A Fluffy Signal Problem

Casper Hsiao

2020-01-20

Abstract: The objective of this problem is to locate the trajectory of a marble inside Fluffy's intestines. The vet obtained the ultrasound data over a small area of Fluffy's intestines where the marble was suspected to be. To clean up the noise, the signals were averaged in the spectral domain to determine the center frequency of the marble. Furthermore, Gaussian filter was applied to the signals in the spectral domain around the center frequency to filter out the noise. As a result, the processed signals provided a clean trajectory of the marble inside Fluffy's intestines. Fortunately, the marble was located and Fluffy was saved from the deadly marble.

I. Introduction and Overview

Fluffy, my dog, accidentally swallowed a huge marble and was suffering from it. I brought Fluffy to the vet as soon as I realized something was wrong with her. The vet obtained a set of ultrasound data regarding the spatial domain of a small area of Fluffy's intestines where the marble was suspected to be. However, the signals obtained is highly noisy due to Fluffy's movement and the motion of internal fluid in the intestines. Therefore, the vet has no idea about the location of the marble with the noisy signals. As a result, the vet was unable to locate the marble inside Fluffy's intestines. Fortunately, I was able to apply my knowledge of signal processing that I learnt in AMATH 482 to filter out the noise in the signal and locate the trajectory of the marble inside Fluffy's intestines. First of all, the set of ultrasound data was transformed to the spectral domain with the fast Fourier transform. Furthermore, the signals in the spectral domain was averaged to denoise the signals, therefore, the center frequency of the marble was identified. With the center frequency determined, the signals were filtered with the Gaussian filter in the spectral domain where the center frequency is. Lastly, the processed signals were transformed back to the spatial domain with the inverse fast Fourier transform to determine the trajectory of the marble. As a result, an intense acoustic wave was focused on the location of the marble to break it.

II. Theoretical Background

The Fourier transform was implemented intensively throughout the course of the ultrasound data processing. The Fourier transform is a generalization of the complex Fourier series of a non-periodic function. The Fourier series is an expansion of a periodic function in terms of a sum of sines and cosines of different frequencies. In the case of this problem, the Fourier transform was applied to transform the ultrasound data from the spatial domain into the frequency domain for filtering. The fast Fourier transform (FFT) is the specific Fourier transform algorithm implemented for the purpose of computational performance. Furthermore, averaging and filter were implemented in the frequency domain of the data to attenuate the noises.

II.A. Fourier Series

The Fourier series expands a given function into a trigonometric series of sines and cosines of different frequencies as shown in Eq. 1. The series produce a 2π-periodic function due to the expansion of sines and cosines on the interval $x \in (-\pi, \pi]$. The Fourier coefficients a_n and b_n are solved as shown in Eq. 2 and Eq. 3 respectively.

$$f(x) = \frac{a_o}{2} + \sum_{n=1}^{\infty}(a_n \cos nx + b_n \sin nx) \qquad x \in (-\pi, \pi] \tag{1}$$

$$a_n = \frac{1}{\pi}\int_{-\pi}^{\pi} f(x)\cos nx dx \qquad n \geq 0 \tag{2}$$

$$b_n = \frac{1}{\pi}\int_{-\pi}^{\pi} f(x)\sin nx dx \qquad n > 0 \tag{3}$$

II.B. Fourier Transform

As shown in Eq. 4 and Eq. 5, the Fourier transform and its inverse is an integral transformation on the entire interval $x \in [-\infty, \infty]$. The Fourier transform decomposes the original function $f(x)$ into to the frequency domain in terms of the wavenumbers k. The absolute value of the transformation determines the amplitude of that frequency in the original domain. Furthermore, the kernel of the transformation e^{-ikx} represents the oscillatory behavior of the transformation.

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

$$f(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} e^{ikx} F(k) dk \tag{5}$$

II.C. Fast Fourier Transform

The fast Fourier transformation (FFT) is a highly efficient algorithm developed to perform the forward and backward Fourier transformation. The operation count of FTT dropped to $O(NlogN)$. This was a huge improvement compared to the other algorithms at the same period, which were Gaussian elimination and LU decomposition ($O(N^3)$). The FFT transformation is performed on a finite interval $x \in [-L, L]$, and the solutions have periodic boundary conditions. Furthermore, the FFT discretizes the inverval $x \in [-L, L]$ into 2^n points, which lowers the operation count to $O(NlogN)$.

