AMATH 482 Assignment 1: A Submarine Problem

Ming Lou

Abstract:

The project mainly tracks the path of a submarine through applications of the Fourier transform. We were given a noisy data file at the beginning. During the process of denoising, the approaches like averaging and filtering plays an important role. After the data was cleaned, I determined the path of the submarine with the denoised data.

Section I. Introduction and Overview :

To decide where we should send our P-8 Poseidon sub-tracking aircraft, we are hunting for a submarine in the Puget Sound using noisy acoustic data. The new submarine technology emits some unknown acoustic frequency that we need to detect. What we have is data obtained over a 24-hour period in half-hour increment. Unfortunately, the submarine is moving all the time, so we need to determine its path and location.

As the raw data is very noisy which is hard for us to track the acoustic frequency emitted by the submarine, we have to denoise the data and create a clearer view to determine the path and location of the submarine. To start, we firstly determine the central frequency by averaging approach, which helps to remove the white noise. With the point of the central frequency, we are able to create a filter around this 3-D point. After the data was denoised, the location of the submarine can be determined.

Section II. Theoretical Background

Fourier Transform: The most basic logic for this project is based on the following two equations ((1) and (2)) called Fourier Transforms. The Fourier transform can use $\hat{\boldsymbol{f}}(\boldsymbol{k})$ to represent a function $\boldsymbol{f}(\boldsymbol{x})$**.** The following two equation can be transferred to each other once

we know one of them. The main advantage of Fourier transforms is that we can converts between space (or time) and frequency; However , the Fourier Transform assumes infinite domain, this is not what we finally want.

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

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

Fourier Series: Fourier series (eqation 3) illustrate that a function $f(x)$ can be expressed as a function with **cos** and **sin** within finite domain**.**

$$f(x) = \frac{a_0}{2} + \sum_{k=1}^{\infty} \left(a_n \cos(kx) + b_n sin(kx) \right),\ x \in [-\pi, \pi] \qquad (3)$$

Averaging: Averaging is a strategy used to clean the white noise and help us to determine where the central frequency is. The main logic of averaging is just sum up several Fourier components. Since the white noise is normally distributed, which contains mean zero, when the number is large, they will be detracted by themselves.

Filtering: Filtering is a function that we will multiply the signal by. Once we know where the central frequency is, we can utilize the filtering function to filter out the noise in order to better detect the signal. A common used filtering function is Gaussian function (4). The τ determines the width of the filter and the constant k_0 determines the centre of the filter.

$$F(k) = e^{-\tau(k-k_0)^2} \qquad \textbf{(4)}$$

Section III. Algorithm Implementation and Development

Main Steps :

1. Loading the dataset to MATLAB and visualizing the raw data
2. Averaging the raw data and find the central frequency
3. Multiplying the filtering function to the signal and tracking the trace of the submarine
4. Determining the final point

Details:

Firstly, I loaded the SUBDATA into MATLAB and defined some basic parameters like spatial domain and Fourier modes. The raw data is a 262144 × 49 dataset, I reshaped it to 49 different 64 × 64 ×64 matrices. Then I set a For Loop to make 49 plots to see how these noisy data performed. The result shows they are really noisy, almost nothing I can detect from these plots.

Secondly, I applied the averaging method to the raw data. To do so, I created a 64 × 64 ×64 zero matrix called **ave** and set a for loop to sum up the 49 different matrices reshaped from the raw data. In the For Loop, I used the **fftn** command to transform all of the data in the spatial domain to the frequency domain. After the For Loop, all the data are saved together in the **ave** matrix I created at the beginning**.** Divided by the column number 49, most white noise was cleaned. Then I utilized the **fftshift** command to rearrange the Fourier Transform. As the result shows, I got a better view to determine the central frequency. To obtain the specific point of the central frequency, I used **[M,I] = max(avg)** command and **[i,j,l] = ind2sub([64,64,64], I).**

With the central frequency point, I was able to create the Gaussian filter. I replaced my 3-D point into the Gaussian function, which is $e^{(-\tau\times((cx^2+(cy)^2+(cz)^2)}$. To save all the locations that the submarine took, I created a 49×3 matrix. Through the for loop, I multiply each 49 matrices to the filtering function. After denoising, I used the three location vectors I created to record the index of the location. Finally, all spots are recorded in my **spot** matrix. Finally, I used the **plot3** command to make the graph for tracking the submarine. The last step is vey easy now, I only need to show the last point in my **spot** matrix, which is exactly the final location of the submarine.

Section IV. Computational Results

1. The Figure 1 for the signal after averaging shows a relatively clean view. We can make sure the place of the acoustic signal

