Spacial Signal Analysis Using the Fourier Transform

Rachel Carroll

01/24/2020

Abstract

This paper will follow the heroic tale of one, Rachel Carroll, who bravely used her knowledge of the Fourier transform, and its inverse, plus a really strong laser, to save her dog Ralph from certain death. It will outline the mathematics and data analysis techniques used identify an object of interest that is moving in time, with highly noisy data. In this case, the object is a marble inside Ralphs intestines. In particular, it will discuss how the Fourier transform is used to de-noise data, extract the relevant spacial frequency, and ultimately locate the marble... before it's too late.

1 Introduction and Overview

Yesterday, my dog Ralph swallowed a marble. For Ralph to live, it was imperative that I find the exact coordinates of the marble. The data provided contained ultrasound measurements of his intestines at twenty points in time. Unfortunately, but not surprisingly, Ralph could not sit still during the ultrasound. So the marble moved and the data was very noisy data due to the movement of Ralph's intestinal fluids.

To address these obstacles the Fourier transform was employed to plot the ultrasound data in its frequency domain, average the 20 measurements to de-noise the data, and then capture the spacial frequency of the marble. After the frequency was acquired, a Gaussian filter was used to filter the data at the identified frequency. The data was then transformed back to the spacial domain so the marble's location could be identified at different points in time. Finally, the location of the marble at the twentieth measurement was extracted. From there, all we needed to do was point an intense sound wave at the exact right location at the time of the twentieth measurement and, voila, the marble was crushed up to oblivion and Ralph is back to normal.

2 Theoretical Background

The Fourier Transform
The key to this analysis is using the Fourier transform. The foundation of the Fourier transform that any function can be represented in terms of sines and cosines. It also leverages the fact that sin and cos are orthogonal in function space. Because of these properties, the Fourier transform can be used to define a new coordinate system of frequencies. This allows us to translate between space or time domains to frequency domains. The Fourier transform and its inverse are defined as follows:

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

Where values of k are wave numbers and values of x are from the original coordinate system which could be from a spacial or time domain.

As shown above, the Fourier transform assumes an integral over the entire line from $-\infty$ to ∞. However in this analysis we will be working in a finite computational domain $x \in [-15, 15]$. The Fourier transform calculations will be done using the MATLAB function fft(), which discretizes the domain at a set number of Fourier nodes. In this analysis we will use 64 nodes. While any number of nodes can be chosen, it is best to chose 2^n points to optimizes the function by lowering floating point operation counts in the algorithm. Another important thing to note is that the fft function assumes a periodic domain of 2π. The analysis takes this into account when creating the frequency domains as discussed in the following section.

Noise in Data
A frequent issue in any signal processing analysis is the prevalence of noise in the data. When noise is large enough, which is often it, it can completely hide the signal of interest. One method to address this issue is to average many measurements. This works well because noice will eventually average to zero while the signal remains consistent.

In space this will work only if the object is stationary for each of the measurements. In this analysis the object (the marble) was moving. However, because the spacial frequency doesn't change, we used the averaging technique in the frequency domain after applying the Fourier transform.

3 Algorithm Implementation and Development

The following steps were performed in MATLAB to locate the marbles trajectory

1. **Preparing Frequency and Spacial Domains**
   The first step was to prepare the frequency and spacial domains. The spacial domain was on an interval from -15 to 15 with 64 Fourier nodes. Thus, a 3 dimensional coordinate system was created when the x, y, and z coordinates ranged from -15 to 15 in 64 steps. Since the Fourier transform is on a periodic space, the first and last points are the same. So, the first element of this domain was eliminated when building the spacial grid. The frequency grid has the domain from -2π to 2π in 64 steps, as required for the Fourier transform.

2. **Identify the Spacial Frequency of the Marble**
   To de noise the data, the Fourier transform was applied to the spacial data to so that the 20 measurements could be represented in the frequency domain. In the frequency domain, the 20 measurements were averaged. Since noise averages to zero (or very low frequencies), the averaged data's peak frequency is the frequency of the marble. It was important to average the data in the frequency domain since the marble was moving in space and time. Had the data been averaged in the spacial domain, it is likely that the marbles frequency would also average to zero, making if still indistinguishable from the noise. However, even though the marble is physically moving in space, its spacial frequency remains the same. Once the spacial frequency was identified, the location of that frequency was extracted. The following outlined the steps taken.