II.D. Noise Attenuation

The averaging of signals in the frequency domain requires two conditions to produce useful signal information: the noise is white noise and collect multiple signal data over a time period. Under these conditions, the averaged signal "zero outs" the noises, which produces a clean, localized signature in the frequency domain. However, the averaged signal losses the time domain dynamics as a consequence.

Once the center frequency is found from spectral domain averaging, spectral filters can be applied on the signals in the frequency domain to attenuate the undesired frequencies. In this case, the spectral filter applied was the Gaussian filter as shown in Eq. 6. k_o is the desired center frequency that the filter isolates. τ is the bandwidth of filter.

$$F(k) = e^{-\tau(k-k_o)^2} \tag{6}$$

III. Algorithm Implementation and Development

III.A. Create the mesh-grid for the spatial and frequency domains

The ultrasound data was loaded into MATLAB, where the 20 rows of the data corresponds to the 20 different ultrasound signals that were taken in time. The Fourier nodes were defined to sample n points over the spatial domain. A spatial domain array was created through *linspace* with $n + 1$ points evenly spaced over the spatial domain $[-L, L]$. Furthermore, the last entry of the array was excluded, due to periodic boundary conditions, to create a n sized array. A frequency domain array was created by scaling the period from $2L$ to 2π.

Furthermore, the array was shifted through $fftshift$, where $x \in [0, L] \rightarrow x \in [-L, 0]$ and $x \in [-L, 0] \rightarrow x \in [0, L]$, due to the FFT default. The three dimensional frequency and spatial domain mesh-grid was created through $meshgrid$ with the arrays.

III.B. Identify the center frequency of the marble through spectral domain averaging

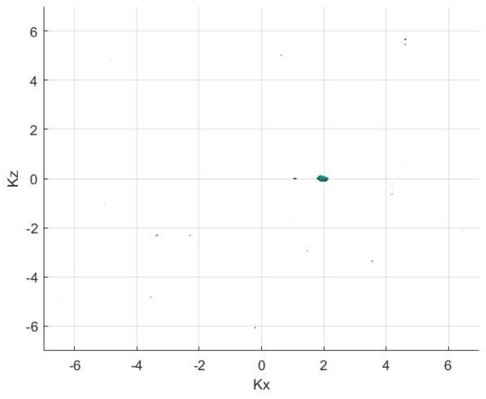
A for loop was created to iterate each row of the ultrasound data, where the rows were rearranged into three dimensional matrices through $reshape$, and stored in a cell array. Furthermore, all the matrices inside the cell array were transformed through $fftn$, and averaged to create a single matrix of average frequency domain signal. The center frequency was determined through max by identifying the location of the maximum value in the averaged signal matrix.

III.C. Apply the Gaussian filter and identify the spatial trajectory of the marble.

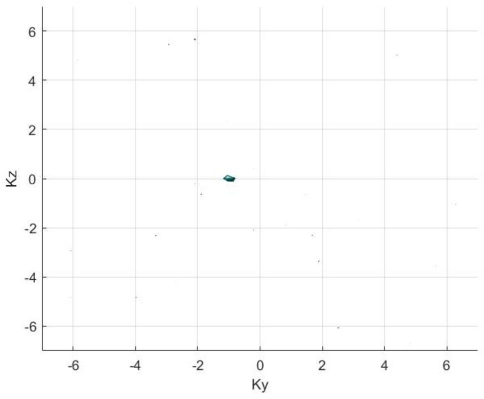
A three dimensional Gaussian filter was created with the center frequency information and an reasonable bandwidth value τ. A for loop was create to apply the filter to the 20 different frequency domain signals, and the filtered signals were transformed back to the spatial domain through $ifftn$. As a result, the trajectory of the marble was found by locating coordinates of the maximum value in each time domain signal.

IV. Computational Results

As shown in Figure 1, the isosurface plot of the spectral domain averaged signal identified the center frequency of the marble easily as the noise was attenuated. The coordinates of the green pebble sized volume displayed in Figure 1a and 1b represented the center frequency of the marble in XZ and XY frequency domain. Table 1 showed the k_ox, k_oy and k_oz wavenumbers correspond to the center frequency of the marble.



(a) The isosurface plot of the spectral domain averaged signal XZ



(b) The isosurface plot of the spectral domain averaged signal YZ

Fig. 1: The isosurface plot of the spectral domain averaged signal showing the center frequency of the marble.

$k_o x$	$k_o y$	$k_o z$
1.8850	−1.0472	0

Table 1: The wavenumber of the center frequency of the marble.

