Coding Project 1: Detecting objects through frequency signatures

Wietse Vaes

Abstract

Help! A Kraken is under the Climate Pledge Arena during a match. Maybe we can use this to our advantage! In this short paper a method will be developed to find the Kraken under the ice. The vibrations can be picked up by our radar detector, but it has a lot of white noise. Using the Fourier transform to go from spatial domain to frequency domain we will be able to center on the frequencies caused by the kraken by diminishing the importance of the white noise added through time through spectral averaging. After finding the frequencies cause by the Kraken, we will use spectral filtering in order to find the exact location of the Kraken in the spatial domain.

1 Introduction

Radio detection and ranging, radar, is something that has changed the life of every human being, direct or indirect, and has fundamentally made technology to what it is today. In the early days of radar, around the late 19th century, Heinrich Hertz began experimenting with radio waves. He soon found out that metallic objects reflected them. Christian Hülsmeyer then (20th century) began using this characteristic to build simple detection systems whereas the British then began further development.

The golden age of radar development came during the second world war. This as it was extremely useful to find enemy ships and tanks. Radar detection is still used today in many applications such as, navigation of ships and aircrafts, environment mapping, the military even space! On of the problems, however, is that radar detections never only pick up what we want them to

pick up. They often bring noise with them, one in particular is white noise.

Because of this white noise, we would like to process the data coming in from the radar detector. In the following sections we will give a small theoretical background of how we will diminish the effects of the white noise. We will, namely, use the Fast Fourier transform, and spectral averaging to locate the frequencies caused by the Kraken, thereafter we will use spectral filtering on the transformed data in order to almost negate the white noise. After explaining the implementation of these techniques, we will be able to locate the Kraken and gives their path.

2 Theoretical Background

In order for us to understand what is being detected and given to us, a computer must process the measured data. Using the techniques developed in this section, such as the Fourier transformation and spectral filtering, we hope to go from relatively raw, noisy data to an understanding of what is happening and what we want to find out, the location of the kraken in this case.

2.1 The Fourier Series

One of the key and efficient methods of analyzing data is through the frequencies and their amplitudes of said data. In order to get these frequencies and amplitudes we will be using the Fourier transformation. This will be a transformation on a continuous function, but in order to get a better feel for what this does, we will first look at the Fourier Series. Say that we have a function $f : [-\pi, \pi] \to \mathbb{R}$, it has been proven that if this function satisfies a couple of conditions, such as continuity, it can be perfectly described as a sum of sines and cosines:

$$\begin{aligned} f(x) &= \frac{a_0}{2} + \sum_{n \in \mathbb{Z}_0} a_n \cos(nx) + b_n \sin(nx) \\ &= \sum_{n \in \mathbb{N}} c_n e^{inx}. \end{aligned}$$

Here we see what the frequencies and their corresponding amplitudes are that are used in this case, namely, we have the frequencies $n \in \mathbb{N}$ and their

corresponding amplitudes a_n and b_n or, together, c_n. Note that in one sum we chose $\mathbb{Z}$ and the other we chose $\mathbb{N}$. We could look at this change by defining c_n using a_n and b_n:

$$c_n = \begin{cases} -b_n i & n > 0 \\ a_{-n} & n \leq 0 \end{cases}.$$

Here we also must have that a_n and b_n are always real and zero when n is positive, respectively, negative in the second summation. This is one way of seeing the link between both summation, this is however not how c_n will usually be defined.

The frequencies are known, however, the amplitudes are what we want to find. This can easily be found using the orthogonality of the basis $(1, \sin(x), \sin(2x), \ldots, \cos(x), \cos(2x), \ldots)$, over inproduct $\langle u, v\rangle = \int_{-\pi}^{\pi} uvdx$, for these functions. Note also that

$$\int_{-\pi}^{\pi} \cos(nx)^2 dx = \pi, \text{ and } \int_{-\pi}^{\pi} \sin(nx)^2 dx = \pi.$$

Therefore we have that, for $n \geq 0$,

$$a_n = \frac{1}{\pi}\int_{-\pi}^{\pi} f(x)\cos(nx)dx,$$

and

$$b_n = \frac{1}{\pi}\int_{-\pi}^{\pi} f(x)\sin(nx)dx.$$

Lastly Note that one should not think that they are limited to approximating functions only over the domain $[-\pi, \pi]$. Other, bounded, domains can be achieved using transformations over the variable.

