Title: Track a submarine

Author: Wendy Jiang

Abstract: This report aims to record the process of tracking the submarine, that is, of finding its location and moving path. To locate the submarine, data was collected of acoustic frequency over a 24-hour period in half-hour increments. To achieve our goal, we get the average of the spectrum, distinguish the frequency signature which generate by the submarine, filter the data and denoise the data, find the path of the submarine and locate it.

Section I. Introduction and Overview

This report would conclude the theoretical background, algorithm implementation, and computational result, also the plot and conclusion of solving this problem.

To track the submarine, the first step is to denoise the signals. Therefore, it is necessary to filter by averaging the realizations. We keep the effective signals, which are the submarine produced, and base on these signals of frequency and trace back to the realizations to draw the path of the submarine. This is a 3-D model since the submarine under the ocean and the P-8 Poseidon subtracting aircraft detect above the ocean surface, the z value, which is the height would become the opposite direction.

Section II. Theoretical Background

1. The equation of the Fourier Transform, we define the Fourier transform of $f(x)$, written $\hat{f}(k)$:

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

The Fourier transform takes a function space (or time), x, and converts it to a function of frequencies, k.
Therefore, the inverse Fourier transform:

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

The Fourier transform here assumes an infinite domain.

2. Fourier Series given a function $f(x)$ for a limited range of x values, so all frequencies would not fit into the interval we take. So, we need consider $\sin(kx)$ and $\cos(kx)$ with integer k since these are the only frequencies that lead to functions that completely repeat over the interval. Given by infinitely many frequencies $k = 1,2,3,4, ...$, and write $f(x)$ as the sum of sines and cosines:
$f(x) = \frac{a_0}{2} + \sum_{k=1}^{\infty} (a_n \cos(kx) + b_n \sin(kx)),\ x \in [-\pi, \pi]$ (3).
$\frac{a_0}{2}$ is a constant term to up and down shifting the $f(x)$ function.

3. Fourier Coefficients a_k and b_k:

$$a_k = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \cos(kx)\, dx. \quad (4)$$

$$b_k = \frac{1}{\pi}\int_{-\pi}^{\pi} f(x)\sin(kx)\,dx. \quad (5)$$

$$a_0 = \frac{1}{\pi}\int_{-\pi}^{\pi} f(x)dx. \quad (6)$$

a_0 represents the average of $f(x)$ in the interval $[-\pi, \pi]$

4. Since sine and cosines are periodic functions, the function $f(x)$ we use as Fourier series must a periodic function.

5. Sometimes, the Fourier series could be written in terms of exponentials:

$$f(x) = \sum_{k=-\infty}^{\infty} c_k e^{ik\pi/L} \quad (7)$$

$$c_k = \frac{1}{2L}\int_{-L}^{L} f(x)e^{-ik\pi x/L}dx\,, k\ \in Z, x \in [-L, L]. \quad (8)$$

6. Discrete Fourier Transform (DFT): The DFT is just like a Fourier series but truncated at some maximum frequency. We are given a sequence of N values that are function sampled at equally spaced points $\{x_0, x_1, x_2, \dots, x_{N-1}\}$. The discreate Fourier transform is given by $\hat{x}_k = \frac{1}{N}\sum_{n=0}^{N-1} x_n e^{\frac{2\pi ikn}{N}}$, $k = 1,2,3, \dots, N-1$. (9)

7. FFT (Fast Fourier Transform) is able to convert a signal from the time domain to the frequency domain. IFFT (Inverse Fast Fourier Transform) is the inverse of FFT, which can convert a signal from the frequency domain to the time domain.

8. The thinner a function is, the wider its Fourier Transform is. If the function is thinner, there is a high probability that the particle is in a confined area near the peak of the function. Similarly, with the momentum, the more we can localize the position of a particle, the less we know about the momentum.

9. There is more than one method to filter and denoise. Since the noise is white noise and the submarine keeps producing the signals when we detect, averaging the realizations over the frequency domain is the work before we filter the signals.

10. Denoise is a process to remove the undesired frequencies.

11. The center frequency of a filter or channel is a measure of a central frequency between the upper and lower cutoff frequencies.