The signal was filtered through the Gaussian filter, and the trajectory of the marble in the spatial domain was plotted as shown in Figure 2. Furthermore, the initial and final location coordinates of the marble was tabulated in table 2. With the information of the final coordinate of the marble inside Fluffy's intestines, the vet focused the intense acoustic wave at that position to break the marble.

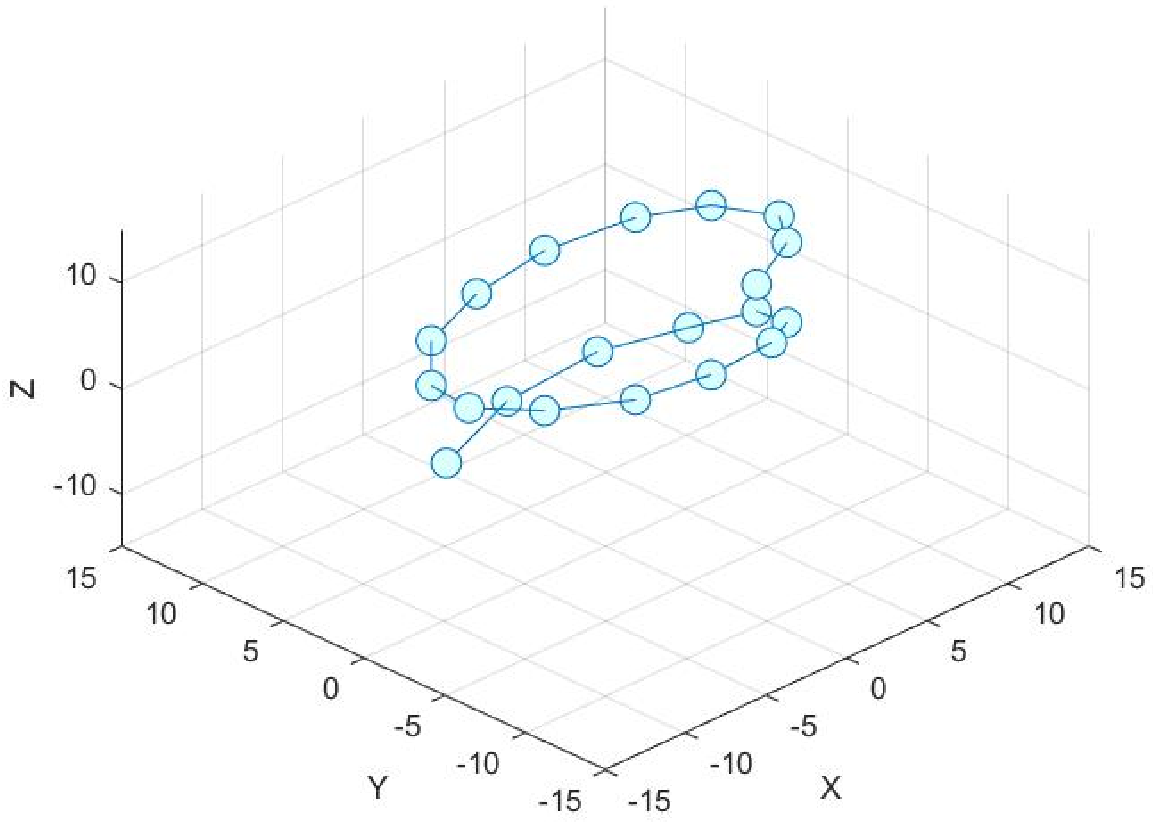


Fig. 2: The trajectory of the marble in the spatial domain.

Data	X	Y	Z
1	4.6875	-4.6875	9.8438
20	-5.6250	4.2188	-6.0938

Table 2: The initial and final location coordinates of the marble

V. Summary and Conclusions

The spectral domain averaging is a powerful tool in attenuating noises, especially the white noise. Although the time domain dynamics are loss in the process of averaging, the information extracted, center frequency, from the averaged signal is utilized in conjunction with the spectral filter to attenuate the noises in each data set that were taken in time. Furthermore, the cleaned signals with the center frequency isolated were used to track down the path of the marble in the spatial domain. As a result, the spectral domain averaging and spectral filter are two powerful tools that are applied complementary in signal processing. This reflects the key concept of time frequency analysis, which is to resolve information between time and frequency domain to trace the target.