2.2 The Fourier Transform

From this description, the Fourier series, of f we would like to expand to a continuous spectrum of frequencies and an unbounded domain of the function: $f : \mathbb{R} \to \mathbb{R}$. In order to achieve this we will define the Fourier Transform and it's inverse. The Fourier transform can be given as

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

and its inverse is given by

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

Here $F(k)$ will live on the frequency domain and $f(x)$ on the spatial domain. Looking at the Fourier Series as some sort of discrete inverse Fourier Transform, it is easy to notice why we now use an integral instead of a sum. We would like to emphasize that the Fourier Series is not immediately related to the discrete Fourier Transform, but we only look at it like that in a hand wavy way. Noticing these similarities we get a feeling for what $F(k)$ is: the amplitudes of the frequencies k.

It is also important to show that these transformations are indeed each other's inverse. The proof of this is not as easy as it may seems, trying to prove it gives us,

$$\begin{aligned} \mathcal{F}\left(\mathcal{F}^{-1}(F(K))[x]\right)(k) &= \frac{1}{2\pi} \int_{-\infty}^{\infty} e^{-ikx} \int_{-\infty}^{\infty} e^{ilx} F(l) dl dx \\ &= \frac{1}{2\pi} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} e^{ix(k-l)} F(l) dl dx \end{aligned}$$

Both directions of proving that they are each other's inverse, would depend on proving that

$$f(x) = \frac{1}{2\pi} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} e^{ik(x-s)} f(s) ds dk,$$

which is a lot more difficult than it seems. This last equality is the Fourier integral theorem.

These formulas can be used for any function f which satisfies certain condition, such as it being in the L_2 space. More importantly, we have that f is a continuous function. This results in two major problems: real-life data is almost never continuous and computers are unable to comprehend a continuous function. This means that, most of the time, a function would approximate the integral as it is unable to compute the actual, "continuous", integral. Therefore the computer would use the Discrete Fourier Transform which has an operation count of $\mathcal{O}(N^2)$ where N are the amount of data points.

2.3 The Fast Fourier Transform

The order of operation count of the DFT method seems good, but it is possible to do better. That is why the Fast Fourier Transform has been developed. It lowers the operation count from $\mathcal{O}(N^2)$ to $\mathcal{O}(N\log(N))$ if N, the amount of data points, is large enough. Using the MATLAB function `fft` or `fftn`, for multidimensional data sets, one can calculate the frequency domain. One must however note that the frequencies are shifted here, one can shift them back using `fftshift` and `ifftshift`. Thereafter one can go back to the spatial domain using `ifft` or `ifftn`.

2.4 Spectral Averaging

A direct advantage of looking at the frequency domain instead of the spatial domain is that we can use it to negate the consequences of noise, particularly white noise. A definition of white noise, from Oxford languages, goes as follows:

"noise containing many frequencies with equal intensities."

Thus, white noise on the spatial domain will change it with many different frequencies, these frequencies are seen in the frequency domain as little spikes in the amplitudes, however they have equal intensities. If we thus have many data sets of a solitary wave over a period of time and all with random white noises, we have a bunch of random frequencies with a little higher amplitude, these frequencies might sometimes be the same, but we will always have those frequencies from our non-distorted solitary wave. Therefore, if we were to take the average of our function on the frequency domain, the frequency of our non-distorted wave will stand out as it is always present in all times. In conclusion: this spectral averaging will negate the effects of the white noise and make the important frequencies standout.

