Signal Processing Techniques for Sonar Submarine Localization

Ben Francis

January 27, 2020

Abstract

Signal processing techniques are applied to hydro-acoustical data to determine the path of an unknown submarine in Puget Sound. Frequency spectrum averaging is used to remove white noise and determine the signature frequency used for its active sonar navigation. A Gaussian filter is then used to filter around this frequency and determine the path of the submarine. This three-dimensional trajectory is projected down to a two-dimensional path for an antisubmarine aircraft to follow, with a final position of $(-5, 0.9375)$.

1 Introduction and Overview

An unknown submarine has been identified moving within Puget Sound. For national security reasons, it is of paramount importance to identify the location of this submarine. Upon localizing the trajectory of the submarine, a P-8A Poseidon aircraft will be deployed to further maintain surveillance on the vessel. The submarine uses a signature frequency of sonar for navigation allowing passive sonar measurements from within Puget Sound to identify the location of the vessel. Using already in-place equipment, 24 hours of data were recorded in half-hour increments. The purpose of this report is to detail the signal processing techniques used in analyzing this data, and to present the results of the approximate trajectory of the submarine.

1.1 Submarine Navigation

Submarines constitute an important part of all modern navies, due to their ability to turn water into a three-dimensional medium, whereas most ships are constrained to the two-dimensional surface. Being able to move at arbitrary depth allows for the vehicle to avoid radar and satellite navigation. While this confers many benefits, one downside is difficulty in navigation. Since light from the surface rapidly diminishes beyond 200 meters, navigation beyond periscope-depth is conducted using other means.[1] Submarines generally use GPS for positioning. However, if they enter a compact area such as Puget Sound, they will need to use active sonar for navigation. This presents an opportunity for locating them, if passive sonar recordings are made in the nearby vicinity of the vessel.[3]

1.1.1 Sonar Measurements

Passive sonar listens for changes in water pressure, while active sonar both emits and listens to changes in water pressure. Passive measurements have been recorded from different regions in Puget Sound and processed to give an estimate of water pressure at each point, within a cube of volume. A cube of 64x64x64 data points are given, with unknown units. The data constitutes a 10x10x10 unit region in the spatial domain. Since the unknown submarine is using active sonar navigation, there should be a characteristic spatial frequency within water pressure measurements centered around the position of the vessel. These will change at each time point.

2 Theoretical Background

2.1 Time and Frequency Signals

Analog data is recorded in the space and time domain. In order to use computational techniques for signal processing, this signal is then digitized. Much information can be obtained by decomposing the signal into its frequency components. The spatial equivalent of frequency is often called *wavenumber*, though in practice the terms can often be used interchangeably. The mathematical framework for this decomposition is the representation of a function as a Fourier series, which allows a function $f(x)$ to be represented by a trigonometric series of sines and cosines [2]:

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

Similarly, it can be expressed on the arbitrary domain of length $2L$ by the expansion:

$$f(x) = \sum_{-\infty}^{\infty} \left(c_n e^{in\pi x/L} \right) \quad x \in (-L, L] \tag{2}$$

Each of the coefficients c_n then determines how much the nth frequency contributes to the function $f(x)$. It is often useful to transform the function $f(x)$ to a frequency domain k, where the output $F(k)$ is now equivalent to the amount each frequency contributes to the original function $f(x)$. k is called the wave number, and is now allowed to be complex. This is called the Fourier transform and is represented by:

$$F(k) = \int_{-\infty}^{\infty} e^{-ikx} f(x) dx \tag{3}$$

The original function can be recovered by the Inverse Fourier Transform:

$$f(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} e^{ikx} F(k) dk \tag{4}$$

The intuition behind the purpose of this transform is that signals have behaviors that exist on long time scales and on short time scales, such as the difference between climate change and weather change. In the case of the data given, these will be spatial frequencies. Fish may cause higher frequency effects while boats may cause lower frequency effects.