   (a) Reshape data into 3 dimensional volume

   (b) Take Fourier trasform

   (c) Average all 20 measurements

   (d) Find the max value of the averaged data

   (e) Find the location of the max value

   (f) Put this location in terms of the frequency coordinate system defined in part 1.

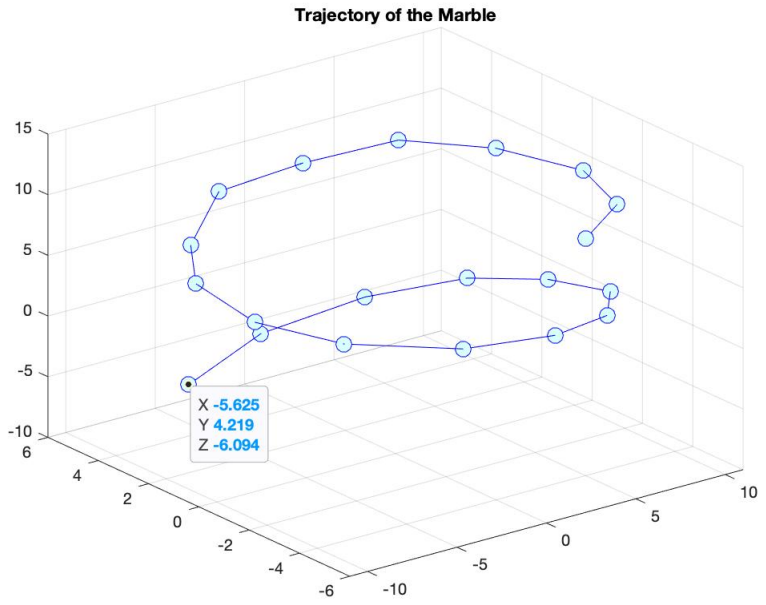
3. **Filter the data**
   Once the location of the highest frequency was identified, a gaussian filter was created and applied to the transformed (frequency) data at the location of marbled frequency. This brought other frequencies to zero.

4. **Transform back into Spacial Domain**
   Once the data is filtered, the inverse transform is used to bring the results back to the spacial domain the the physical location of the marble can be identified for each of the 20 measurements in time. Once the data is inverse transformed to the spacial domain, the location of the max value is the location of the marble.

5. **Plot Trajectory**
   For each of the 20 measurements, the x,y, and z coordinates of the marble were stored a plotted to show the trajectory fo the marble over time.

4 Computational Results

The figure below displays the trajectory of the marble and shows its final location of $\sim (-5.625, 4.219, -6.094)$.Therefore the intense acoustic wave should be focused at this location at the time of the 20th measurement.



5 Summary and Conclusions

This project demonstrates the usefulness using a the Fourier transform to interpret data with so much noise that the signal of interest is hidden. The key take away is that in some domains the data is cannot be interpreted will reliability. However, by transforming to another coordinate systems and taking advantage of certain propertied of that coordinate system, we can suddenly analyze the data well and identify the underlying signal with mush higher confidence.

6 Appendix A MATLAB functions used and brief implementation explanation

The following functions were important in this anaysis:

- fftn() - computes the fourier transform for an n dimensional array of data
- ifftn() - computed in the inverst fourier transform for an n dimensional array of data
- fftshift() - shifts the domain to its original order because, to increase computational efficiency, the fftn() function shifts the domain so that $[-L, 0] \to [0, L]$ and $[0, L] \to [-L, 0]$.
- isoplot - plots volume data as a surface at a given isovalue. This was used to plot the ultrasound data. After denoising, the plot showed the location of the highest frequency. It was also used to plot the location of the marble atfer inverse transforming the data
- plot3 - Creates 3 dimensional plots. This was used to plot the trajectory of the marble
- ind2sub - Finds location of a given value in an array. This was used to find the locaiton of the maximum frequency in the transformed data and the location of the marble after inverse transforming the filteres data back into the spacial domain. This function indexes differently than meshgrid so the indecies found had to be applied to the meshgrid dataset instead of used directly.
- meshgrid - Creates a grid from given axis vectors. This was used to create the spacial and frequency coordinate systems.