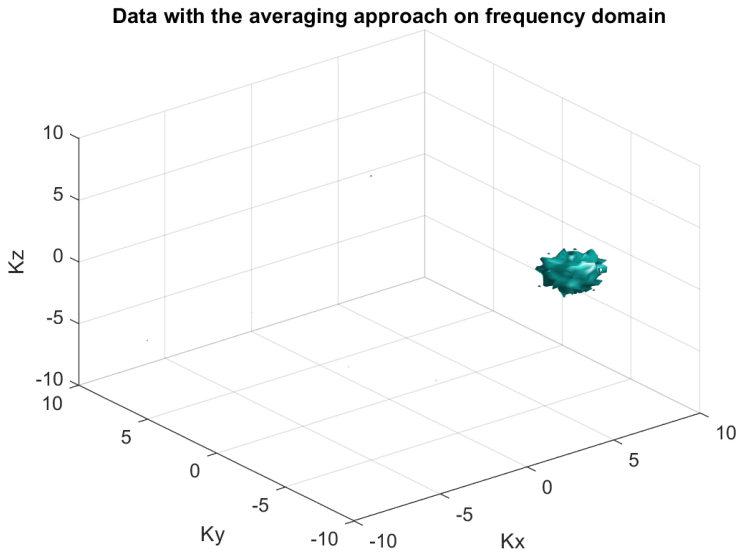

Figure 1 : Data with averaging approach on frequency domain

2. Figure 2 shows the path of the submarine, totally 49 points

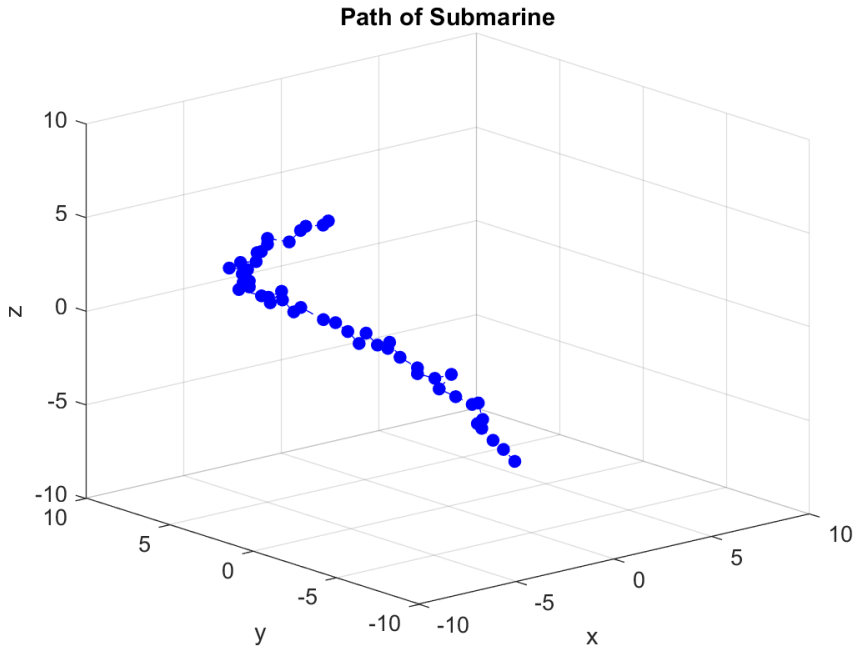

Figure 2 : path of Submarine

3. The coordinates of the center frequency of the submarine([x,y,z]): [5.3407, -6.9115, 1.8850]
4. The coordinates of the final location of the submarine([x,y,z]): [-5.3125, 0.9375, 6.2500]

Section V. Summary and Conclusions

By applying the Fourier Transform with averaging and filtering, I finally determined the path of the submarine, and get the final position ([x,y,z]): [0, 5.3125, -2.8125], the spot that we can send the P-8 Poseidon subtracking aircraft. This project applied a 3-D version of the Fourier Transform which is very meaningful for the noisy data cleaning. In the future, these knowledges and skills will play important roles in data analysis.

Appendix A. MATLAB functions used and brief implementation explanation

1. ifftn: This function computes the inverse discrete Fourier transform using a fast Fourier transform algorithm

2. fftn: Returns the multidimensional Fourier transform of an N-D array by using a FFT algorithm.

3. fftshift: This function shifts the zero-frequency component of discrete Fourier transform to center of spectrum and it helps the visualization of FFT in the frequency domain.

4. isosurface: $isosurface(X, Y, Z, V, isovalue)$ computes isosurface data from the volume data V at the isosurface value specified in isovalue. It helps us visualize the figure of data in 3-D in this project.

5. ind2sub: Used together with the find function to locate the frequency of the signal

6. plot3: plots coordinates in 3-D.

7. meshgrid: returns the 3-D grid coordinates defined by the vectors x, y, and z

8. reshape: reshapes the object A into an array with given sizes

Appendix B. MATLAB codes

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

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

 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


%% plot the averaged signal and find the center frequency
ave =  zeros (64 ,64 ,64);

for j=1:49
    Un(:,:,:)=reshape(subdata(:,j),n,n,n);
    Ut =fftn(Un);
    ave(: ,: ,:) = ave + abs (reshape(Ut, n ,n, n));
end
ave = ave ./49;
ave = abs (ave) / max (abs(ave(:)));

isosurface (Kx, Ky, Kz, fftshift(ave), 0.7)
axis ([-L L -L L -L L]), grid on, drawnow
xlabel("Kx"), ylabel("Ky"), zlabel("Kz")
title ("Data with the averaging approach on frequency domain")

aveshift = fftshift (ave);
[maximum,index] = max (aveshift(:));
[ii,jj,ll] = ind2sub ([n, n, n ], index);
cx = ks (jj) ; cy = ks (ii) ; cz = ks (ll);  %get the center frequcy

%% plot the path of the submarine
tau = 0.5;
filter = exp(-tau*((cx).^2 + (cy).^2 + (cz).^2));
```

```
spot = zeros(49, 3);
for k = 1:49
 Un(:,:,:)=reshape(subdata(:,k),n,n,n);
 utn = fftn(Un);
 unft= fftshift(filter).*utn; % Apply the filter to the signal in
frequency space
 unf= ifftn(unft);
 [Max, index] = max(abs(unf(:)));
 [index_x, index_y, index_z] = ind2sub([n,n,n], index);
 spot(k,:) = [X(index_x, index_y, index_z), Y(index_x, index_y,
index_z), Z(index_x, index_y, index_z)];
end

figure;
plot3(spot(:,1), spot(:,2), spot(:,3), "b.--",  "Markersize", 20);
axis([-L L -L L -L L]), grid on
xlabel("x"); ylabel("y"); zlabel("z");
title("Path of Submarine");
set(gca, "Fontsize", 10);

%% find the final location
final_location = [spot(49,1), spot(49,2), spot(49,3)];
```