There are many applications for this transformation, which is a foundational element of signal processing. In this application, there are two primary purposes: removing white noise using signal averaging, and Gaussian filtering about the signature frequency.

2.2 Signal Processing

2.2.1 Signal Averaging

White noise is a ubiquitous form of noise in signals, described by random signal having equal intensity at all frequencies. [3] White noise is then more naturally handled in the frequency domain, where it has a uniform distribution. The primary mechanism for removing white noise is by averaging the frequency-domain signal. Since the white noise is random, it will average to zero with a large number of samples, while signals other than white noise will not average to zero.

2.2.2 Gaussian Filtering

Second, it is known that the submarine uses a signature frequency signal for sonar navigation. While this signal is unknown, if the spatial acoustic measurements are examined in the frequency domain at each time step, then the signal should show up as a spike about a particular frequency, after white noise is removed.

The central frequency can be algorithmically determined by finding the frequency k with the highest $F(k)$ amplitude, after averaging.

Since we are only interested in this fequency, we can apply a filter about this frequency. A filter is a process that removes unwanted parts of a signal. Since the only frequency of interest here is the signature frequency, all of the other frequencies can be removed. A Gaussian filter diminishes frequencies about a central frequency in the shape of a Gaussian distribution centered about the central frequency. One benefit of this filter is that it does not sharply cut off all other frequencies, making the filter more robust to error when estimating the signature frequency.

2.3 Fourier Transforms

While the mathematics of the Fourier transform has been understood since the 1800's, its implementation in computational systems has additional requirements, namely (1) discrete data and (2) non-infinite data. Data is handled as discrete primarily to maintain robust signal communication. While mathematical techniques often use infinite spaces, physical data never contain any infinities. As such, the spatial domain is defined to be between $-L$ and L, where $L = 10$. This also affects the wavenumbers k under consideration, which equally cannot be infinite. The wave number assumes 2π periodic signals, so the wavenumbers must be scaled to $\frac{2\pi}{2L}$. The Fourier Transform assumes periodic boundary conditions.

These requirements results in a change from the Continuous Fourier Transform to the Discrete Fourier Transform and its inverse:

$$F(k) = \sum_{n=1}^{N} f(n) \exp[-i\frac{2\pi(k-1)}{N}(n-1)] \quad 1 \leq k \leq N \tag{5}$$

$$f(x) = \frac{1}{N}\sum_{k=1}^{N} F(k) \exp[i\frac{2\pi(k-1)}{N}(n-1)] \quad 1 \leq n \leq N \tag{6}$$

This provides a need to determine a discrete frequency domain, determined by the number of wavenumber points k used. There is a further constraint on this resolution, in that it must be a power of 2. The reason for this is so that the Fast Fourier Transform (FFT) can be used, which will be examined in the next section.

Since the hydro-acoustic waves of interest are spatial waves, the spatial domain will be transformed. At each time step, this gives a three-dimensional grid of wavenumbers, each with an associated value determining its contribution to the spatial signal.

2.3.1 Fast Fourier Transform (FFT)

The brute force algorithm for computing the Fast Fourier transform has $O(n^2)$ computational complexity [2]. In the 1960's a method for computing the Fourier Transform was popularized that allowed for $O(n \log n)$ computational complexity. However, this procedure has some important caveats.

1. The number of data points in the discretized spatial domain must be a power of 2.
2. The frequency domain in each dimension will not be sequential, but rather be $[\frac{k}{2}|...|k|0|...|\frac{k}{2}-1]$. It is "cut in half" and the back half put in front of the front half. This is called the FFT *shift*.
3. The value associated with each frequency along each dimension will be alternating in sign.

The first need is taken care of in the data collection process, since there are 64 data points in the spatial domain. This is needed because the algorithm uses a power-of-two divide and conquer algorithm. [2] Here, data is presented in a 64x64x64 meshgrid, so the constraint is satisfied. The second need does not affect this analysis, though any visualization of the frequency domain would need to take it into account. The shift is undone by the inverse FFT. The third effect does affect the need to find the maximum frequency, since the maximum frequency may be sign-flipped and actually be a minimum. To mitigate this effect, the absolute value of the frequency domain signal is taken prior to finding its maximum.

