Tracking the Submarine in the Puget Sound

Zhiyang Cheng

January 26, 2021

1 Introduction and Overview

This report we are hunting for a submarine in the Puget Sound using noisy acoustic data. It is a new submarine technology that emits an unknown acoustic frequency that you need to detect. Using a broad spectrum recoding of acoustics, data is obtained over a 24-hour period in half-hour increments. Unfortunately, the submarine is moving, so its location and path need to be determined.
The code and report is trying to locate the submarine and find its trajectory using the acoustic signature. Also identify the acoustic admissions of this new class of submarine. The data is provided and it is named subdata.mat. This contains 49 columns of data for measurements over a 24-hour span at half-hour increments in time.

2 Theoretical Background

2.1 Fast Fourier Transform

The Fast Fourier Transform (Eqn.1) can take a discrete function in time space and convert it into frequency space by breaking up the function into sines and cosines. The transformed function can then be converted back into time space by applying an Inverse Fast Fourier Transform (Eqn.2).

$$F(k) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} e^{-ikx} f(x) dx \tag{1}$$

$$f(k) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} e^{ikx} f(k) dk \tag{2}$$

The Fast Fourier transform has been widely used for time-dependent phenomena since it is the method for transforming a function of time into a function of frequency.

2.2 Gaussian filtering

Gaussian filtering is another important method that is been used in this report. Gaussian filtering (Eqn.3) in this assignment is used to remove noise and detail.

$$f(k) = e^{-\tau(k-k0)^2} \tag{3}$$

3 Algorithm Implementation and Development

3.1 Setting up

The first several lines of the Matlab codes was provided.
It is defined in the interval from -L to L and then we want to change the time into frequency. As I mentioned

in the Appendix A below, we are building up the 3D matrix using the Matlab function meshgrid.

3.2 Averaging and Finding the Center Frequency

Next, we want to find out the center frequency using the aveaging method. As I mentioned in the Appendix A below, we are transforming the time domain data to be in the frequency domain with the function fftn. A way to understand this is that we are averaging the spectrum so that we will weigh the random noise less important and we can more accurately finding the path of the submarine.
After averaging, we learned that the target frequency will take up the largest portion of the signal. Another way to say it we want to find the maximum value and it Matlab we can do it through using the max function.

3.3 Gaussian filtering And Plotting the Result

Now, we can set up the Gaussian filter function. In the Appendix B which is my Matlab code, I named this function 'filter.' Multiplying the filter function and the data we got from above, we can take away all the noise. As I mentioned in the Appendix A below, ifftn is the Inverse Fast Fourier transform function and we use it here to transform the data back to be in time domain. Lastly, with all the steps we did above, we only have the signal data that we want which is data with no random noise. With the cleaned data, we can plot out the path of the submarine in the Puget Sound.
The path of the submarine is shown below.

4 Computational Results

My computational results is below including a graph of the submarine movement and a table indicating the locations.

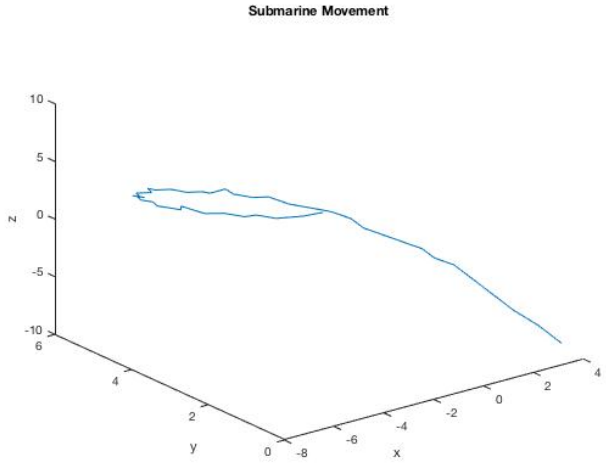

Figure 1: Here is a picture of the path of the submarine in the Puget Sound.