Appendix A

$linspace(x1, x2, n)$: Creates a vector of n points evenly spaced between $x1$ and $x2$.
$fftshift(x)$: Rearranges x by shifting the zero-frequency component to the center.
$meshgrid(x, y, z)$: Creates a three dimensional grid with the coordinates of x, y and z.
$reshape(x, n, n, n)$: Rearranges x into a n-by-n-by-n matrix.
$cell(n, n)$: Creates an empty n-by-n cell array.
$zeros(n, n)$: Creates an n-by-n matrix with entries of zero.
$fftn(x)$: Computes the multidimensional fast Fourier transform of x.
$max(x)$: Finds the maximum value in x and the index of the value.
$abs(x)$: Computes the complex magnitude of each element in x.
$ind2sub(sz, I)$: Computes the subscript value that corresponds to the linear indices I for a matrix of size sz.
$isosurface(X, Y, Z, V, ISOV)$: Computes the isosurface plot with the three dimensional grid X, Y and Z from the volume data V at the isosurface value specified $ISOV$.
$ifftshift(x)$: Rearranges x by the inverse of $fftshift$.
$ifftn(x)$: Computes the inverse multidimensional fast Fourier transform of x.
$plot3(x, y, z)$: Computes plot of x, y and z coordinate in a three dimensional space.

Appendix B

Contents

```
clc; clear all; close all;

load('Testdata.mat')
```

Create the spatial and spectral domain.

```
L = 15;
N = 64;
x = linspace(-L, L, N+1);
x = x(1:N);
y = x;
z = x;
k = pi/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);
```

Transform the data into spectral domain and averaging the signals.

```
c = size(Undata, 1);
Un = cell(1, c);
Unt = cell(1, c);
Untave = zeros(N, N, N);
for j = 1 : c
    Un{j} = reshape(Undata(j, :), N, N, N);
    Unt{j} = fftn(Un{j});
    Untave = Untave + Unt{j};
end
Untave = Untave/c;
```

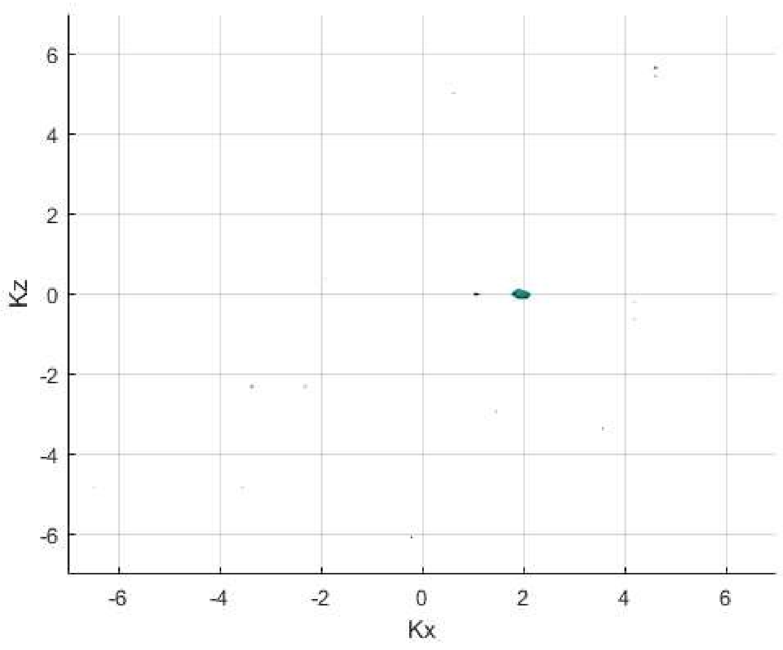
Locate the center frequency in the averaged signal.

