An Ultrasound Problem

W. Hoyt Thomas

1/24/2020

I. Abstract

Fluffy the dog has swallowed a marble, armed with 20 ultrasounds, it must be located to break it up. Fourier transforms were used to determine the frequency of the marble which then would provide spatial coordinates of the marble. However, the ultrasounds came with significant noise, to de-noise them spectral averaging and filtering were used to determine the exact location of the marble within fluffy.

II. Introduction

My dog, fluffy, swallowed a marble and was promptly brought to a veterinarian for treatment. However, the veterinarian believed the marble had moved into the canine's intestinal tract where it would be much harder to remove. 20 ultrasounds were performed to obtain data concerning the spatial variations of a small section of the intestines where the marble was believed to be. Unfortunately, fluffy's movements caused internal fluid movement through the intestines which resulted in an extremely noisy signal.

The objective of this algorithm was to de-noise the data to the point where the marble could be located for each of the 20 ultrasounds. If the trajectory and location of the marble could be determined, a high-intensity wave would be focused on that location to break up the marble and (hopefully!) save fluffy.

III. Theoretical Background

The Discrete Fourier Transformation provided the theoretical basis of the analysis performed in the following code. Given a series of values (or vectors) from a function, the DFT will use sines and cosines at different frequencies to represent the function provided. By employing Fourier analysis, it is possible to determine the frequencies that are most responsible for constructing the function, the frequency most responsible is considered the 'central' frequency. The DFT is commonly simplified using Euler's identity, $\exp(\pm ix) = \cos(x) \pm i \sin(x)$, to compress the function and to capture complex results. The DFT is defined as:

$$\widehat{x_k} = \sum_{n=0}^{N-1} x_n \, e^{\frac{-2\pi i k n}{N}}$$

where N = total number of points provided. The inverse DFT will generate real points in space given Fourier-transformed function. This can be useful to generate a function of real points once the central frequency has been determined. The inverse DFT is defined as:

$$x_n = \frac{1}{N}\sum_{n=0}^{N-1} \widehat{x_k} \, e^{\frac{2\pi i k n}{N}}$$

once again N = total number of points. An important note regarding Fourier Transforms, because sines and cosines are used to construct the function, periodic functions work best. Practically the DFT, Order(N^2), is replaced by the Fast Fourier Transform, Order(NlogN). The FFT breaks N into factors of two to shorten the order, besides this fact they are identical operations.

Most functions are accompanied by highly variable amounts of 'noise' which make finding central frequencies difficult. Two methods, that are employed in this analysis, exist for removing this noise: filtering and averaging. Both methods are applied to the function after Fourier transform (i.e. in frequency space). Averaging is a technique that takes multiple transformed signals and averages them over the number of signals, the goal is to discover trends and possibly a central frequency. Filtering is technique that takes a transformed signal and multiplies it by a 'filter' function, the filter will usually be centered on a specific frequency to amplify it and remove other frequencies. The most common filter function is a gaussian (normal) filter:

$$f(k) = e^{-\tau(k-k_0)^2}$$

where k_0 is the central frequency and tau is the width of the filter.

IV. Algorithm Development

The first step was to create arrays of spatial [X,Y,Z] and frequencies [Kx, Ky, Kz] to be able plot the data provided. Frequencies must be re-scaled by (2*pi/L) since FFT assumes 2*pi periodic signals. With the spatial array, visualization of the data was possible, since the data came as a complex [20x262144] it had to be reshaped into a [64x64x64] array at each time step for it to be plotted by isosurface. Using isosurface to view each of the 20 ultrasounds showed extremely noisy data with no discernible pattern.

To de-noise the data, an [64x64x64] was created and filled with all zeros. An FFT was performed at each of the 20 intervals and added into this array. Note the array was not appended, at each timestep the FFT output was added to the previous one. This array ('Untave') was divided by the number of iterations, in this case, 20. Now this array contained the average transform of all twenty ultrasounds, and a simple max() command was used to find the maximum value at the central frequency. This is not the central frequency but the transform's value at the central frequency. To determine the actual central frequency, simply plot 'Untave' using isosurface() but use the maximum from the previous step as the isovalue. This will ensure only points with that value (the maximum) are plotted, the resulting plot showed a small object and the vertices of this object were averaged to determine the Kx, Ky, Kz coordinates where the max occurs. These Kx, Ky, Kz values are the central frequency.