7 Appendix B MATLAB codes

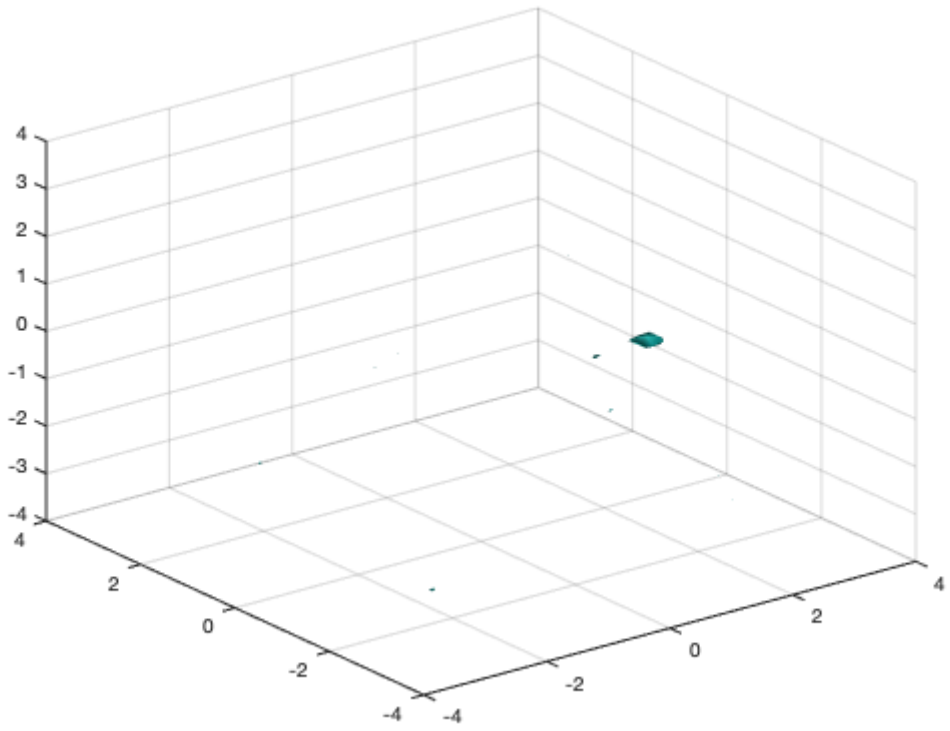
Table of Contents

Load Data and Set Up Domains

```
clear all; close all; clc;
load Testdata
L=15; % 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); % frequency
 components of FFT
[X,Y,Z]=meshgrid(x,y,z);
[Kx,Ky,Kz]=meshgrid(ks,ks,ks);
```