3 Algorithm Implementation and Development

Import data from `subdata.mat`
Define spatial and frequency domains as 3D meshgrids, with frequency domain shifted in each dimension
Define `avg` variable
for $j = 1:49$ **do**
  Reshape `subdata` at time j to 3D coordinates
  Perform 3D FFT on 3D subdata
  Add result to `avg`
end for
`avg` $\leftarrow$ `avg`/49
Find the 3D indices corresponding to maximum of absolute value of `avg`
Use 3D indices to determine 3D components of signature frequency
for $j = 1:49$ **do**
  Perform 3D FFT on reshaped 3D subdata
  Multiply frequency domain signal by Gaussian filter centered at signature frequency
  Perform 3D Inverse FFT on data
  Determine coordinates at current time step with max value
end for
Plot submarine coordinates at all times as a trajectory
Plot 2D projection of submarine coordinates for aircraft to follow

---

3.1 Algorithm Development

With the theoretical background in place to understand the elements of the submarine localization algorithm, it will now be discussed in more detail. Specifically, the rationale for certain steps will be described and details about the algorithm's instantiation in MATLAB will be reviewed.

3.1.1 Defining the Spatial and Spectral Domain

64 data points are given in the spatial domain at each time. These give a resolution to the spatial domain, which is given as having height, width, and depth of 10. An identical discrete interval is made for each of the three dimensions, and then a meshgrid is made by combining all of these dimensions. This is done for both the spatial and frequency domains.

While the Fast Fourier Transform can operate on the data without any presupposed frequency domain, it is useful to predefine the frequency domain so that results obtained on the Fourier transformed function can be interpreted as specific frequencies. This will be useful when determining the signature frequency. Since the FFT gives an $F(k)$ for each point in the spatial domain, the size of the three-dimensional meshgrid should also be 64x64x64. The FFT algorithm assumes a 2π periodic function, so the wavenumbers are scaled as $\frac{2\pi}{2L}$. This scaling tells us how to construct the wavenumbers from the spatial domain. Because of the FFT shift, each frequency axis should start at 0, go up to $(\frac{n}{2}-1)\frac{2\pi}{2L}$, then jump to $-\frac{n}{2}\frac{2\pi}{2L}$, and finally increment up to $-\frac{2\pi}{2L}$. The $\frac{n}{2}-1$ comes from the fact the FFT assumes periodic boundary conditions. The last point is the same as the first point, so it is not needed. This is done for each axis to define a `kx`, `ky`, and `kz`. A three-dimensional meshgrid is then created out of these three axes.

3.1.2 Finding the Signature Frequency Using Signal Averaging

The first step in the signal processing is to find the three-dimensional signature wavenumbers, where there is one signature wavenumber in each dimension. This can be done by averaging the signal in the frequency domain, which gradually removes the white noise. To do so, an empty 64x64x64 array called `Uave` is created. Then each time step is looped through, the Fast Fourier Transform of each signal is taken, and then added to `Uave`. After stepping through all time steps, `Uave` is divided by the total number of time steps, 49.

This should significantly diminish the effects of the white noise in the acoustical data. There should then be a large peak at one point in the wavenumber grid, whose three-dimensional coordinates give the signature wavenumbers. To find this maximum, the `max` function is used, which finds the maximum value in an array, and can return both the maximum value and its index. Only the index is needed here, since only the corresponding wavenumbers are needed.

However, the index returned uses MATLAB's index scheme that uniquely identifies a point in an n-dimensional array by one number. In order to find the signature wavenumbers, subscripts of the maximum value will be needed, which can then be compared with the constructed frequency domain to find the wavenumbers. To convert between the single index number to the three needed subscripts, the `ind2sub` function is used. From the frequency meshgrid, these can be used to identify the signature wavenumbers in each dimension, `sign_fx`, `sign_fy`, `sign_fz`.

