampening the sound waves continuously and has the potential to have better performance in cancelling sound waves of frequency higher than 1kHz than conventional noise cancelling headphones.

Appendix

A. Link to Source Code

Source Code or `https://github.com/EvanY14/AE370-Group-Project-2`

B. Selections from Source Code

```python
import numpy as np
import pandas as pd
import sys
import os
from tqdm import tqdm
import matplotlib.pyplot as plt
sys.path.insert(0, os.path.abspath(
    os.path.join(os.path.dirname(__file__), '..')))

from trap import Simulator

# Dummy wave function
input_wave = np.sin(880*np.linspace(0, 10*np.pi, 10000))
print(input_wave[7500])
sampling_rate = 96e3 # Hz

# Simulation parameters
c = 343 # Speed of sound in air m/s
a = 0 # Position of speaker (origin)
b = 0.025 # Distance from speaker to ear in m

n = int((b-a)/c*sampling_rate) # Number of data points in wave between ear and speaker

u_b = np.zeros(len(input_wave)) # Boundary condition at ear
u_a = input_wave # Boundary condition at speaker

initial_condition = np.zeros(n) # Initial condition for wave function

u_dot = np.cos(880*np.linspace(0, 10*np.pi, 10000)) # Derivative of wave function
u_dot_initial = np.zeros(n) # Initial condition for derivative of wave function

u = np.append(initial_condition, u_dot_initial).flatten() # Initial condition for simulation
print(np.shape(u))
u_store = np.zeros((len(input_wave), n)) # Store data points for plotting (times in rows,
    positions in columns)
sim = Simulator(c, 1/sampling_rate, initial_condition, 0, len(input_wave), n, (b-a)/n)
for i in tqdm(range(len(input_wave)-1)):
    u = sim.trap_forward_prop(u, i+1, u_a, u_dot)
    u_store[i+1] = u[:n]
    # print(u)

np.save(os.path.join(os.path.dirname(__file__), '..', 'data', 'output.npy'), u_store)
# plt.plot(np.linspace(0, 10*np.pi, 10000), input_wave)
# plt.show()
```

```python
import numpy as np
```

```python
from tqdm import tqdm
import sys
import os

sys.path.insert(0, os.path.abspath(
    os.path.join(os.path.dirname(__file__), '..')))

class Simulator:
    A = None
    B = None
    I = None
    def __init__(self, c, dt, x0, t0, tf, n, dx):
        self.c = c
        self.dt = dt
        self.x0 = x0
        self.t0 = t0
        self.tf = tf
        self.n = n
        self._B(n, c, dx)
        self.I = np.eye(self.B.shape[0])
        self.F = np.linalg.inv(self.I - self.dt/2*self.B)

    def _A(self, n, c, dx):
        A = np.diag(np.ones(n-2), -1) + np.diag(np.ones(n-2), 1) + np.diag(-2 * np.ones(n-1))
        self.A = c**2/(dx**2)*A

    def _B(self, n, c, dx):
        self._A(n, c, dx)
        self.B = np.block([[np.zeros((n-1,n-1)), np.eye(n-1)], [self.A, np.zeros((n-1, n-1))]])

    def trap_forward_prop(self, u, t, u_a, u_dot):
        u_i_plus_1 = np.copy(u)
        # boundary conditions
        u_i_plus_1[1:-1] = self.F@(u[1:-1] + self.dt/2*(self.B@u[1:-1]))
        u_i_plus_1[0] = u_a[t]
        u_i_plus_1[-1] = 0
        u_i_plus_1[int(len(u)/2)-1] = 0
        u_i_plus_1[int(len(u)/2)] = u_dot[t]
        return u_i_plus_1

if __name__ == '__main__':
    sim = Simulator(1, 0.01, 0, 0, 1)
    sim.trap_forward_prop(np.array([[1, 2], [3, 4]]), 0.1, 0.1, 0.1, 0.1)
```


Acknowledgments

The authors of this paper would like to thank the AE 370 course staff in their support for developing this complex problem.

References

[1] Peris, E., "Noise pollution is a major problem, both for human health and the environment," *EEA Newletter*, 2020.

[2] Dragan, L., "What Your Noise-Cancelling Headphones Can and Can't Do," *Wirecutter*, 2020.

[3] Hannah, L., "Wind and Temperature Effects on Sound Propagation," *New Zealand Acoustics*, Vol. 20, No. 2, 2007.