AMATH 482 Homework 1

Ariel Luo

January 24, 2020

Abstract

This homework is about locating a submarine and finding its trajectory by analyzing an unknown acoustic frequency it emits using the Fourier Transform.

1 Introduction and Overview

In this homework, we are trying to locate a submarine in the Puget Sound which emits an acoustic frequency that we can detect. The data provided is obtained over a 24-hour period in half-hour increments while the submarine is moving. Our goal is to find its location and its trajectory using the given noisy data by

1. Determining the frequency signature generated by the submarine
2. Denoising the data and find its trajectory
3. Find its locations through the time period

2 Theoretical Background

As we learned from our lectures, the Fourier transform can decompose functions in space or time into functions in frequency.

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

This also works backwards. We can use inverse Fourier transform to obtain the functions in space or time from functions in frequency.

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

However, as we can see from 1 and 2, the domain is infinite for both equations. Since our data is not infinitely large, these two equations are not very practical. We need to perform the Fourier Transform on a limited domain, so we have to use the Fourier Series. The Fourier Series writes the Fourier transform into an infinite sum by breaking the function into cos(even) and sin(odd) functions.

$$f(x) = \frac{a_0}{2} + \sum_{i=1}^{\infty} (a_n \cos nx + b_n \sin nx) \quad x \in (-\pi, \pi]. \tag{3}$$

In this homework, we are going to use the Fast Fourier Transform(FFT) in Matlab which is a faster algorithm of the Discrete Fourier Transform(DFT). DFT is the same as the Fourier Transform, but it only defines a discrete set of points on the function. It picks out each frequency corresponding to each point.

3 Algorithm Implementation and Development

The algorithm can be summarized and separated into two main parts:

Algorithm 1: Averaging spectrum and find the frequency signature

```
Import data from subdata.mat
Create variable ave
for j = 1 : 49 do
    Reshape data point from subdata into 3D
    Perform Fast Fourier Transform
    Add to ave
end for
Perform FFTshift and calculate the average
Find the peak which is the frequency signature
```

Algorithm 2: Filtering

```
Find the center frequency location in 3D coordinates
Create filter by using the location
for j = 1 : 49 do
    apply filter
    Perform inverse FFT(IFFT) to go back to the time domain
    Find the peak where the frequency is located
    Find its coordinates
    Plot point using plot3
end for
```

4 Computational Results

The center frequency is 176.5852.
See Table 1 for the locations of submarine.
See Figure 1 for the trajectory of the submarine.

5 Summary and Conclusions

By applying what we learned in class on Fourier Transform and Fourier Series, we successfully filtered out noises in the original data. By averaging the spectrum using the Fast Fourier Transform, we also found the center frequency at 176.5852. After we filtered out the data, we were able to obtain the locations of the submarine over time and its path as we can see from the graph and the table using the Inverse Fast Fourier Transform.

Appendix A MATLAB Functions

Add your important MATLAB functions here with a brief implementation explanation. This is how to make an **unordered** list:

- `y = linspace(x1,x2,n)` returns a row vector of `n` evenly spaced points between `x1` and `x2`.
- `[X,Y] = meshgrid(x,y)` returns 2-D grid coordinates based on the coordinates contained in the vectors `x` and `y`. X is a matrix where each row is a copy of `x`, and `Y` is a matrix where each column is a copy of `y`. The grid represented by the coordinates `X` and `Y` has `length(y)` rows and `length(x)` columns.
- `B = reshape(A,sz1,...,szN)` reshapes A into a sz1-by-...-by-szN array where sz1,...,szN indicates the size of each dimension. You can specify a single dimension size of [] to have the dimension size automatically calculated, such that the number of elements in B matches the number of elements in A.

