University of Washington, Seattle

Ultrasound Signal Processing to Identify Objects with Denoising

With Application in Localizing Objects in Animal Body

Abstract:

This paper illustrates an introduction to Fourier transform and its application to engineering fields, including an application to time-frequency analysis and feature extraction. This paper demonstrates the denoising ability of Fourier transform through an application in detecting a marble's location inside a dog's intestine through ultrasound data (fake data).

Hung Vinh Ngo
AMATH482
Professor Kutz
Wednesday, January 16, 2019

Section I: Introduction and Overview

In applications that involve capturing natural properties of phenomenon and movement objects, it is usually best to decompose it into wave components due to the fact that many things in nature can be described incredibly well in frequency. For instance, the music concert that our phones capture in a video is not always just one definite soundwave, it is composed of many sounds from the surrounding, including noise from people or even the minimal sounds come from errors in the phone's hardware. Therefore, we need a way to remove the noise and just focus on the main music sound that we want the phone to capture. Physically, each sound has its corresponding pitch or wave frequency. This frequency measures the air pressure oscillation from the audio sources that get translated by our ears as sound. Therefore, if we can deconstruct the original sound data to a series of sound components existing in the original data, we can simply pick out the frequency that we want to hear. This technique of transforming a set of data from time domain to frequency domain, which is its series of constructing frequency components, is known as "Fourier transform". Fourier transform is one of the most significant mathematical concepts that are widely used today, due to the fact that it is often more intuitive to understand a phenomenon based on its components. If we know each "ingredient" that exists in our original data set, we can easily analyze, compare or even modify each ingredient to generate new data. In our previous case of sound recording, if we know the soundwave components, we can simply remove the noise frequency that we do not want and even enhance the frequencies that we want to hear such as the bass or the beat. Fourier transform is heavily used in image applications such as image filtering and image compression. For example, JPEG image format represents the original image data in frequency domain that records existing pixel patterns and by removing unimportant ones, it can greatly reduce the file size.

This change of perspective in data representations is a branch of mathematics called "transformation" and this field is widely applied in fields such as electrical, civil and computer engineering. For instance, when analyzing the seismic activity of an area for the past 10 year, we can dissect the data into its earthquake "strength". We can estimate the biggest earthquake occured in that time period so engineers can construct more rigid houses that can endure. We can also apply time-series data in radar detection, where we continuously sending electromagnetic waves outward and recording data that returns. We need to approximate if there is any data that corresponds to an object, such as an intruder, that reflects our signal or is it simply noise reflection from static objects such as trees.

Section II: Theoretical Background

Let's consider a signal filtering problem to better understand Fourier transform. There is sound recording in a time period and imagine we need to discover how many sound components it was composed of, in other words, to find the frequency of each contributing sound. Intuitively, the process of Fourier transform is a "machine" that has many "ingredient extraction filter" in it. By "pouring" the original noisy data, this machine is able to detect which ingredient exists. In our sound case, ingredient will be each component's wave frequency. [1]

[1] "An Interactive Guide To The Fourier Transform – BetterExplained." https://betterexplained.com/articles/an-interactive-guide-to-the-fourier-transform/. Accessed 22 Jan. 2019.

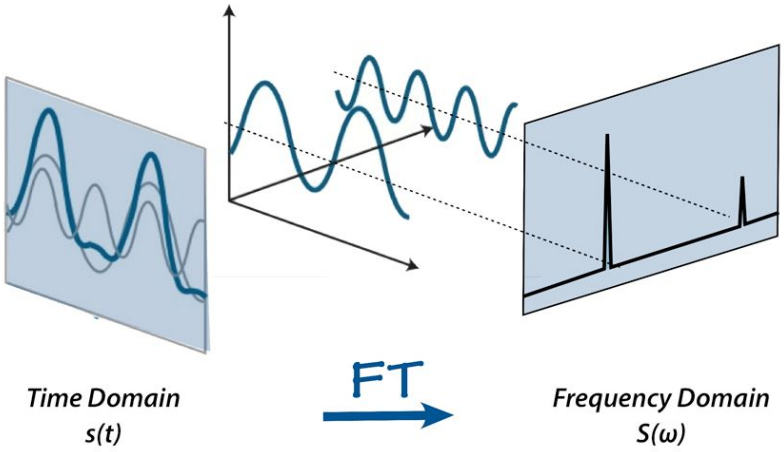

Figure 1: Using Fourier transform to switch from time domain to frequency domain[2]

