# AMATH 482 Homework 1

Jiaqi Su

January 27, 2020

**Abstract**

Identifying signals with certain frequency can help us detect and locate target objects. In this project, the main goal is to track a submarine based on its 24-hour records of acoustics. Using signal processing method, the specific frequency of the submarine can be found, and by matching this desired frequency, we can get the location coordinate at each time and therefore a path over past 24-hour to track it.

## 1 Introduction and Overview

The goal of this problem is to find the locations of a submarine in the Puget Sound using noisy acoustic data. The only data we have is a record of acoustic over last 24 hours in half-hour increments. The difficulties include that the frequency generated by the submarine is unknown, and the data we have has noises. Therefore, the first step would be to find the center frequency through averaging our records of acoustic, and then we can try to denoise the data to find the position based on the center frequency.

## 2 Theoretical Background

### 2.1 Fourier series and Discrete Fourier Transform

The basic idea of signal processing is that a given function $f(x)$ can be broken into an infinite series of sines and cosines of different frequencies:

$$f(x) = \frac{a_0}{2} + \sum_{i=1}^{\infty} (a_n \cos nx + b_n \sin nx) \tag{1}$$

in a finite domain (-$\pi$,$\pi$] with coefficients $a_n$ and $b_n$.

However, in the real world, the data collected will be discrete point of value instead of a continuous function $f(x)$. Therefore, a discrete version of the Fourier Transform is needed. Given N equally-spacing points $x_1, x_2, ..., x_N$, the discrete Fourier transform returns a sequence of numbers:

$$\hat{x}_k = \frac{1}{N} \sum_{n=0}^{N-1} x_n e^{\frac{2\pi ikn}{N}}. \tag{2}$$

Each $x_k$ corresponds to the value of frequency k in our set of data. Therefore we have corresponding N value with responds to each frequency $x_k$ in the frequency domain. And Fast Fourier Transform(FFT) is just a improved way of doing discrete Fourier transform using divide and conquer algorithm so that the complexity decreases from $\mathcal{O}(N^2)$ to $\mathcal{O}(N \log N)$

### 2.2 Noise and Filters

In real world, the received data of signal is a combination of information of our target and noises. The way to identify desired information is to decompose the received signal and match with the specific frequency of our target. Filters are designed for this purpose.

A filter is a function we multiply the signal by to separate the noise and desired part of frequency signature. A normal filter is the Gaussian function:

$$F(k) = e^{-\tau(k-k_0)^2}. \tag{3}$$

The parameter $\tau$ determines the width(standard deviation) of the filter while the parameter $k_0$ is the center frequency we are looking for. Through applying filters to the signal in the frequency domain, we can filter out noisy far from our center frequency and better detect our target in the spatial domain.

### 2.3 Averaging

A part of the difficulties in real world is that the target frequency signature is sometimes unknown, which requires some work to find it before using it to create appropriate filters. The method introduced to address this problem is averaging. It states that the center of frequency would be significant in the frequency domain when the signal function or data is averaged in multiple realizations, based on the fact that white noises have a zero means and would cancel out in sufficiently many realizations.

## 3 Algorithm Implementation and Development

### 3.1 Question 1

The first step is too find the target frequency of the submarine using a set of 3-dimensional signal data collected. To implement this algorithm, we need following developments:

```
Algorithm 1: Finding Center Frequency
    Import data from subdata.mat
    Set up the spatial and Fourier domain
    for $j = 1 : 49$ do
        Extract the record $j$ from subdata
        Reshape to 3-dimensional array and switch to the frequency domain as Un
        Add current record Un to the total sum Unsum
    end for
    Take average of the total sum over 49 realizations as Unave
    Find the center frequency where the averaged $Unave$ has the maximum in the frequency domain
```

1. When set up the spatial and frequency domain, the frequency should be re-scaled to $2\pi/L$ where $L$ is our spatial domain since FFT assumes $2\pi$ periodic signal.
2. MATLAB implementation gives the FFT result in a shifted version such the nonzero frequency is at left and the negative part is at the right. Therefore, fftshift is needed each time we use FFT to shift the zero frequency back to the center as a standard form.
3. To find the center frequency where average value has maximum, we should take absolute value since fft would produce complex value that cannot be compared.

### 3.2 Question 2

With the center frequency in each dimension, now we can create an appropriate filter to denoise the signal data to locate the position of the submarine and find its 24-hour path.