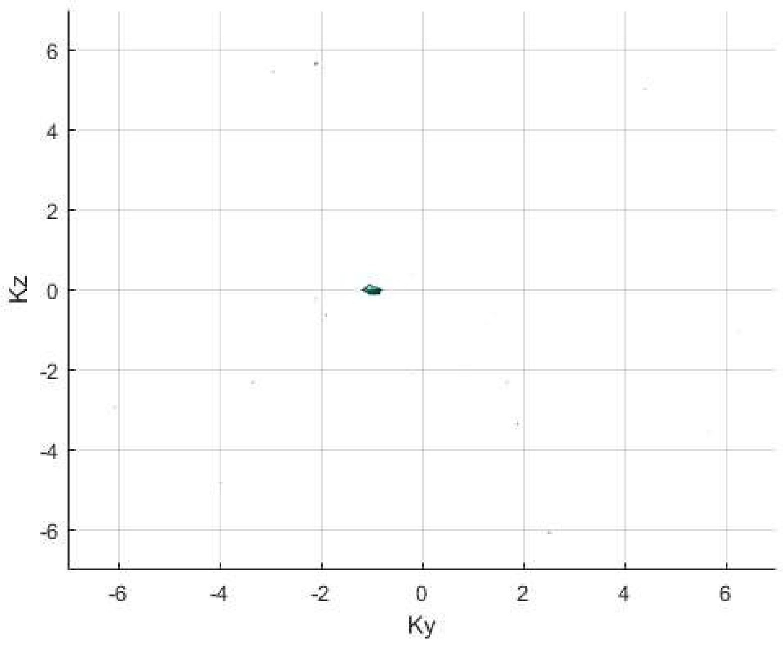
```
[M, I] = max(fftshift(abs(Untave)), [], 'all', 'linear');
[Ix, Iy, Iz] = ind2sub(size(Untave), I);
k0x = Kx(Ix, Iy, Iz);
k0y = Ky(Ix, Iy, Iz);
k0z = Kz(Ix, Iy, Iz);

figure(1)
```

```
isosurface(Kx, Ky, Kz, fftshift(abs(Untave))/M, 0.7);
xlabel('Kx');
ylabel('Ky');
zlabel('Kz');
axis([-7 7 -7 7 -7 7]);
view(0, 0);
grid on;

figure(2)
isosurface(Kx, Ky, Kz, fftshift(abs(Untave))/M, 0.7);
xlabel('Kx');
ylabel('Ky');
zlabel('Kz');
axis([-7 7 -7 7 -7 7]);
view(90, 0);
grid on;
```





Create the Gaussian filter.

```
tau = 0.2;
filter = exp(-tau*(ifftshift(Kx) - k0x).^2).*exp(-tau*(ifftshift(Ky) - ...
    k0y).^2).*exp(-tau*(ifftshift(Kz) - k0z).^2);
```

Filter the signals.

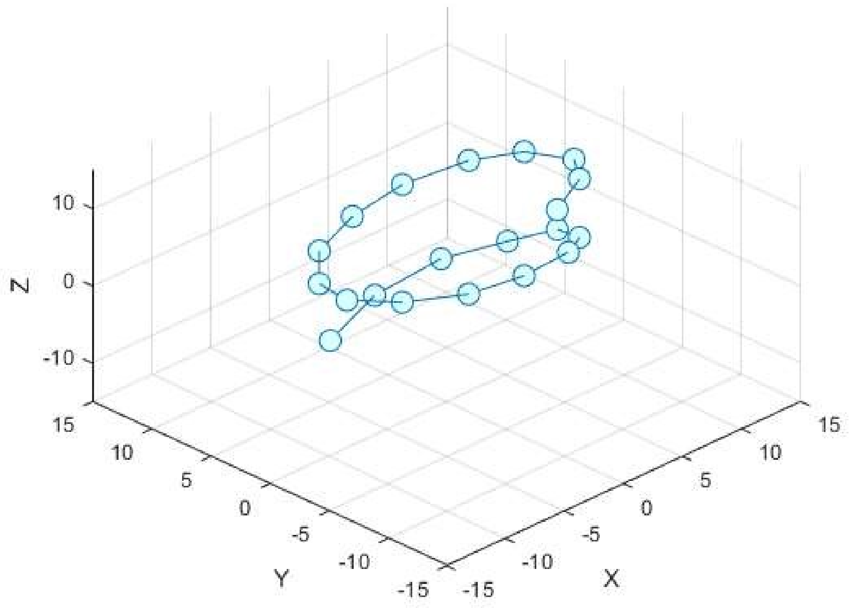
```
Untf = cell(1, c);
Unf = cell(1, c);
for j = 1 : c
    Untf{j} = filter.*Unt{j};
    Unf{j} = ifftn(Untf{j});
end
```

Locate the trajectory of the marble

```
XX = zeros(1, c);
YY = XX;
ZZ = XX;
for j = 1 : c
    [M , I] = max(abs(Unf{j}), [], 'all', 'linear');
    [Ix, Iy, Iz] = ind2sub(size(Unf{j}), I);
    XX(j) = X(Ix, Iy, Iz);
    YY(j) = Y(Ix, Iy, Iz);
    ZZ(j) = Z(Ix, Iy, Iz);
```

```
end

figure(3)
plot3(XX, YY, ZZ, '-o', 'MarkerSize', 10, 'MarkerFaceColor', '#D9FFFF');
axis([-15 15 -15 15 -15 15]);
xlabel('X');
ylabel('Y');
zlabel('Z');
view(-45, 45);
grid on;
```



Published with MATLAB® R2019b