Once the central frequency was determined, it was possible to filter the signal around that frequency. A gaussian filter was defined and applied to transformed data in frequency space to remove frequencies anything that was not the marble. An inverse FFT was then applied to get spatial data of the marble and normalized. The spatial data was passed onto isosurface to plot the marble position at each of the twenty times. Through trial and error, multiple taus were tested

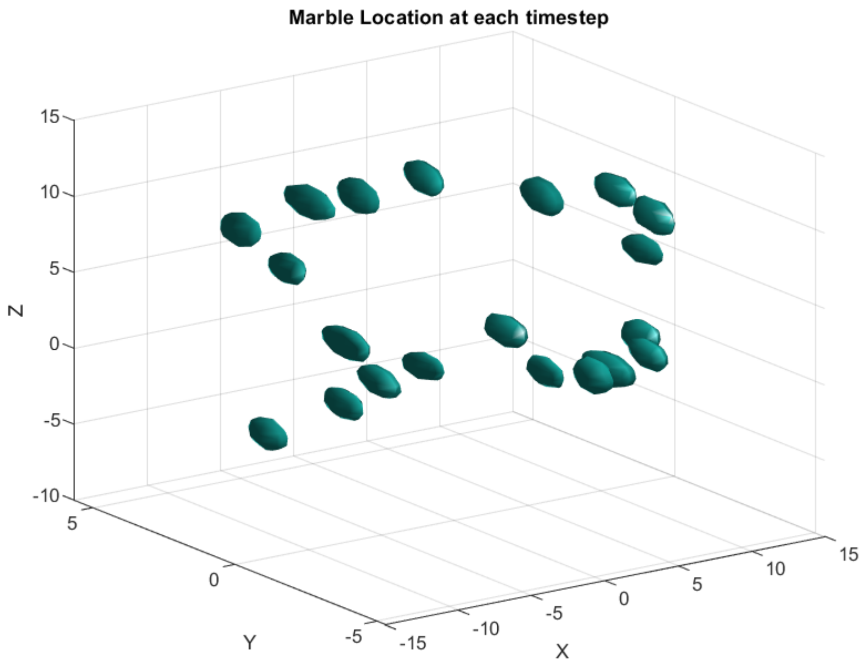

Figure 1: Marble locations at each of the timesteps plotted in time

until it produced a marble-looking shape (Fig. 1). Since isosurface can pass on the coordinates of the vertices to a matrix, one was set equal to isosurface. The vertices were summed, then averaged by the number of vertices to condense each marble into a single point. This matrix was then passed onto plot3 to plot the path of the marble (Fig. 2).

V. Computational Results

Averaging determined the central frequency to be [Kx, Ky, Kz] =

[-4.1871, 5.655, -6.7021].

Figure 1 shows the location of the marble at each of the 20 ultrasounds. Figure 2 shows the path the marble has taken to reach the current position at the twentieth ultrasound.

VI. Conclusion

By employing the use of the FFT and spectral averaging and filtering, the marble swallowed by fluffy the dog has been located. To break up the marble, an intense acoustic wave should be focused at [X,Y,Z] = [-5.6246, 4.2188, -6.0937].

