Amath 482 Homework 1: A Submarine Problem

Yilin Cao

January 27, 2021

Abstract

In this report, I will describe the process and result of de-noising a dataset obtained by a 24-hour period in half-hour increments using Fourier Transforms, averaging, and filtering. The dataset contains 49 columns of data, all of which are collected from a broad spectrum recoding of acoustics, aiming at detecting the clear trajectory of the submarine and its final location.

I. Introduction and Overview

Fast Fourier Transform (FFT) calculates the discrete Fourier Transform of a sequence or its inverse. It converts a signal from its space time to Fourier space and improves the runtime of computing by transiting the calculation from $O(N^2)$ to $O(Nlog(N))$. In this project, the dataset contains 49 columns of data from different times and is noisy due to the movement of the submarine. Since FFT will facilitate in finding out the desired frequency and getting rid of the noises, we will use the FFT as the main tool to find the most significant frequency in signals and extract the trajectory.

In the following sections, I will describe more methodology and concepts being used for this problem, as well as their implementation in MATLAB. I will present the result of the actual trajectory of the submarine and its final position at the very end.

II. Theoretical Background

1. FFT Algorithm

Fourier Transform is defined over an infinite domain by the following equation:

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

where the function takes signal for each value of k and project it with the oscillation function of frequency k, together normalized by square roots of 2π.

Conversely, the inverse of Fourier Transform takes a function of frequencies k and reverses it back to a function of space. It is defined as the following:

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

Fourier Series is the discrete version of Fourier Transform, and it is given by an infinite sum of sines and cosines as following:

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

FFT Algorithm is implemented to calculate the Fourier Transform of a function. Following the techniques of the divide and conquer algorithm, the computational time has reduced significantly from $O(N^2)$ to $O(Nlog(N))$.
2. Time Averaging
3. Spectral Filtering

III. Algorithm Implementation and Development

- Clean workspace and load subdata.mat into workspace
- Startup Code
    - Set up the spatial domain and Fourier modes.
    - Define the grid vectors
    - Create 3D Cartesian grid for both the spatial domain and frequency domain using *meshgrid* function.
- Step 1
    - Construct an empty 3D matrix of predetermined size for averaged signals.
    - Reshape each column of subdata into 3D
    - Fourier Transform the data into frequency domain.
    - Add up each iteration and take average
    - Normalize the data with the largest element and take the absolute value
    - Visualize the data with *isosurface*
    - Determine the center frequency
- Step 2
    - Define the Gaussian Filter with the center frequency found previously
    - Apply the filter to each signal in frequency space to denoise the data
    - Use *ifftn* to transform the signal back into spatial domain
    - Normalize the data with the largest element and take the absolute value
    - Find the center of each signal sent by the submarine
    - Store the location at each time step and plot the locations into path
    - Visualize the path using *plot3*
- Step 3
    - Find the final position of the submarine at final time step
    - Visualize the location with *isosurface*

IV. Computational Results

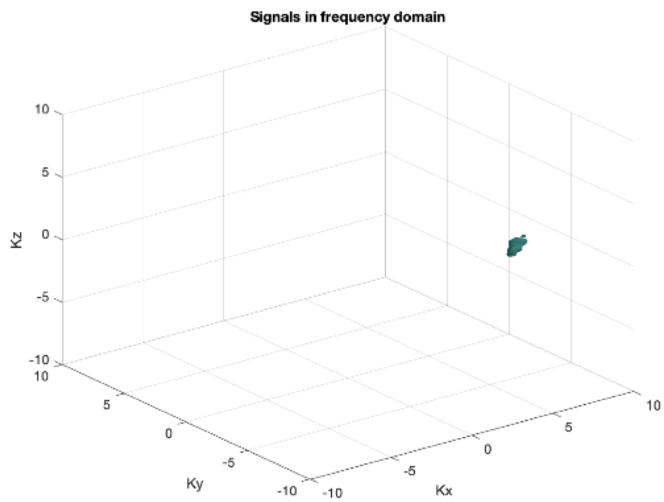

Figure 1: Frequency signature of signals

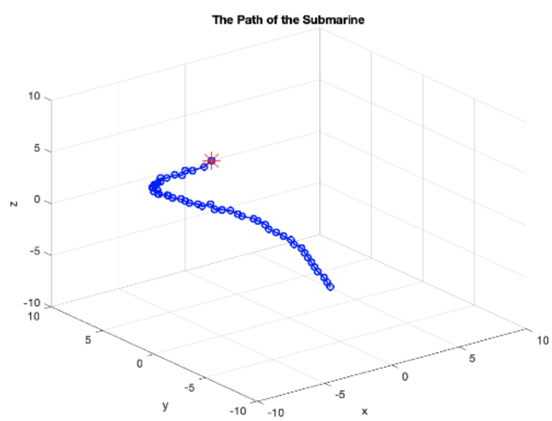

Figure 2: Plot3 visualization of the submarine trajectory

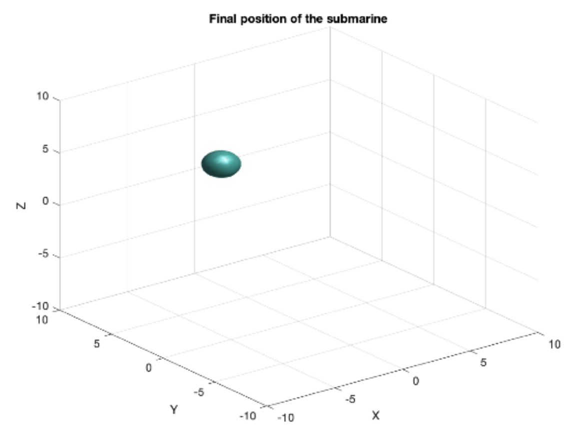

Figure 3: Isosurface visualization of final position of the submarine

Figure 1 shows the center frequency located at (5.3407, -6.9115, 2.1991) after time averaging.