	x	y
1	3.125	0
2	3.125	0.3125
3	3.125	0.625
4	3.125	1.25
5	3.125	1.5625
6	3.125	1.875
7	3.125	2.1875
8	3.125	2.5
9	3.125	2.8125
10	2.8125	3.125
11	2.8125	3.4375
12	2.5	3.75
13	2.1875	4.0625
14	1.8750	4.6875
15	1.875	4.6875
16	1.5625	5
17	1.25	5
18	0.625	5.3125
19	0.3125	5.3125
20	0	5.625
21	-0.625	5.625
22	-0.9375	5.9375
23	-1.25	5.9375
24	-1.875	5.9375
25	-2.1875	5.9375
26	-2.8125	5.9375
27	-3.125	5.9375
28	-3.4375	5.9375
29	-4.0625	5.9375
30	-4.3750	5.9375
31	-4.6875	5.625
32	-5.3125	5.625
33	-5.625	5.3125
34	-5.9375	5.3125
35	-5.9375	5
36	-6.25	5
37	-6.5625	4.6875
38	-6.5625	4.375
39	-6.875	4.0625
40	-6.875	3.75
41	-6.875	3.4375
42	-6.875	3.4375
43	-6.875	2.8125
44	-6.5625	2.5
45	-6.25	2.1875
46	-6.25	1.875
47	-5.9375	1.5625
48	-5.3125	1.25
49	-5	0.9375

Table 1: The locations of the submarine

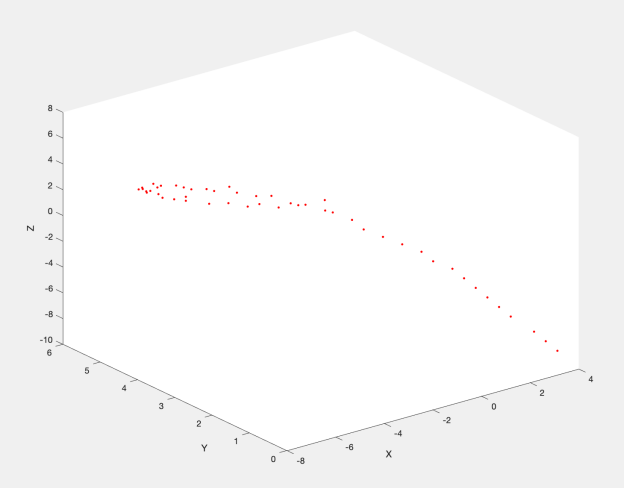

Figure 1: The path of the submarine.

- `Y = fftn(X)` returns the multidimensional Fourier transform of an N-D array using a fast Fourier transform algorithm.
- `Y = fftshift(X)` rearranges a Fourier transform X by shifting the zero-frequency component to the center of the array.
- `[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.
- `k = find(X)` returns a vector containing the linear indices of each nonzero element in array X.

Appendix B MATLAB Code

```
%% Algorithm 1
% 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);

Ut = zeros(n,n,n,49);
ave = zeros(n,n,n);
 for j=1:49
     Un(:,:,:)=reshape(subdata(:,j),n,n,n);
     ut = fftn(Un);
     Ut(:,:,:,j) = ut;
     ave = ave + ut;
 end
ave = abs(fftshift(ave))/49;
center = max(ave,[],'all')

%% Algorithm 2
[kx,ky,kz] = ind2sub(size(ave),find(ave == center));
k1 = Kx(kx,ky,kz);
k2 = Ky(kx,ky,kz);
k3 = Kz(kx,ky,kz);
[kx1,ky1,kz1]=meshgrid(k,k,k);
tau = 0.2;
filter = exp(-tau*((kx1 - k1).^2+(ky1 - k2).^2+(kz1 - k3).^2));
unft = zeros(64, 64, 64, 49);
unf = zeros(64, 64, 64, 49);
location = zeros(49,3);
for j = 1:49
    unft(:,:,:,j) = filter.*Ut(:,:,:,j);% apply filter
    inverse = abs(ifftn(unft(:,:,:,j)));
    unf(:,:,:,j) = inverse;
    peak = max(inverse,[],'all');
    [a1,b1,c1] = ind2sub(size(inverse),find(inverse == peak));
    a = X(a1,b1,c1);
    b = Y(a1,b1,c1);
    c = Z(a1,b1,c1);
    location(j,:) = [a,b,c];
    plot3(a,b,c,'.','Color','r')
    hold on
    xlabel('X')
    ylabel('Y')
    zlabel('Z')
%     pause(1)
end
```

Listing 1: Code for HW1