Section III. Algorithm Implementation and Development

Our goal in this problem is to convert the data between frequency domain and time domain. To achieve our goal, we assume it's a 2π periodic signals in this problem and set the frequency domain k, by using spatial domain L and the re-scale function $2\pi/L$. Then set 3-D grids of X, Y, and Z coordinates then do the transform into frequency domain to get Kx, Ky, and Kz by using the ks, which the k dealt with function ifft.

Function $y = fft(x)$ computes the discrete Fourier Transform (DFT) of x using FFT algorithm. "The FFT functions (fft, fft2, fftn, ifft, ifft2, ifftn) are based on a library called FFTW [1] [2]." (form https://www.mathworks.com/help/matlab/ref/fft.html#buuutyt-11)

Function $x = ifft(y)$ computes the inverse discrete Fourier Transform using FFT algorithm. "The ifft function tests whether the vectors in Y are conjugate symmetric. A vector v is conjugate symmetric when it equals conj(v([1,end:-1:2])). If the vectors in Y are conjugate symmetric, then the inverse transform computation is faster, and the output is real." (from https://www.mathworks.com/help/matlab/ref/ifft.html)

Section IV. Computational Result

- x, y, z are domain discretization. (all 49*1 double)
- X, Y, Z are grids. (64*64*64 double)
- When we set τ = 0.2, the center frequency we get:
  $center\_Kx$ = 5.3407, $center\_Ky$ = -6.9115, $center\_Kz$ = 2.1991;
- Because we have $64 \times 64 \times 64$ data and 49 times detected, we use 49 times filter, and the filter is a $64 \times 64 \times 64$ double set.
- Filter function of 3-D:
  $$e^{(-\tau*(\mathrm{Kx}-\mathrm{center_{Kx}})^2+(Ky-center_{Ky})^2+(Kz-center_{Kz})^2)}$$
- Index of the sub-data to generate coordinates of submarine x, y, and height of the aircraft z (ind_x, ind_y, ind_z): $ind_x = 36$, $ind_y = 17$, $ind_z = 49$;

Section V. Summary and Conclusions

Figure 1: The path of the submarine (same plot in two different perspectives)

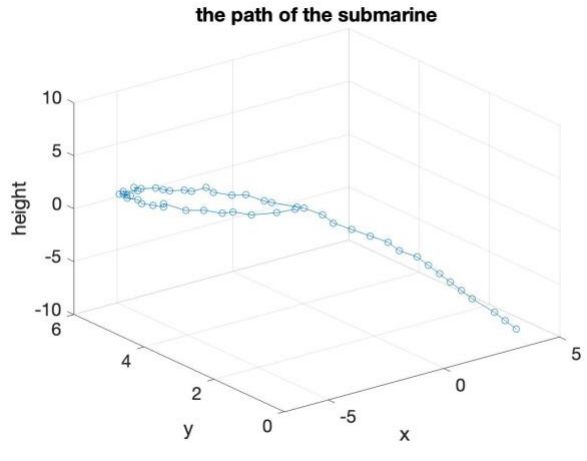

Figure 1

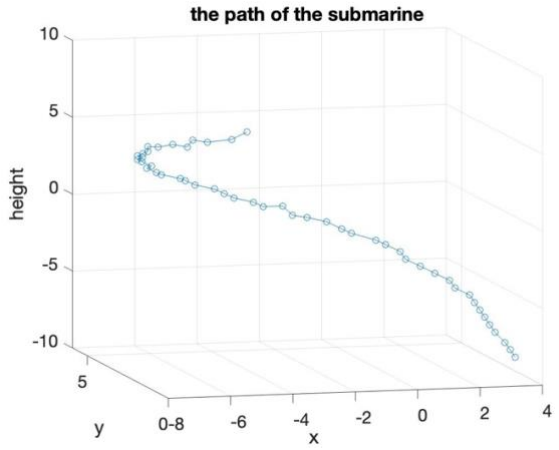

Figure 2

Table of x and y coordinates: (separate into two lines)