After transforming our data to frequency domain, we can easily categorize the components in the original data. The spike in a frequency domain indicates that particular frequency is more likely to exist in the data.
Mathematically, to obtain the frequency domain transformation, Fourier transform applies the idea of representing any function as a series of sines and cosines functions, which correspond to the "ingredients" in the analogy.

$$f(x) = \frac{a_0}{2} + \sum_{n=1}^{\infty} (a_n cosnx + b_n sinnx) \quad ; x \in (-\pi, \pi]$$

$$a_n = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x)\, cosnx dx \,;\; b_n = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x)\, sinnx dx$$

Equation 1: Fourier transform of breaking the original function as a series of sines and cosines

By trying out theoretically all values of n, the transform is checking all possible sine and cosine with respect to the original data, in other words, it is checking its composing "ingredients".
Intuitively, the formulas to calculate the a_n and b_n coefficients will automatically determine if the original data exhibits similar pattern with the current filter that it is considering.
If the formula extended to consider on the domain over the entire line $x \in (-\infty, \infty)$ and apply Euler's identity for sines and cosines definition:

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

Equation 2: Fourier transform with imaginary number over real domain

Essentially, it is still the same concept of transforming our original data in time domain, $f(x)$, to frequency domain, $F(k)$, with k representing wavenumber (frequency) of the detection filter.

[2] "Fourier Transform (FT) - Questions and Answers in ... - MRI Questions." http://mriquestions.com/fourier-transform-ft.html. Accessed 22 Jan. 2019.

In reality, time-series data usually contain noise in it, which prevent us from detecting underlying dominant components. For example, in video recording, there can be white noise from the surrounding that "blurs" the sound that we want to measure. Therefore, the frequency domain in Fourier transform will not be significant enough since there are too many frequency components are joining at the same time. However, because we can take advantage of that white noise is random, there will be moments that this particular frequency noise appears at this time step, but does not appear in another time. Therefore, if the frequency signal is averaging overtime, theoretically, all frequency signal should be approximately close to 0. If the object that we try to detect consistently radiates the same frequency overtime, its frequency should does not change over time and as a matter of fact, it will be the most significant in the frequency domain.

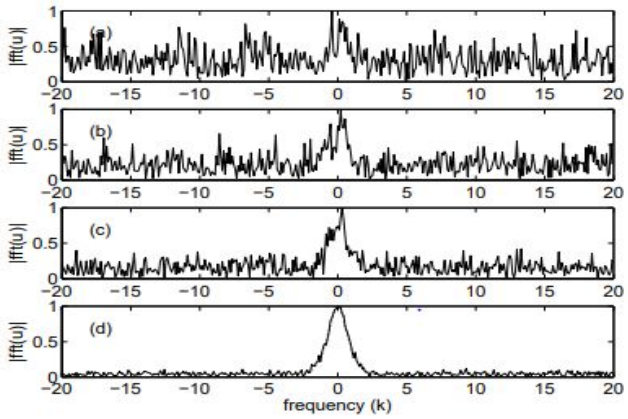

Figure 2: Average frequency for one, two, five and one hundred timesteps of data.[3]

However, one flaw of the Fourier transform is that it can detect the frequency components, but does not tell us when those components occur in time. For instance, if we are interested in seeing when one particular frequency happens in a time period, we will need to use spectral filtering. Spectral filtering is the technique of focusing on to a significant frequency in frequency domain by considering only a small neighborhood around that frequency (and zeros out others). By inverting that filtered data back to time domain with inverse Fourier transform, we have better sense of how much that particular frequency exists at each timestep.

Section III: Algorithm Implementation and Development

This paper will apply the principles of Fourier transform to analyze noisy ultrasound data and detect object in animal's body. In the data provided in the appendix B, there are the data collected from 20 different timesteps recording ultrasound signal data concerning spatial variations in small area of intestines of a dog. This dog accidentally swallowed a marble which

[3] "Basic Computation & Visualization - University of Washington." https://faculty.washington.edu/kutz/KutzBook/KutzBook.html. Accessed 22 Jan. 2019.

we need to identify its location inside the body through the data provided.