After applying the Gaussian Filter to denoise the data, Figure 2 shows the path of the submarine by plotting out the center of the submarine at each time step.

Through isosurface visualization, Figure 3 shows the final position of the submarine, which is found to be (-5, 0.9375, 6.5625)

V. Summary and Conclusions

In the beginning of this project, we have constructed the spatial domain and frequency domain for our dataset. Then, by applying the averaging method of the 49 measurements within a 24-hour span, we successfully located the center frequency at (5.3407, -6.9115, 2.1991). Using this location, we can apply the Gaussian filter with a bandwidth $\tau = 0.2$ to denoise the data. In this way, we were able to estimate the path of the submarine and find the final location to send P-8 Poseidon subtracking aircraft, which is (-5, 0.9375, 6.5625).

Appendix A. MATLAB functions

abs(X): calculates the absolute value of the elements of X. When input is complex, output returns the complex modulus (magnitude).
fftn(X): returns the N-dimensional discrete fourier transform of the N-D array X.
fftshift(X): rearranges the elements of X by shifting zero-frequency component to center of spectrum.
ifftn(X): returns the N-dimensional inverse discrete fourier transform of the N-D array X.
ind2sub(SIZ, IND): returns row and column subscripts corresponding to the index matrix IND for a matrix of size SIZ.
isosurface(X,Y,Z,V,ISOVALUE): computes isosurface geometry for data V at isosurface value ISOVALUE.
load: load data from MAT-file into workspace.
linspace(X1, X2, N): generates N points between X1 and X2. This function is used to create the linearly spaced vector for the spatial domain.
max(X): calculates the maximum value of the input element.
meshgrid(xgv, ygv, zgv): replicates the grid vectors xgv, ygv, zgy to produce a Cartesian grid in 3-D space.
plot3(X, Y, Z): plot lines and points in 3-D space with matrices X, Y, Z of the same size
reshape(X, M, N, P, ...): reshapes the input X into matrix or tensors with the desired size as given.
zeros: generates a matrix filled with zeros with the predetermined size.

Appendix B. MATLAB codes

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

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

%% Set up

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);

%% Step 1.

% Average of the spectrum
ave = zeros(n,n,n);
for j=1:49
    ave = ave + fftn(reshape(subdata(:,j),n,n,n));
end
ave = abs(fftshift(ave))/49;
maxAve = max(abs(ave),[],'all');

% Plot averaged signal
figure(1)
isosurface(Kx,Ky,Kz,ave./max(ave(:)),0.6)
axis([-10 10 -10 10 -10 10]), grid on, drawnow
xlabel("Kx"), ylabel("Ky"), zlabel("Kz")
title("Signals in frequency domain");

% Determine the center frequency
[a,b,c] = ind2sub([n,n,n],find(abs(ave) == maxAve));
x_cf = ks(b);
y_cf = ks(a);
z_cf = ks(c);
```

```matlab
%% Step 2.

% Define the Gaussian Filter
tau = 0.2;
filter = exp(-tau * ((Kx - x_cf).^2 + (Ky - y_cf).^2 + (Kz - z_cf).^2));

% Determine the path of the submarine
path = zeros(49,3);
for i = 1:49
    un(:,:,:) = reshape(subdata(:,i),n,n,n);
    utn = fftshift(fftn(un));
    unft = filter.*utn; % Apply the filter to the signal in frequency space
    unf = ifftn(unft);
    maxUnf = max(abs(unf),[],'all');
    [pathX,pathY,pathZ] = ind2sub([n,n,n], find(abs(unf)==maxUnf));
    path(i,1) = X(pathX,pathY,pathZ);
    path(i,2) = Y(pathX,pathY,pathZ);
    path(i,3) = Z(pathX,pathY,pathZ);
end

% Plot the path of the submarine
figure(2)
plot3(path(:,1),path(:,2),path(:,3),'b-o','LineWidth',1.5);
axis([-10 10 -10 10 -10 10]), grid on, drawnow
title('The Path of the Submarine');
xlabel("x"), ylabel("y"), zlabel("z")
hold on;
plot3(path(49,1),path(49,2),path(49,3),'r*','MarkerSize',15);
hold off;

%% Step 3.

% Location to send P-8 Poseidon subtracking aircraft
loc = path(end,:);
isosurface(X,Y,Z,abs(unf)./max(abs(unf(:))),0.5)
axis([-10 10 -10 10 -10 10]), grid on, drawnow
xlabel("X"), ylabel("Y"), zlabel("Z")
title("Final position of the submarine");
```