Jeremy Aguirre

AMATH 482

January 24, 2020

Assignment #1

Abstract

In this assignment, data is provided containing 3-dimensional coordinate information collected across twenty distinct time points by an ultrasound imaging device. This data hypothetically represents the position of a marble traveling through the intestines of a dog, though it is too noisy to be useful. The data is processed (denoised and filtered) to produce a clean signal in order to determine the trajectory and last known position of the marble.

I. Introduction and Overview

The hypothetical scenario which has been proposed is as follows. I have taken my dog Fluffy to his veterinarian after I found he swallowed a marble. From what the vet can tell, the marble has passed into his intestines and will soon kill him. Ultrasound data was collected which detected spatial variations due to the marble, however Fluffy was moving during the ultrasound collection which led to a high level of noise within the data. The vet enlisted me to manipulate the ultrasound data in order to surmise the trajectory and current location of the marble. With this information an intense acoustic wave could be focused on the marble to break it up.

II. Theoretical Background

Fourier Transform and Inverse Fourier Transform

The Fourier Transform is an integral transform defined over $x \in [-\infty, \infty]$, however the computational domain is finite over $x \in (-L, L)$. With this in mind, and the Fourier Transform's ability to characterize oscillatory behaviors with e^{-ikx}, it can be thought of as a discrete sum of eigenfunctions and corresponding eigenvalues represented as wavenumbers. It is expressed in numerous different forms. One such form is written as:

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

where k is the wavenumber. The Inverse Fourier Transform also has many different forms and can be written as:

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

Both are commonly used in time-frequency analysis.

Denoising by Averaging

The general concept is that white noise can be removed from a signal by averaging the signal. This idea is based on the concept that white noise can be accurately modeled by adding a normally distributed set of random variables with zero mean and unit variance to a signal.

Therefore, over longer signals white noise averages to zero and can be removed through averaging.

3D Gaussian Filter

A 3D Gaussian Filter is represented by the following equation:

$$F(k_x, k_y, k_z) = \exp\left(-\sigma_x(k_x - a)^2 - \sigma_y(k_y - b)^2 - \sigma_z(k_z - c)^2\right) \tag{3}$$

where σ_x, σ_y, and σ_z represent the filter's bandwidths in each respective direction (x, y, z); a, b, and c represent the center-frequency values for their respective directions; and k_x, k_y, and k_z represent the wavenumbers for their respective directions.

III. Algorithm Implementation and Development

After loading the data collected from the ultrasound, a few parameters were set. The spatial domain was set $L = 15$ and there were $n = 64$ Fourier modes which were the discrete values used to set the computational domain for the Fourier transform. The wavenumber k range was set from 0 to $n/_2 - 1$ and $-n/_2$ to -1 to avoid aliasing and rescaled by $2\pi/_{2L}$ so that the signal had a period of 2π while the domain was $x \in (-L, L)$. k was adjusted using fftshift to make the center frequency zero. Then meshgrid was used to set a $64x64x64$ coordinate system for K_x, K_y, K_z and X, Y, Z (frequency domain and time domain) to match with the data provided. The data provided was reshaped row by row using a for loop to put it in a form that matched the 3D coordinate system that was established, and the Fourier transform was applied to each row (fftn). At this point the signal was denoised by averaging in order to better determine the maximum intensity frequency using the max function. The index of the maximum intensity frequency was converted using ind2sub into matrices corresponding to its subscript values. These yielded the center frequencies needed to construct the 3D Gaussian filter in the form of equation (3) with a bandwidth of 0.2 chosen to resemble the size of the marble. fftshift was used on the filter to counteract the shift that was applied earlier to k. The filter was then applied to (multiplied with) the Fourier transform of the reshaped data provided. The inverse Fourier transform was applied (ifftn) and the same process applied after denoising involving the max and ind2sub functions was utilized. The resulting 'center locations' (corresponding to the previous center frequencies) were stored in the 20x3 matrix named marble to be plotted using plot3. The plot (Figure 1) was produced showing the trajectory of the marble

IV. Computational Results

Frequency Signature (Center Frequency)

The center frequencies are represented in Appendix B by variables which match those found in equation (3). It was found through averaging of the spectrum that $a = 1.8850$, $b = -1.0472$, and $c = 0$.