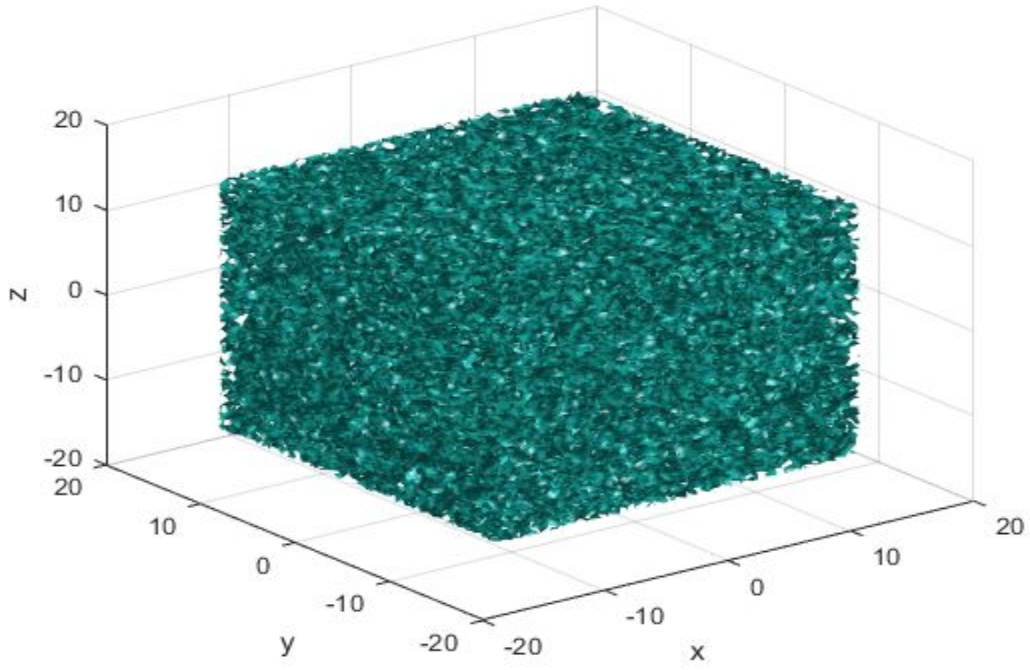

Figure 3: Spatial data of the ultrasound in 20th timestep

Because various fluid movement inside the dog's body generates highly noisy data, no significant pattern can be discovered in the data. To transform spatial data to frequency domain, we can apply Fourier transform.

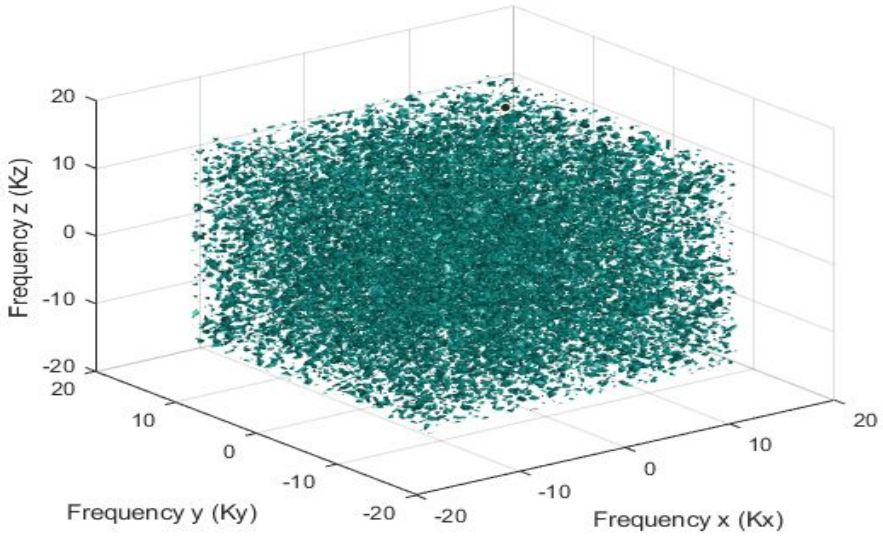

Figure 4: Frequency domain signal plot for signal at 20th timestep

The frequency seems to cluster at around the origin center frequency but it still contains noise frequency from various sources inside the body so it is hard to distinguish the marble's signal. However, since the data is collected periodically through 20 timesteps, the noise from internal fluid movement approximately will not follow any pattern so essentially the noise will be distributed similar to a normal distribution. Furthermore, the normal distribution will have mean close to 0 since fluid movement is periodical movement around an equilibrium, the average displacement should be close to 0. Thus, if the time series data is averaging across timesteps, it will theoretically cancel out all the noise and leaves with the consistent data reflecting from the marble.