2.5 Spectral Filtering

The spectral averaging is able to tell us where the important frequencies are when working with white noise. Now that we know where the important frequencies are centered around, which stay the same over time since we are dealing with a solitary wave, we want to focus on these. One way of doing this is through filtering the frequencies domain, otherwise known as spectral

filtering. There are multiple effective ways of filtering the frequency domain. The filter used will depend on the goal of filtering the date.

The first filter that would comes to mind is a discontinuous filter on the frequency domain that ignores certain frequencies, such as,

$$\mathcal{H}(k) = \begin{cases} 1 & k \in I, \\ 0 & \text{elsewhere.} \end{cases}$$

This filter clearly does as we would want. If we were to multiply this function G with $F(k)$ over the frequency domain, it will keep the amplitudes of the frequencies in I and set the other amplitudes to zero.

From here one might want to, instead of a discontinuous filter, use a continuous filter. Instead of going from 1 to 0 abruptly, we can go there smoothly. By smoothing frequencies, we also smooth out the noise. Since the frequencies of the noise might be close to the frequencies of the important data, we would not like to use a discontinuous filter. A lot of different ways to smoothly convert from one to another exist, but one that is very efficient for frequency filtering is the Gaussian filter:

$$\mathcal{G}(k) = e^{-\tau \|k-k_0\|_2^2},$$

with τ the bandwidth of the filter, k the multidimensional wavenumber or frequencies and k_0 the frequency around which we would like to emphasize importance in filtering. Since we choose importance as being of the highest amplitude, this filter is a high-pass filter. The Gaussian filter has certain advantages to it, such as the Fourier Transform of a Gaussian staying a Gaussian. This comes in handy along with the convolution theorem.

3 Implementation

Now that the theory has been explained, it will be implemented into MATLAB in order to go from "raw" data to understandable results, in this case the route of the Kraken. The data is given as a $64^3 \times 49$ matrix where the rows represent the realizations recorded under the ice and the columns represent the time. The data exists over the spatial domain $[10, -10[^3$ and time $[0, 60]$ in minutes. It is recorded every 1.25 minutes. First we start by loading in this data, discretize the one dimensional spatial $[-10, 10[$ and frequency

domain over 64 nodes and then converting it the three dimensional domains using the `meshgrid()` function.

Thereafter we want to find out which frequencies are created by the Kraken and which are just noise, this is where the spectral averaging over time comes into play. Before doing that, the data needs to be reshaped into a more comprehensible tensor, we do this by reshaping the original data at time t_i, with $i = 1, \ldots, 49$, into a $64 \times 64 \times 64$ tensor. Afterwards the three-dimensional Fourier transform `ffn()` can be used on this data so that we can sum it to the summed data of all previous times. In order to average it, we divide it by 49, but this is not necessary as the amplitude of the dominant, with the largest amplitude, frequency will stay dominant nonetheless. The dominant frequency can easily be found by using the `max` function.

Now that we know which frequency is dominant, we know around what we would like to filter the data with a smooth filter. Say we would only keep the dominant frequency, or a region around it, and set the amplitudes of the rest to zero. We would just get a sine or cosine-like function out of the data. This would thus not accurately represent the location of the Kraken. This is one of the reason why we use a smooth filter, namely, the Gaussian. Here we use a bandwidth of $\frac{1}{L}$. We apply this filter to the Fourier transformed (`ffn()`) data through point wise multiplication at each time separately, hereafter we use the inverse Fourier transform (`iffn()`) of the filtered data. Note that shifting was never necessary. This gives us the original spatial data, but with the noise dampened. Therefore, we can now easily identify the approximate location of the Kraken by the point where the data is the largest (in absolute value). These points are put into a vector and returned so that we can plot the route of the Kraken over time.

