AMATH 482 Homework 1

Yiqing He

January 27, 2021

Abstract

This report discusses the theoretical and mathematical process in hunting for a submarine in the Puget Sound. I would introduce the situation first, and then find the theoretical background related to it. After that, I would illustrate my algorithm implementation and development to show the techniques I used in this project.

1 Introduction and Overview

1.1 The submarine problem

I was hunting for a submarine in the Puget Sound using noisy acoustic data. This submarine was a new technology that emitted an unknown acoustic frequency that I was trying to detect. The data is obtained over a 24-hour period in half-hour increments by using a broad spectrum recoding of acoustics. The submarine was moving, so the location and path should be determined, through which I located the submarine and found its trajectory using the acoustic signature. After that, I identified the acoustic admission of this new class of submarine. The total number of data obtained is 49, and I will use this set of data to start this journey.

1.2 Averaging

The first problem I overcame was to average the spectrum. The aim for this step was to determine the frequency signature, that was, the center frequency, generated by the submarine.

1.3 Filtering

The next step was to filter the data around the center frequency in order to denoise the data and determine the path of the submarine. After that, I plotted the path of the submarine to show the result of filtering.

1.4 Sending

At the last step, I determined the position to send my P-8 Poseidon subtracking aircraft.

2 Theoretical Background

This problem used Fourier transform to back it up. This problem was just like the radar problem discussed in class. The radar tower sends electromagnetic waves continuously. When there is an aircraft, the target, the electromagnetic wave would be bounced back that the radar tower could receive. The code of the function and its Fourier transform would include the time slot to transform, number of Fourier modes, time discretization and frequency components. Then, the ideal signal in the time domain should be determined, which is the sech function of the independent variable. Thus, the data could be plotted to give a graphical illustration.

One thing that should take care of is the noise. In the lecture, the definition and way to make white noise was discussed. In this example, the noise was provided in the dataset. The most important step is the filtering. Filtering could make people better detect the signal. There are large number of filters and the one that I used for this project is $e^{-\tau((x-x0)^2+(y-y0)^2+(z-z0)^2)}$.

3 Algorithm Implementation and Development

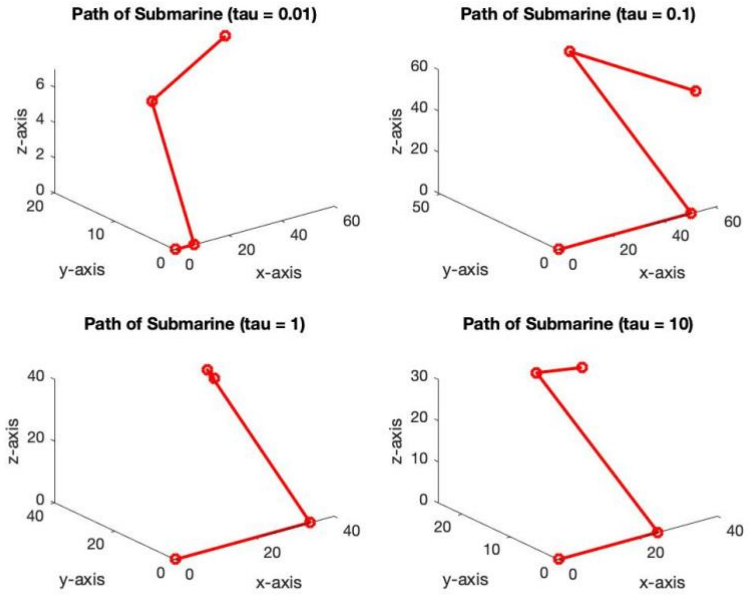
At the very first step, I created a 64*64*64 empty set, and then I averaged it over 49 realizations. The dataset subdata.mat was implemented as the noise for this submarine hunting process. The specific function “ind2sub” was used to determine the maximum point when averaging, and the result would be the center frequency generated by the submarine.

Then, I tried four numbers for tau to determine the width of the filter and x_center, y_center and z_center to determine the center of the filter. Then I plotted the four graphs to show the path of the submarine. Through which, I used plot3 to plot the path of the submarine.

4 Computational Results

The coordinates for the center frequency was (-4.7124, 3.1416, -7.8540).

The graphs for the path of the submarine were listed on the top of the next page.



5 Summary and Conclusions

Through averaging, I got the frequency signature for this hunting. Besides, I gained the paths of submarines for four different situations. This submarine problem is quite challenging but interesting. A lot of concepts and MATLAB implementations I learned from class are very useful for this project.

Appendix A

fftshift: rearranges a Fourier transform X by shifting the zero frequency component to the center of the array.
zeros: returns an n by n matrix of zeros.
[I1,I2,...,In] = ind2sub(sz,ind): returns n arrays I1,I2,...,In containing the equivalent multidimensional subscripts corresponding to the linear indices ind for a multidimensional array of size sz. Here sz is a vector with n elements that specifies the size of each array dimension.
ifftn: returns the multidimensional discrete inverse Fourier transform of an N-D array using a fast Fourier transform algorithm.