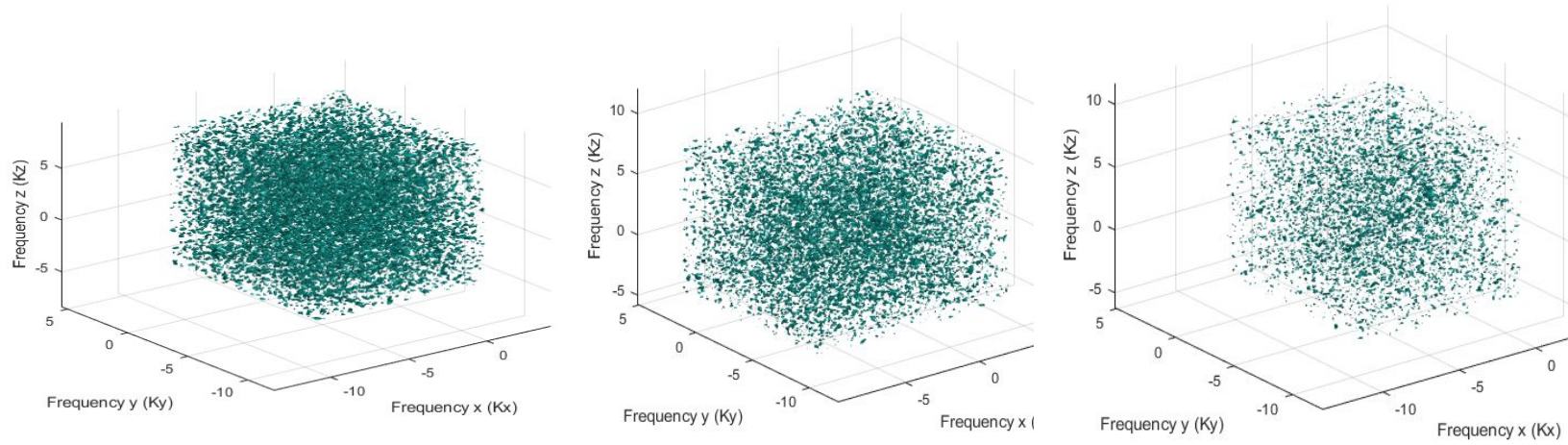

Figure 5: Average spectral contents for a) 5 b) 12 c)20 realizations of data

After averaging out the frequency data, a lot of the noise are canceled out and the data is left with "common" frequency across 20 timesteps. However, another assumption of the given data is that the marble continuously produces consistent signal data, so after averaging, the largest frequency signal left will be from the marble since it almost does not decrease through time. As a result, we can find the marble's signal as the largest value. So now the problem reduces down to matching the marble's original spatial location in the intestine with that signal. To convert back from frequency domain to our original spatial coordinate system, we can linearly transform back through inverse Fourier transform. Since we know the frequency that we want to relate the position to, we can apply the ideas of spectral filtering to restrict our window of interest and locate the source of that frequency. In this example, we will choose a Gaussian filter to clear out the other signal and only focus on the local area of our desired signal.

$$G(Kx,\ Ky,\ Kz) = e^{-\alpha(K_x - K_x^0)^2 - \alpha(K_y - K_y^0)^2 - \alpha(K_z - K_z^0)^2}$$

Equation 2: Gaussian filter for multidimensional data

where ɑ measures the bandwidth of the filter and K_x^0, K_y^0, K_z^0 is the center wavenumber that we want to focus on. After filtering with the Gaussian filter centered at the marble's frequency at each timestep, we can inverse transform that frequency window to get the corresponding spatial

window that yields the desired signal. To extract the expected location of the marble, we can simply find the location that produces the highest signal in the spatial window.