3.1.3 Tracking the Submarine Using Signal Filtering

This has now identified the signature frequency of the submarine's active sonar system. To locate the submarine at each time step, every other frequency should be filtered out, leaving just the water pressure effects from the submarine's sonar. In practice, the Gaussian filter acts to focus upon a certain center frequency and diminish frequencies further away from the center frequency. Now a three-dimensional Gaussian filter is used, centered about the signature wavenumbers in each dimension. For each time step, the signal is again Fast Fourier Transformed, and then filtered using the signature frequency-centered Gaussian filter. The signal is then Inverse Fourier Transformed to return to the spatial domain. This leaves a spatial signal with signals mostly due to the signature frequency.

The submarine should correspond to the maximum point of these signals. The strongest effect from the signature frequency should be centered around the submarine, where it is emitted prior to dissipating. As such, the `max` function is used again to find the index corresponding to the maximum value. This index is again transformed to subscripts using `ind2sub`, which is used with the spatial meshgrid to determine the approximate location of the submarine at each point in time.

At each time step, this gives one three-dimensional position coordinate. For all of the time steps, this gives 49 coordinates that represent the trajectory of the submarine over time. This can be plotted in three-dimensional space with the `plot3` function. The aircraft only needs the xy projection of these, since it will fly over the submarine, and so the z dimension can be discarded.

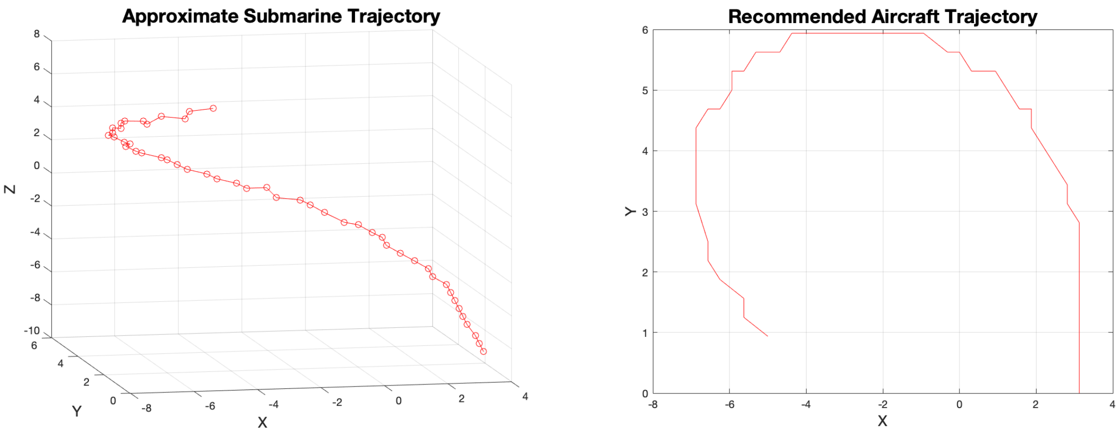
4 Computational Results

Figure 1A gives the estimated trajectory of the submarine. Figure 1B gives the recommended trajectory of the aircraft so as to follow the submarine overhead for surveillance, if the data was recorded in live time. The units were not given in the data, so they are also not present in the results. The final position of the submarine was found to be $(-5, 0.9375)$, which is where the aircraft should be sent.

5 Summary and Conclusions

Signal processing techniques such as signal averaging in the time domain and Gaussian filtering were used to identify the path of a target emitting a signature frequency from data with white noise. This path was then projected down to two dimensions to provide a corresponding trajectory for the antisubmarine aircraft to follow, allowing surveillance and security in Puget Sound. From the latest measurements, the position of the submarine was estimated as $(-5, 0.9375)$, which is where the aircraft is recommended to be sent. These signal processing techniques may also be rapidly applied to any new data to provide an update on this estimate.