x	y	x	y
3.1250	0.0000	-2.8125	5.9375
3.1250	0.3125	-3.1250	5.9375
3.1250	0.6250	-3.4375	5.9375
3.1250	1.2500	-4.0625	5.9375
3.1250	1.5625	-4.3750	5.9375
3.1250	1.8750	-4.6875	5.6250
3.1250	2.1875	-5.3125	5.6250
3.1250	2.5000	-5.6250	5.3125
3.1250	2.8125	-5.9375	5.3125
2.8125	3.1250	-5.9375	5.0000
2.8125	3.4375	-6.2500	5.0000
2.5000	3.7500	-6.5625	4.6875
2.1875	4.0625	-6.5625	4.3750
1.8750	4.3750	-6.8750	4.0625
1.8750	4.6875	-6.8750	3.7500
1.5625	5.0000	-6.8750	3.4375
1.2500	5.0000	-6.8750	3.4375
0.6250	5.3125	-6.8750	2.8125
0.3125	5.3125	-6.5625	2.5000
0.0000	5.6250	-6.2500	2.1875
-0.6250	5.6250	-6.2500	1.8750
-0.9375	5.9375	-5.9375	1.5625
-1.2500	5.9375	-5.3125	1.2500
-1.8750	5.9375	-5.0000	0.9375
-2.1875	5.9375		

Conclusion: In figure 1, we could briefly get the area the submarine runs. Figure 2 shows that we when detected, the depth underwater actually changed. The trajectory of a submarine is similar to a spiral. The submarine neared the surface of ocean as we check the table and compared the x, y coordinates into the plot Figure 2.

Appendix A. MATLAB functions used and brief implementation explanation

1. meshgrid(): create a 3-D grid coordinates and separate the data into grid.
2. fftshift(): rearrange a Fourier Transform x and shifting zero frequency component to the center of the array.
3. ifftshift(): rearrange a zero-frequency-shifted Fourier Transform back to the original transform output.
4. linespace($x1, x2, n$): this functions is used to generate points between two values.
5. fftn: returns the 3-D dimensions array using Fourier transform algorithm.
6. ifftn(): returns the discrete inverse Fourier transform of an 3D array using FFT algorithm.

7. [row, col, page]= ind2sub(size(A),ind): convert a linear index of a 3-D array to a subscript index. “ind” means corresponding to which index of the element of the array.

Appendix B. MATLAB Code

```
clear all; close all; clc
load subdata.mat

L = 10; % spatial domain
n = 64; % Fourier modes
x2 = linspace(-L,L,n+1); % domain discretization
x = x2(1:n); % only the first n points (periodicity)
y = x; % set y
z = x; % set z (since the 3-D dimension)

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);

% Reshape the data
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

% averaging of the spectrum
Un = zeros(n,n,n);
Utnave = zeros(n,n,n);
for j = 1:49
    Un(:,:,:)=reshape(subdata(:,j),n,n,n);
    Utn = fftshift(fftn(reshape(subdata(:,j),n,n,n)));
    Utnave = Utnave + Utn;
end
Utave = Utnave/49;

% determine the frequency signature (center frequency)
[m,ind] = max(Utave(:));
[ind_x,ind_y,ind_z] = ind2sub([n,n,n],ind)
center_Kx = Kx(ind_x,ind_y,ind_z);
center_Ky = Ky(ind_x,ind_y,ind_z);
center_Kz = Kz(ind_x,ind_y,ind_z);

% filter the signal
tau = 0.2; % center of the window
filter = exp(-tau*((Kx - center_Kx).^2 +(Ky - center_Ky).^2 +(Kz -
center_Kz).^2)); % define the filter
```

```
% set original x,y,and z
x = zeros(49,1);
y = zeros(49,1);
z = zeros(49,1);
% use the signal of subdata and rearrange the data
for j = 1:49
    Un(:,:,:)=reshape(subdata(:,j),n,n,n);
    Utn = fftshift(fftn(reshape(subdata(:,j),n,n,n)));
    unf = filter.*Utn; % build the function with filter
    unfshift = ifftshift(unf);
    un = ifftn(unfshift); % rearrange its order and get into the time domain
    [m,ind] = max(un(:)); % find the max of the signals as the target
    [ind_x,ind_y,ind_z] = ind2sub([n,n,n],ind); % get index in 3 dimension to
use target ind
    x(j,1) = X(ind_x,ind_y,ind_z);
    y(j,1) = Y(ind_x,ind_y,ind_z);
    z(j,1) = Z(ind_x,ind_y,ind_z);
end
figure(3);
% plot the path of the submarine
plot3(x,y,z,'o-'); grid on, drawnow;
xlabel('x');
ylabel('y');
zlabel('height');
title('the path of the submarine');
set(gca,'Fontsize',16);

% record the data of x and y in a table to locate the submarine
T = table(x,y);
writetable(T,'location_table.xls');
```