Average Data to Reduce Noise

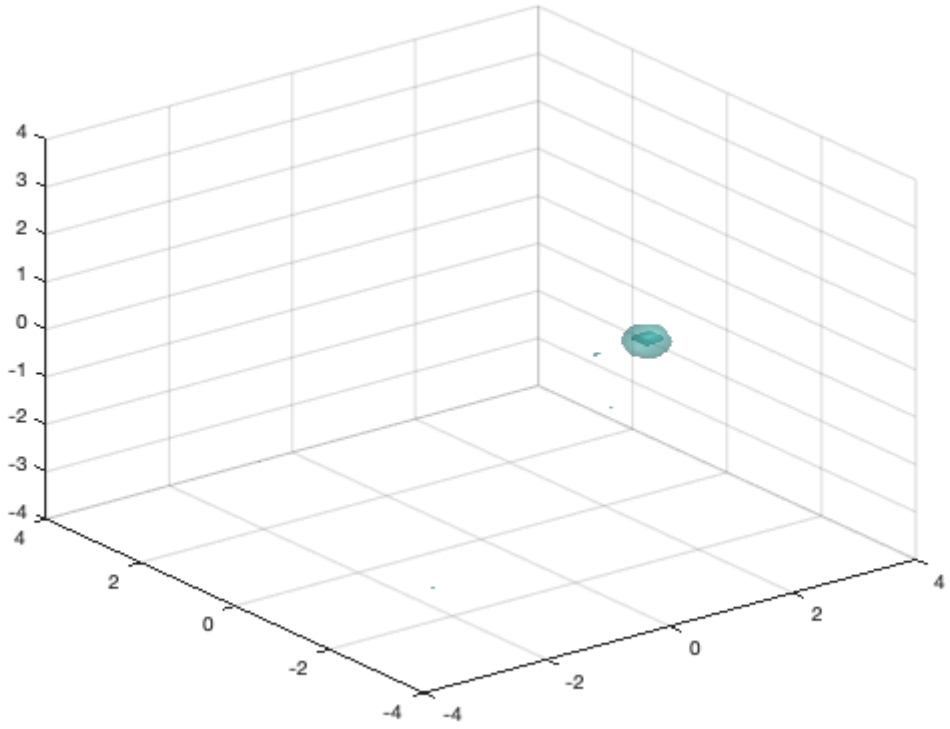
```
%Averages
Utave=zeros(n,n,n);
for j=1:20 %loop through the 20 measurments
    Un(:,:,:)=reshape(Undata(j,:),n,n,n); %put into 64x64x64 format
    Utn = fftn(Un,[n,n,n]);%apply fourier transform to bring to
 frequency domain
    Utave=Utave+Utn; %add each measurment
end
Utave=fftshift(Utave)/20; %shift transformed data and take average

%Plot Average Transform
close all
figure(1)
isosurface(Kx,Ky,Kz,abs(Utave)/max(abs(Utave(:))),.7)
axis([-4 4 -4 4 -4 4]), grid on, drawnow
```



Filter the Data

From the plot the max frequency should be around (2,-1.0)

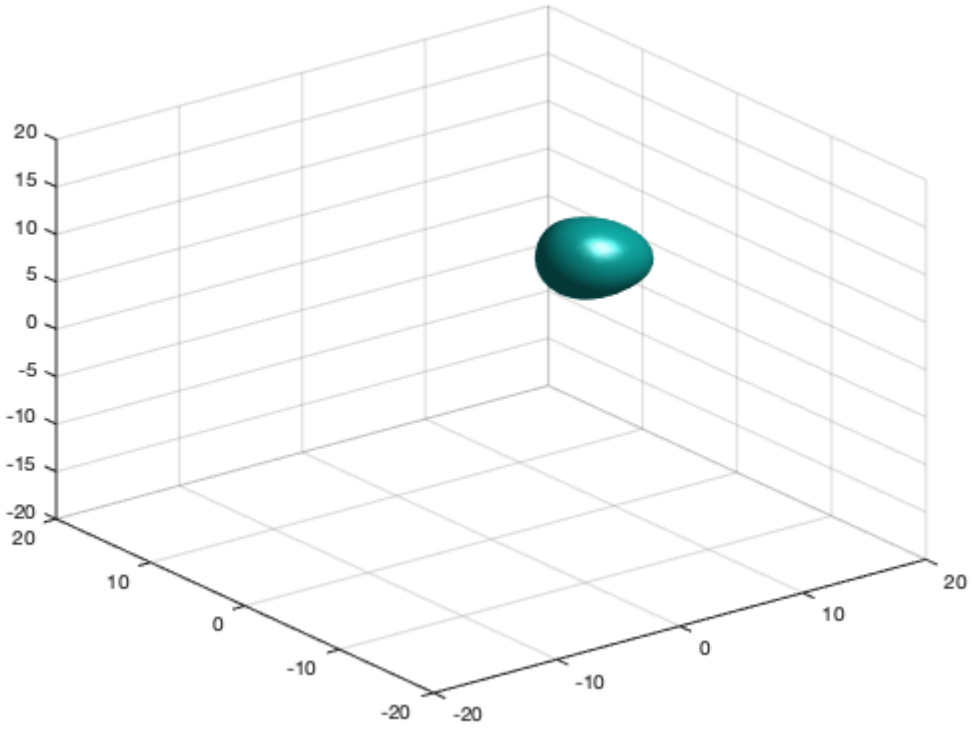
```
%find loacation of max freq
[max_freq,arg_freq]=max(abs(Utave(:)));
[i1,i2,i3]=ind2sub(size(Utave),arg_freq);
x_filt=Kx(i1,i2,i3);
y_filt=Ky(i1,i2,i3);
z_filt=Kz(i1,i2,i3);

filt = exp(-3*((Kx-x_filt).^2+(Ky-y_filt).^2+(Kz-z_filt).^2));

%%Add Filter to Plot
isosurface(Kx,Ky,Kz,filt,.7)
alpha(.3)
```