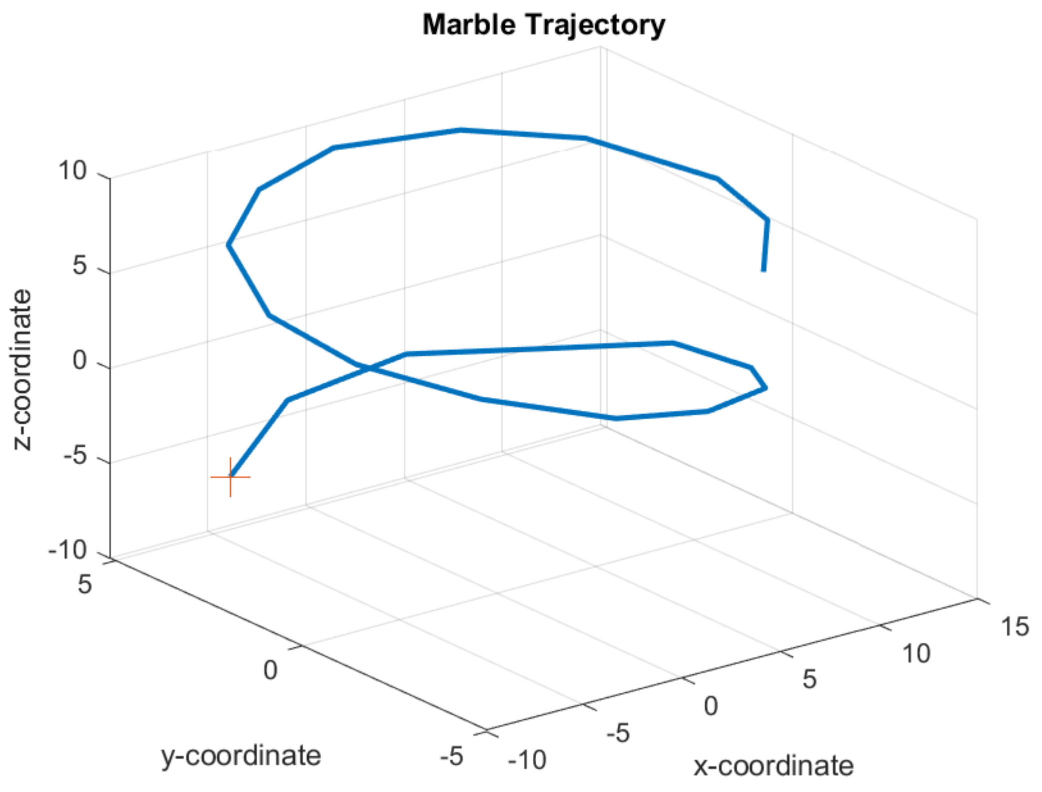

Figure 1. The trajectory of the marble traveling through the dog Fluffy's intestines and its final position marked with a red cross.

The denoised data revealed the path that the marble took through the dog's intestines and is represented by Figure 1.

Marble Location at 20th data measurement

An intense acoustic wave should be focused at the coordinates $(-5.6250, 4.2188, -6.0938)$ denoted by the red cross in Figure 1.

V. Summary and Conclusions

Data was provided containing the 3-dimensional position of a marble passing through the intestines of a dog collected from an ultrasound imaging device across twenty distinct time points. The signal contained in the data was too noisy to be interpreted so it was denoised through averaging and filtered using a 3D Gaussian filter around the center frequency, with a bandwidth roughly the size of the marble. Post-filtering the signal could be interpreted to reveal the travel trajectory of the marble and its position to focus an intense acoustic wave. The methods of signal processing used in this exercise could have many other potential applications such as radar detection, interpretation of other medical imaging systems such as MRI, speech or image processing in telecommunications, etc.

Appendix A

fftn – used to apply the Fourier transform

fftshift – used to shift the center frequency to zero and back again as needed for easier analysis

ifftn – used to apply the inverse Fourier transform

reshape – used to change the dimensions of a matrix while maintaining the same number of elements

plot3 – used to plot in 3 dimensions

ind2sub – converts index values into an array of corresponding subscript values

Appendix B

```
%% AMATH 482: Homework #1
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);

%% Denoising Signal by Averaging
Utave = zeros(n,n,n);
for i=1:20
    Un(:,:,:) = reshape(Undata(i,:),n,n,n);
    Utave = fftn(Un) + Utave;
end
Utave = abs(fftshift(Utave)) / max(abs(Utave),[],'all');  %
Denoised Signal

%% 3D Gaussian Filter Construction
[M,I] = max(Utave,[],'all','linear');
[Ix,Iy,Iz] = ind2sub([n n n],I);
a = Kx(Ix,Iy,Iz);  % Center Frequency for x
b = Ky(Ix,Iy,Iz);  % Center Frequency for Y
c = Kz(Ix,Iy,Iz);  % Center Frequency for Z
sigmax = 0.2;  % X-direction Bandwidth
sigmay = 0.2;  % Y-direction Bandwidth
sigmaz = 0.2;  % Z-direction Bandwidth
```

```
filter = exp(-sigmax*(Kx-a).^2 - sigmay*(Ky-b).^2 - sigmaz*(Kz-
c).^2);
filter = fftshift(filter);  % 3D Gaussian Filter

%% Application of 3D Gaussian Filter to find Marble Locations
marble = zeros(20,3);
for j=1:20
    Un(:,:,:) = reshape(Undata(j,:),n,n,n);
    Unt = fftn(Un);
    Unft = filter .* Unt;
    Unf = ifftn(Unft);

    [M,J] = max(Unf,[],'all','linear');
    [Jx,Jy,Jz] = ind2sub([n n n],J);
    marblex = X(Jx,Jy,Jz);  % Marble X locations
    marbley = Y(Jx,Jy,Jz);  % Marble Y locations
    marblez = Z(Jx,Jy,Jz);  % Marble Z locations

    marble(j,1) = marblex;
    marble(j,2) = marbley;
    marble(j,3) = marblez;
end

%% Visualization of Marble Trajectory
plot3(marble(:,1),marble(:,2),marble(:,3),'linewidth',2), hold
on
plot3(marble(20,1),marble(20,2),marble(20,3),'+','MarkerSize',15
)
grid on
title('Marble Trajectory')
xlabel('x-coordinate')
ylabel('y-coordinate')
zlabel('z-coordinate')
print(gcf, '-dpng', 'marble_trajectory.png')
```