Figure 1: Submarine and Aircraft Trajectories



(a) Estimated trajectory of the unkonwn submarine in Puget Sound.

(b) Flight path for P-8A Poseidon Aircraft above Puget Sound corresponding to submarine path.

References

[1] *How far does light travel in the ocean?* URL: `https://oceanexplorer.noaa.gov/facts/light-distributed.html`.

[2] Jose Nathan Kutz. *Data-driven modeling & scientific computation: methods for complex systems & big data*. Oxford University Press, 2013.

[3] *Submarine Navigation*. URL: `https://en.wikipedia.org/wiki/Submarine_navigation`.

Appendix A MATLAB Functions

- `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.
- `Y = fftn(X)` is the n-dimensional version of the Fast Fourier Transform. `X` is data in the spatial domain and `Y` is data in the frequency domain. There is a shift and sign flip associated with the FFT algorithm in each dimension.
- `X = ifftn(Y)` is the n-dimensional version of the Inverse Fast Fourier Transform. `X` is data in the spatial domain and `Y` is data in the frequency domain. It expects `Y` to have the shift and sign flip associated with the FFT algorithm.
- `M = max(A)` finds the maximum value of an array `A`. Importantly, if a second output is provided, in the form `[max, ind] = max(input)`, then the `ind` returns the index of the maximum value. However, only one value is returned for the index. If `input` has a dimension higher than one, the index is returned using the MATLAB convention, not as multidimensional subscripts. If only the index is needed, a `~` can be used as a placeholder for the `max`.
- `[I1,I2,...,In] = ind2sub(sz,ind)` takes the size of an n-dimensional array and an index value and returns n subscripts. This is used in tandem with `max` to determine the signature wavenumbers. The index given as input is a one-dimensional way of describing the position in an n-dimensional array, according to MATLAB convention.
- `plot3(X,Y,Z)` plots coordinates in three-dimensional space.

Appendix B MATLAB Code

```
close all; clear all; clc;
load('subdata.mat')

L = 10; n = 64;
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);
Uave = zeros(n,n,n);

for j=1:49
    Un(:,:,:)=reshape(subdata(:,j),n,n,n);
    Utn = fftn(Un);
    Uave = Uave + Utn;
end

Uave = Uave / 49;
[~, ind] = max(abs(Uave(:)));
[I,J,K] = ind2sub([n,n,n], ind);
sign_fx = Kx(I,J,K); sign_fy = Ky(I,J,K); sign_fz = Kz(I,J,K);

filter = exp(-1 * ((Kx-sign_fx).^2 + (Ky-sign_fy).^2 + (Kz-sign_fz).^2));
submarine = zeros(3,49);
for j=1:49
    Un=reshape(subdata(:,j),n,n,n);
    Utn_f = fftn(Un) .* filter;
    Un_f = ifftn(Utn_f);

    [~, ind] = max(abs(Un_f(:)));
    [submarine_x, submarine_y, submarine_z] = ind2sub([n n n], ind);
    submarine(1,j) = X(submarine_x, submarine_y, submarine_z);
    submarine(2,j) = Y(submarine_x, submarine_y, submarine_z);
    submarine(3,j) = Z(submarine_x, submarine_y, submarine_z);
end

figure();
plot3(submarine(1,:), submarine(2,:), submarine(3,:),'r'), grid on;
title('Approximate Submarine Trajectory','FontSize',18), xlabel('X','FontSize',14), ylabel('Y','FontSiz
aircraft_trajectory = submarine(1:2,:);
plot(aircraft_trajectory(1,:), aircraft_trajectory(2,:),'r'), grid on;
title('Recommended Aircraft Trajectory','FontSize',18), xlabel('X','FontSize',14), ylabel('Y','FontSize
```

Listing 1: MATLAB code, available at https://github.com/benfrancis314/AMATH582/blob/main/hw1.m