Find Location of Marble

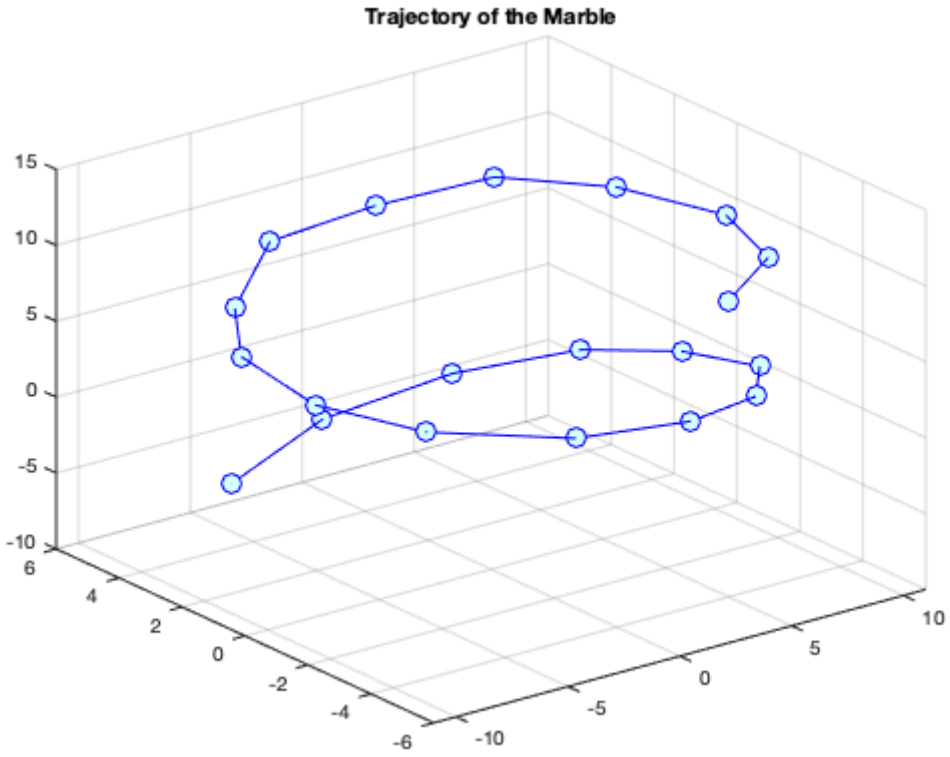
Sanity check to make sure there is a clear location of the marble at each measurment. Just using first measurment (j=1)

```
j=1;
Un(:,:,:)=reshape(Undata(j,:),n,n,n);
Utn = fftn(Un,[n,n,n]);
Utn_shift = fftshift(Utn);

unft=filt.*Utn_shift;
unf=ifftn(unft);

figure(2)
%close all
isosurface(X,Y,Z,abs(unf)/max(abs(unf(:))),.4)
axis([-20 20 -20 20 -20 20])
grid on, drawnow
```



Plot Marble Trajectory

```
x=[]; y=[]; z=[];
for j=1:20
    Un(:,:,:)=reshape(Undata(j,:),n,n,n);
    Utn = fftn(Un,[n,n,n]);
    Utn_shift = fftshift(Utn);
    unft=filt.*Utn_shift;
    unf=ifftn(unft);
    [val,location]=max(abs(unf(:)));
    [x_1,y_1,z_1]=ind2sub(size(unf),location);
    x=[x,X(x_1,y_1,z_1)];
    y=[y,Y(x_1,y_1,z_1)];
    z=[z,Z(x_1,y_1,z_1)];
end


plot3(x,y,z,'-
o','Color','b','MarkerSize',10,'MarkerFaceColor','#D9FFFF')
grid on
axis([-11 11 -6 6 -10 15])
title("Trajectory of the Marble")
```



Published with MATLAB® R2019b