To implement this algorithm, we need following developments:

1. Parameter $\tau$ in the Gaussian filter determines the width of the filter and therefore affects the the filtered results. In this way, different value widths should be tried to find an appropriate filter when finding the path.

**Algorithm 2:** Finding 24-hour path

```
Define a 3-dimensional filter in the frequency domain
for $j = 1 : 49$ do
    Extract the record $j$ from subdata
    Reshape to 3-dimensional array and switch to the frequency domain as Un
    Apply filter on Un to denoise the acoustics data
    Switch filtered data Unf back to the spatial domain as Unft
    Find the coordinate at time $j$ where the acoustic has maximum and store it.
end for
Plot all the coordinates to get the path
```

| Dimension | Center Frequency |
|---|---|
| Kx | 5.3407 |
| Ky | -6.9115 |
| Kz | 2.1991 |

Table 1: Center frequency generated by submarine

| | X-coordinate | Y-coordinate |
|---|---|---|
| 45 | -6.2500 | 2.1875 |
| 46 | -6.2500 | 1.8750 |
| 47 | -5.9375 | 1.5625 |
| 48 | -5.6250 | 1.2500 |
| 49 | -5.0000 | 0.9275 |

Table 2: Center frequency generated by submarine

### 3.3 Question 3

In question 2 we already locate the positions of the submarine at each time points, therefore the latest x/y coordinates can be directly used to track the submarine.

## 4 Computational Results

### 4.1 Q1:Center Frequency

Using Algorithm 1, the center frequency produced by the submarine is shown in Table 1. And the isosurface plot 1 gives the same results.

### 4.2 Q2:Submarine path

Using Algorithm 2, the path of the submarine in the past 24 hours is shown in the plot 2, and the red star denotes and final position of the submarine.

### 4.3 Q3:Tracking position

The $x/y$ coordinates of the submarine can be located using Algorithm 2 and corresponding results from part 2. The latest $5x/y$ coordinates is shown in the Table 2, where the number of order corresponds to the record in the 49 acourstics data over the last 24 hour with half-hour increments in time.

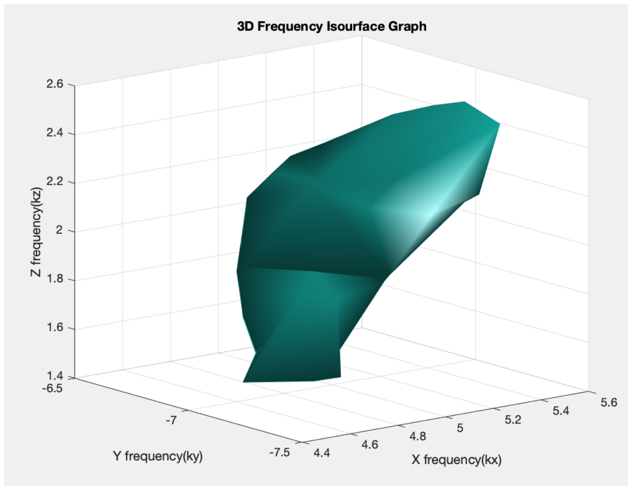

Figure 1: Here is a picture of the isosurface graph of averaged frequency collected.

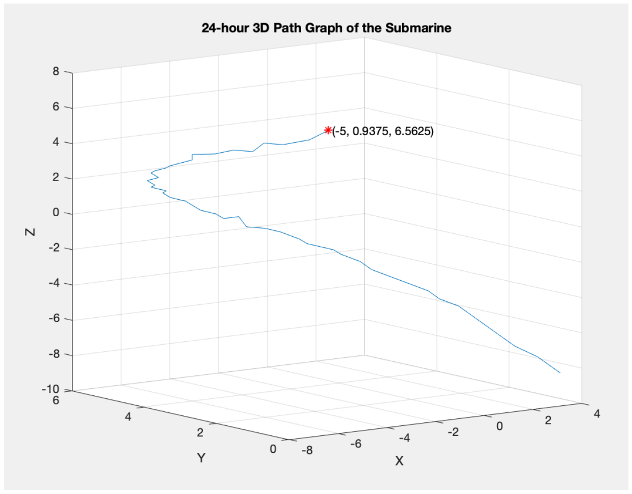

Figure 2: Here is a picture of the 24-hour path graph of the submarine.