	x	y	z
1	3.1250	0	-8.1250
2	3.1250	0.3125	-7.8125
3	3.1250	0.625	-7.5
4	3.1250	1.25	-7.1875
5	3.1250	1.5625	-6.875
6	3.1250	1.875	-6.5625
7	3.1250	2.1875	-6.25
8	3.1250	2.5	-5.9375
9	3.1250	2.8125	-5.625
10	2.8125	3.125	-5.3125
11	2.8125	3.4375	-5
12	2.5	3.75	-4.6875
13	2.1875	4.0625	-4.375
14	1.875	4.375	-4.0625
15	1.875	4.6875	-3.75
16	1.5625	5	-3.4375
17	1.25	5	-3.125
18	0.625	5.3125	-2.8125
19	0.3125	5.3125	-2.5
20	0	5.625	-2.1875
21	-0.625	5.625	-1.875
22	-0.9375	5.9375	-1.875
23	-1.25	5.9375	-1.25
24	-1.875	5.9375	-1.25
25	-2.1875	5.9375	-0.9375
26	-2.8125	5.9375	-0.625
27	-3.125	5.9375	-0.3125
28	-3.4375	5.9375	0
29	-4.0625	5.9375	0.3125
30	-4.375	5.9375	0.625
31	-4.6875	5.625	0.9375
32	-5.3125	5.625	1.25
33	-5.625	5.3125	1.5625
34	-5.9375	5.3125	1.875
35	-5.9375	5	2.1875
36	-6.25	5	2.5
37	-6.5625	4.6875	2.8125
38	-6.5625	4.375	3.125
39	-6.875	4.0625	3.4375
40	-6.875	3.75	3.75
41	-6.875	3.4375	4.0625
42	-6.875	3.4375	4.375
43	-6.875	2.8125	4.6875
44	-6.5625	2.5	5
45	-6.25	2.1875	5
46	-6.25	1.875	5.625
47	-5.9375	1.5625	5.625
48	-5.3125	1.25	5.9375
49	-5	0.9375	6.5625

Table 1: The submarine's location in the Puget Sound.

5 Summary and Conclusions

The Fast Fourier Transform works well to determine the position of a certain object. Using this method in this assignment I was able to use this method to locate a submarine in the Puget Sound using noisy acoustic data. Using the averaging methods and filtering methods helps denoise the data and leaving the useful noise signal data.
Through averaging of the spectrum, the center frequency generated by the submarine is (5.34, -6.91, 2.19). And it is shown in the table that the location we are trying to find is (-5, 0.9375, 6.5625).

6 References

http://hyperphysics.phy-astr.gsu.edu/hbase/Math/fft.html

Appendix A MATLAB Functions

- `fftn(x)` Computes the Fast Fourier Transform of the object x
- `fftshift(x)` shifts the transformed function x
- `ifftn(x)` Computes the inverse Fast-Fourier Transform of the object x.
- `reshape(x, n, n, n)` Reshape the matrix or array x into a n*n*n mratrix
- `plot3(x,y,z)` Plots three vectors or matrices (x, y, z) coordinates in three-dimensional space.
- `[X,Y,Z] = meshgrid(x,y,z)` returns 3-D grid coordinates based on the coordinates contained in the vectors x, y, and z.

Appendix B MATLAB Code

```
% Clean workspace
clear; close all; clc

load subdata.mat

L = 10; % spatial domain
n = 64; % Fourier modes

x2 = linspace(-L,L,n+1);
x = x2(1:n); y = x;
z = x;
k = (2*pi/(2*L))*[0:(n/2 - 1) -n/2:-1];
ks = fftshift(k);

[x1,y1,z1] = meshgrid(x,y,z);
[Kx,Ky,Kz] = meshgrid(ks,ks,ks);

Uave = zeros(n,n,n);
for j = 1:49
    Un(:,:,:) = fftn(reshape(subdata(:,j),n,n,n));
    Uave = Uave + Un;
end

Uave = fftshift(Uave)./49;
[m, Index] = max(Uave(:));
[Ix, Iy, Iz] = ind2sub(size(Uave), Index);
x0 = Kx(Ix, Iy, Iz);  %  5.34
y0 = Ky(Ix, Iy, Iz);  % -6.91
z0 = Kz(Ix, Iy, Iz);  %  2.19

%%
x1 = []; y1 = []; z1 = [];
tau = -0.2;
filter=exp(tau*(((Kx-x0).^2)+((Ky-y0).^2)+((Kz-z0).^2)));

for j = 1:49
    Un = fftn(reshape(subdata(:,j),n,n,n));
    Unt = fftshift(Un);
    Unft = filter.*Unt;
    Unf = ifftn(Unft);
    [Max,idx] = max(Unf(:));
    [b,a,c] = ind2sub(size(Unf),idx);
    x1(j) = a;
    y1(j) = b;
    C(j) = c;
end

plot3(x(x1),y(y1),z(C))
title('Submarine Movement')
xlabel('x')
ylabel('y')
zlabel('z')
```

```
result= [x(x1); y(y1);z(C)]';
```