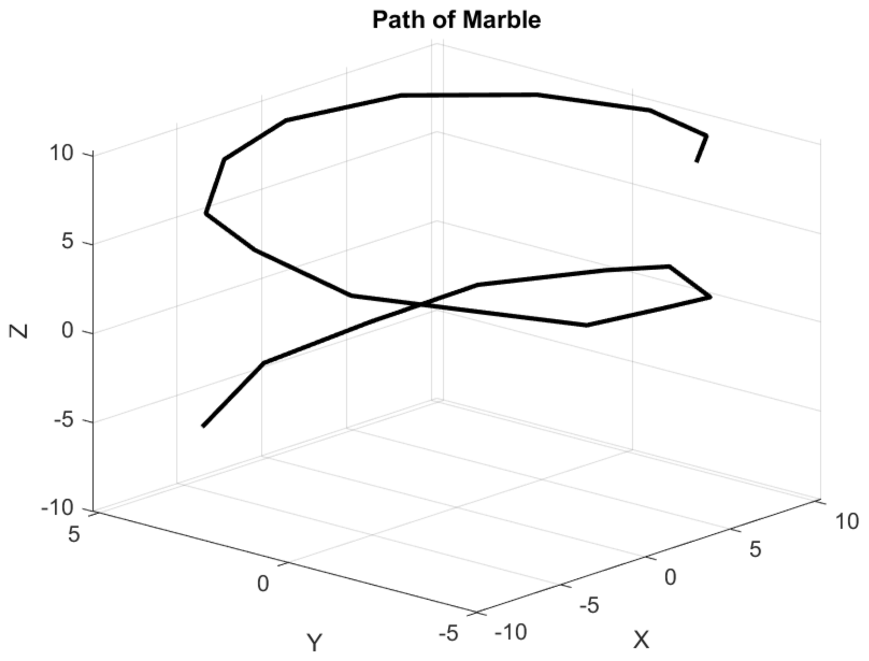

Figure 2: The path of marble in space

Appendix A – Matlab Functions

isosurface(X,Y,Z,V,isovalue)

Inputs: X,Y,Z – cartesian grids ; V – A 3D array with corresponding values to the grid points (must be same size as X,Y,Z); isovalue – sets the values of V that will be plotted

fftn(x) – Multidimensional FFT, equivalent to computing a FFT along each dimension of X

Inputs: X – an array

ifftn(x) – Inverse multidimensional FFT

Inputs: X – an array

meshgrid(x,y,z) – returns the 3D grid coordinates defined by the vectors x, y, z

Inputs: x,y,z – vectors

plot3(X,Y,Z) – plots coordinates in 3D space

Inputs: x,y,z – scalars, vectors, or matrices

Appendix B – Matlab code

```
clear; 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);
[X,Y,Z]=meshgrid(x,y,z);
[Kx,Ky,Kz]=meshgrid(ks,ks,ks);

%Visualize Signal
for j=1:20
    Un(:,:,:)=reshape(Undata(j,:),n,n,n);
    close all, isosurface(X,Y,Z,abs(Un),0.4)
    axis([-20 20 -20 20 -20 20]), grid on, drawnow
    pause(5)
end


%% Finding Central Frequency via Spectrum Averaging

Untave=zeros(n,n,n);
for j =1:20
    Untave = Untave + fftn(reshape(Undata(j,:),n,n,n));
end

Untave=Untave/20;

m = max(abs(Untave),[],'all');

figure(1);
isosurface(Kx,Ky,Kz,abs(Untave),271);
grid on;

Kx_0 = -4.1871;
Ky_0 = 5.655;
Kz_0 = -6.7021;


%% Filter Signal Around Central Frequency
tau = 1;
filter = exp(-tau*((Kx-Kx_0).^2 + (Ky-Ky_0).^2 + (Kz-Kz_0).^2));

centers = zeros(20,3);
for j=1:20
    Unt(:,:,:)= fftn(reshape(Undata(j,:),n,n,n));
    Unft = filter .* Unt;
    Unfti = abs(ifftn(Unft));
    Unfti = Unfti/max(Unfti,[],'all');
    figure(1);
    isosurface(X,Y,Z,Unfti,0.99); %plot marbles in space
    set(gca,'Fontsize',8);
    title('Marble Location at each timestep');
    xlabel('X');
    ylabel('Y');
    zlabel('Z');
    [f,v] = isosurface(X,Y,Z,Unfti,0.99999);
    centers(j,:) = sum(v)/length(v);
    grid on;
end

%plot path of marble
figure(2);
plot3(centers(:,1), centers(:,2),
centers(:,3),'k','Linewidth',2);
title('Path of Marble')
xlabel('X');
ylabel('Y');
zlabel('Z');
grid on,

target_of_acoustic_wave = centers(20,:);
```