4 Results

Applying the implementation of the last section onto the given data set gives us the following positions of the Kraken over time

	0	1.25	2.5	3.75	5	6.25	7.5	8.75	10	11.25	12.5	13.75	15
x	0	0.3125	0.6250	1.2500	1.5625	1.8750	2.1875	2.5000	2.8125	3.1250	3.4375	3.7500	4.062
y	3.1250	3.1250	3.1250	3.1250	3.1250	3.1250	3.1250	3.1250	3.1250	2.8125	2.8125	2.5000	2.187
z	-8.1250	-7.8125	-7.5000	-7.1875	-6.8750	-6.5625	-6.2500	-5.9375	-5.6250	-5.3125	-5.0000	-4.6875	-4.375

Here we see that the x coordinates first increases to $x = 5$ until 3/4 of

	16.25	17.5	18.75	20	21.25	22.5	23.75	25	26.25	27.5	28.75	30
x	4.3750	4.6875	5.0000	5.0000	5.3125	5.3125	5.6250	5.6250	5.6250	5.9375	5.9375	5.9375
y	1.8750	1.8750	1.5625	0.9375	0.6250	0.3125	0	-0.6250	-0.9375	-1.2500	-1.8750	-2.1875
z	-4.0625	-3.7500	-3.4375	-3.1250	-2.8125	-2.5000	-2.1875	-1.8750	-1.8750	-1.2500	-1.2500	-0.9375

	31.25	32.5	33.75	35	36.25	37.5	38.75	40	41.25	42.5	43.75	45
x	5.9375	5.9375	5.9375	5.9375	5.9375	5.6250	5.6250	5.3125	5.3125	5.0000	5.0000	4.6875
y	-2.8125	-3.1250	-3.4375	-4.0625	-4.3750	-4.6875	-5.3125	-5.6250	-5.9375	-5.9375	-6.2500	-6.5625
z	-0.6250	-0.3125	0	0.3125	0.6250	0.9375	1.2500	1.5625	1.8750	2.1875	2.5000	2.8125

	46.25	47.5	48.75	50	51.25	52.5	53.75	55	56.25	57.5	58.75	60
x	4.3750	4.0625	3.7500	3.4375	3.4375	2.8125	2.5000	2.1875	1.8750	1.5625	1.2500	0.6250
y	-6.5625	-6.8750	-6.8750	-6.8750	-6.8750	-6.5625	-6.5625	-6.2500	-6.2500	-5.9375	-5.3125	-5.0000
z	3.1250	3.4375	3.7500	4.0625	4.3750	4.6875	5.0000	5.0000	5.6250	5.9375	5.9375	6.2500

Table 1: The (x,y,z) position (rows) of the Kraken at the different times (columns)

the total time has been spent, afterwards is goes back down. After staying stationary in the y direction for a while it decreases. The z value keeps rising, this means that the Kraken is coming closer and closer to the surface.

These values are good to know exactly so that hard checks can be done on the x and y positions, but it is better to visualize the full path in a plot:

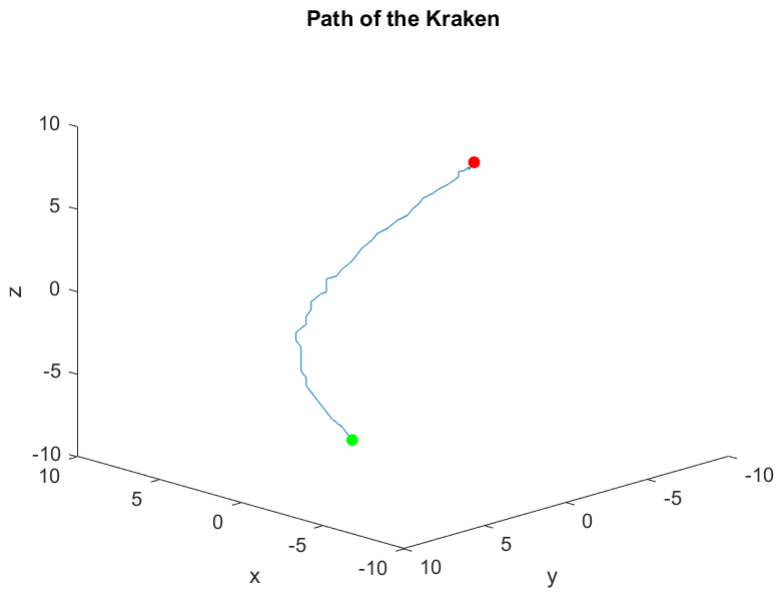

Figure 1: A three dimensional plot of the location of the Kraken over time. It starts at the green dot and goes up to the red dot.