Appendix B

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

load subdata.mat % Imports the data as the 262144x49 (space by time) matrix called subdata 5

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);
[X,Y,Z]=meshgrid(x,y,z);
[Kx,Ky,Kz]=meshgrid(ks,ks,ks);

for j=1:49
   Un(:,:,:)=reshape(subdata(:,j),n,n,n);
   M = max(abs(Un),[],'all');
   close all, isosurface(X,Y,Z,abs(Un)/M,0.7)
   axis([-20 20 -20 20 -20 20]), grid on, drawnow
   pause(1)
end

ave = zeros(n,n,n);
for i = 1:49
   ave = ave + fftshift(fftn(reshape(subdata(:,i),n,n,n)));
end
ave = abs(fftshift(ave))/49;
maxAve = max(abs(ave),[],'all');
[x,y,z] = ind2sub(size(ave), find(ave == maxAve));
x_center = Kx(x,y,z)
```

```
y_center = Ky(x,y,z)
z_center = Kz(x,y,z)

tau = 0.01;
filter = exp((-tau*(Kx - x_center).^2)+(-tau*(Ky - y_center).^2)+(-tau*(Kz - z_center).^2));
dir = zeros(3,49);
for k = 1:49
   Un = reshape(subdata(:,k),n,n,n);
   unft = filter.*(fftshift(fftn(Un)));
   unf = ifftn(unft);
   maxF = max(abs(unf),[],'all');
   [x,y,z] = ind2sub(size(unf),find(abs(unf) == maxF));
   dir(1:k) = Kx(x,y,z);
   dir(2:k) = Ky(x,y,z);
   dir(3:k) = Kz(x,y,z);
end
subplot(2,2,1)
plot3(dir(1,:),dir(2,:),dir(3,:),'-or','Linewidth',2);
xlabel('x-axis')
ylabel('y-axis')
zlabel('z-axis')
title('Path of Submarine (tau = 0.01)')
dir(:,end)

tau = 0.1;
filter = exp((-tau*(Kx - x_center).^2)+(-tau*(Ky - y_center).^2)+(-tau*(Kz - z_center).^2));
dir = zeros(3,49);
for k = 1:49
   Un = reshape(subdata(:,k),n,n,n);
   unft = filter.*(fftshift(fftn(Un)));
   unf = ifftn(unft);
   maxF = max(abs(unf),[],'all');
   [x,y,z] = ind2sub(size(unf),find(abs(unf) == maxF));
   dir(1:k) = Kx(x,y,z);
   dir(2:k) = Ky(x,y,z);
   dir(3:k) = Kz(x,y,z);
end
subplot(2,2,2)
plot3(dir(1,:),dir(2,:),dir(3,:),'-or','Linewidth',2);
xlabel('x-axis')
ylabel('y-axis')
zlabel('z-axis')
title('Path of Submarine (tau = 0.1)')
dir(:,end)

tau = 1;
filter = exp((-tau*(Kx - x_center).^2)+(-tau*(Ky - y_center).^2)+(-tau*(Kz - z_center).^2));
dir = zeros(3,49);
for k = 1:49
   Un = reshape(subdata(:,k),n,n,n);
   unft = filter.*(fftshift(fftn(Un)));
   unf = ifftn(unft);
   maxF = max(abs(unf),[],'all');
   [x,y,z] = ind2sub(size(unf),find(abs(unf) == maxF));
   dir(1:k) = Kx(x,y,z);
```

```
    dir(2:k) = Ky(x,y,z);
    dir(3:k) = Kz(x,y,z);
end
subplot(2,2,3)
plot3(dir(1,:),dir(2,:),dir(3,:),'-or','Linewidth',2);
xlabel('x-axis')
ylabel('y-axis')
zlabel('z-axis')
title('Path of Submarine (tau = 1)')
dir(:,end)

tau = 10;
filter = exp((-tau*(Kx - x_center).^2)+(-tau*(Ky - y_center).^2)+(-tau*(Kz - z_center).^2));
dir = zeros(3,49);
for k = 1:49
    Un = reshape(subdata(:,k),n,n,n);
    unft = filter.*(fftshift(fftn(Un)));
    unf = ifftn(unft);
    maxF = max(abs(unf),[],'all');
    [x,y,z] = ind2sub(size(unf),find(abs(unf) == maxF));
    dir(1:k) = Kx(x,y,z);
    dir(2:k) = Ky(x,y,z);
    dir(3:k) = Kz(x,y,z);
end
subplot(2,2,4)
plot3(dir(1,:),dir(2,:),dir(3,:),'-or','Linewidth',2);
xlabel('x-axis')
ylabel('y-axis')
zlabel('z-axis')
title('Path of Submarine (tau = 10)')
dir(:,end)
```