## 5 Summary and Conclusions

In conclusion, averaging noisy acoustic data in the frequency domain using FFT, we can find the center frequency produced by the target submarine we want to track. And applying appropriate filters based on the center frequency helps up detect the position of the submarine, so that we can send a P-8 Poseidon subtracking aircraft to the final x/y coordinate $(-5, 0.9275)$ to track the submarine.

## Appendix A MATLAB Functions

MATLAB functions used in the implementations of Algorithms.

- `y = linspace(x1,x2,n)` returns a row vector of `n` evenly spaced points between `x1` and `x2`. In Algorithm 1, this is used to set up the spatial domain of our problem.
- `[X,Y,Z] = meshgrid(x,y,z)` returns the coordinates of a 3-D grid based on the coordinates in the input vectors `x` `y`, and `z`. X is a matrix where rows are formed by `x`, `Y` is a matrix where columns are formed by `y`, and `Z` is a matrix where pages are formed by `z`. In Algorithm 1, this is used to create the 3-dimensional array to store 3-dimensional data.

- `y = fftshift(x)` returns a shifted version of `x` through swapping the left and right halves of `x`. It is used to visualize the fast Fourier transform to a spectrum with 0 frequency at the center. As in Algorithm 1, this is used after each fft to shift 0 frequency back to center as a standard form.

- `y = fftn(x)` applies N-dimensional fast Fourier transform on the N-D array `x`. The output is complex and all nonzero entries are put before the negative ones.

- `y = reshape(X,M,N,P)` returns an `M-by-N-by-P` 3-D array with elements in `X`. In Algorithm 1 and Algorithm 2, the extracted data at each time point needs to be reshaped to a `N-by-N-by-N` 3D array where N is the number of Fourier modes we use.

- `y = abs(x)` returns the absolute value of `x`. If `x` is a complex number, `y` is the complex modulus of `x`. In Algorithm 1 and Algorithm 2, the FFT returns complex number so we need to take complex modulus using `abs()` before finding maximum.

- `[M,I] = max(x,[],'all','linear')` returns maximum entry value `M` in `x` and the linear indices `I` of that maximum value. In Algorithm 1, `max()` is used to find the center frequency where averaged signal data has its maximum in the frequency domain. And in Algorithm 2, the indices of maximum value in the spatial domain is the position of the submarine that we wan to track.

- `y = ifftn(x)` applies N-dimensional inverse fast Fourier transform on `x`. In Algorithm 2, we need to switch the filtered signal back to the spatial domain to find position through doing inverse FFT using this function.

## Appendix B MATLAB Code

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

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

% Set up spatial and Fourier domain
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);

% Average through spectrum
Unsum=zeros(n,n,n);
for j=1:49
    Un(:,:,:)=reshape(subdata(:,j),n,n,n);
    Unsum=Unsum+fftn(Un);
end
Unave=abs(fftshift(Unsum))/49;
[M,I]=max(abs(Unave),[],'all','linear');
ix=Kx(I); iy=Ky(I); iz=Kz(I);

% Plot the surface
figure(1)
isosurface(Kx,Ky,Kz,abs(Unave)/M,0.7), grid on
title('3D Frequency Isourface Graph')
xlabel('X frequency(kx)'); ylabel('Y frequency(ky)'); zlabel('Z frequency(kz)');

% Create filter
tau=0.5;
xk0=ix; yk0=iy; zk0=iz;
filter=exp(-tau*(Kx-xk0).^2).*exp(-tau*(Ky-yk0).^2).*exp(-tau*(Kz-zk0).^2);

% Find coordinate at each time
cor=zeros(3,49);
for j=1:49
    Un(:,:,:)=reshape(subdata(:,j),n,n,n);
    Unt=filter.*fftshift(fftn(Un));
    Untf=ifftn(Unt);
    [Mt,idx]=max(abs(Untf),[],'all','linear');
    cor(1,j)=X(idx);
    cor(2,j)=Y(idx);
    cor(3,j)=Z(idx);
end

% Plot the path
figure(2)
plot3(cor(1,:),cor(2,:),cor(3,:)), grid on, hold on
plot3(cor(1,end),cor(2,end),cor(3,end),'r*')
text(cor(1,end),cor(2,end),cor(3,end),...
[' (' num2str(cor(1,end)) ', ' num2str(cor(2,end)) ', ' num2str(cor(3,end)) ')'])
title('24-hour 3D Path Graph of the Submarine')
xlabel('X'); ylabel('Y'); zlabel('Z');
```