Here the Kraken starts at the green dot and ends up at the red dot after an hour. This is a better visualization of the results, but not perfect. Since we are not as interested in dept of the Kraken as we are of the position on the hockey field, we can choose to only plot the path of the Kraken relative to it's x and y position. This then becomes:

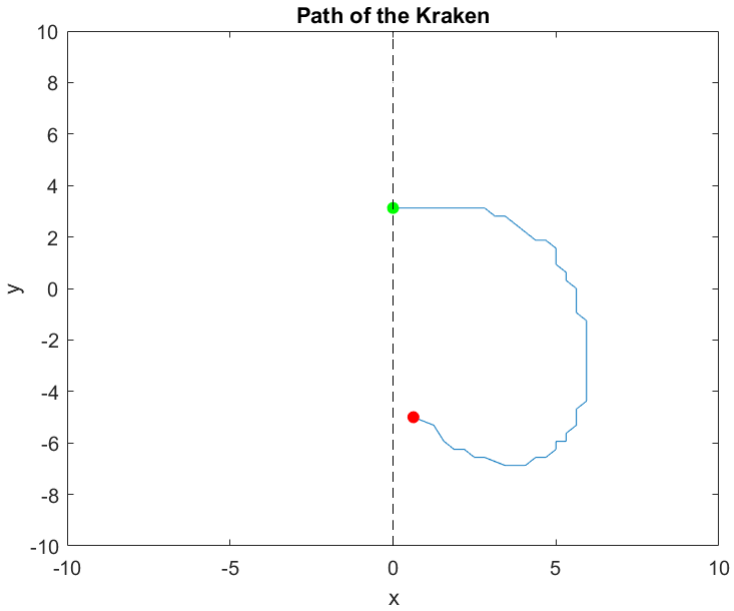

Figure 2: A two dimensional plot of the x and y location of the Kraken over time. It starts at the green dot and goes up to the red dot. The dashed line represents the middle line of the field.

Once again, the green en red dot represent the beginning and end of the path of the Kraken over time. Here a dashed line is also added to represent where the enemy's field begins. This plot perfectly represents where the hard checks must be done in order to destroy the enemy team.

5 Conclusion

Through data gotten from radar detection we had an idea of where the Kraken was located, but there was a lot of noise to be dealt with. Using transformations, namely the Fourier transformations, in order to be able to access the frequency domain of a data set, we were able to use techniques, such as spectral averaging and spectral filter, to dampen the white noise in

the data set. After this was done, the frequencies that we actually wanted to receive were found and converted back to the spatial domain. Using all of this, we were able to find the location at the Kraken at the given timestamp such that we could hard deck the enemies.

The methods used are good and accomplish what we want it to do. These methods, however, do not work for everything. We mainly relied on two characteristics of the data: there was white noise and we were trying to find a solitary wave. The averaging out of the data only gave us the important frequencies since the noise was white. This would not have been the case if we had periodic noise, for example. Besides that we also assumed the Kraken moved in a solitary wave which made it so that the frequencies would always stay the same. If this was not the case, the averaging out would have done the opposite of what we wanted. There are no immediate drawbacks to the other techniques, other filters might be needed if there were to be 2 Krakens (the sum of two Gaussians), but that is more of an inconvenience than a drawback. There are, of course, other methods to go to the frequency domain than the Fourier transform, but this is one of the easiest. There is, however, something important that we must learn from this exercise. Namely that the Kraken is rising to the surface and the players are in fact not safe.

Acknowledgment

I would like to acknowledge my fellow students who asked questions I asked myself while doing this project. The answers of the TAs were very useful to help me if I got stuck.