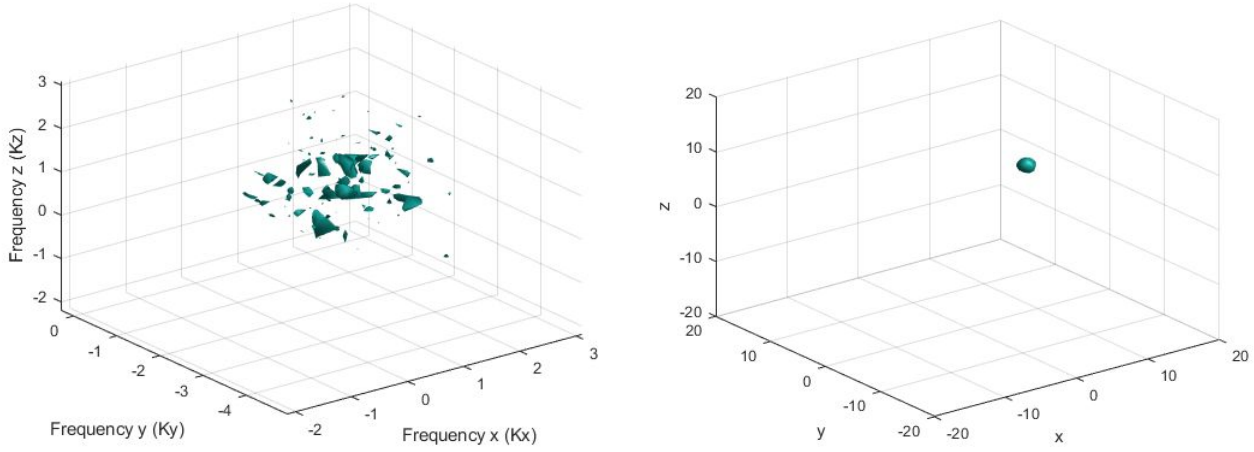

Figure 6: (left) post-filtered signal window in the frequency domain. (right) inverse transform of the signal window in the original spatial domain in the first time step (the marble's initial position)

Section IV: Computational Results

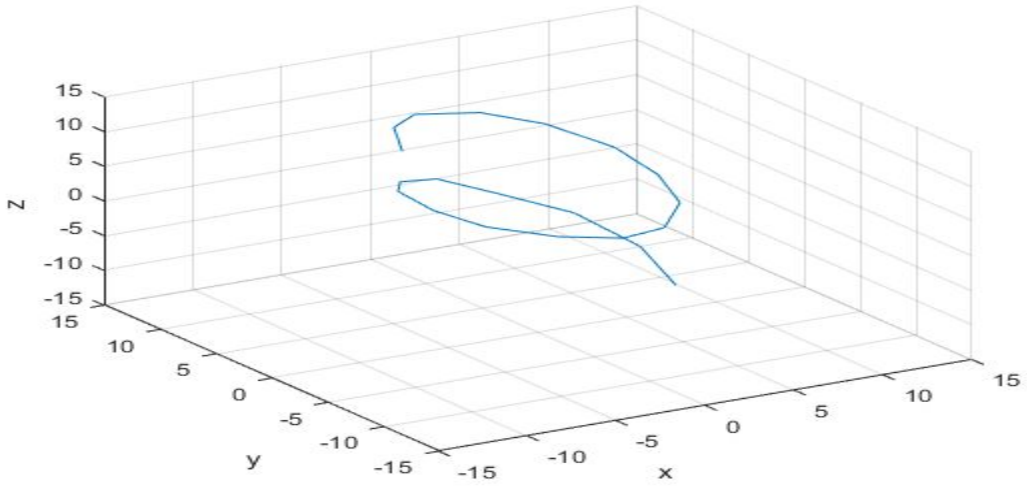

Figure 7: Trajectory location of the marble through 20 timesteps.

Frequency_x	Frequency_y	Frequency_z
28	42	33

Table 1: The frequency signature generated by the marble

Timestep	x	y	z
1	-4.6875	4.6875	9.84375
2	-2.8125	8.4375	9.84375
3	-0.46875	10.3125	9.375
4	2.34375	8.90625	9.375
5	4.21875	6.09375	8.90625
6	5.15625	1.40625	8.4375
7	4.6875	-3.28125	7.96875
8	3.28125	-7.5	7.5
9	0.9375	-9.84375	6.5625
10	-1.40625	-9.84375	6.09375
11	-3.28125	-7.03125	5.15625
12	-4.6875	-2.8125	4.21875
13	-4.6875	1.875	3.28125
14	-3.75	6.5625	2.34375
15	-1.875	9.375	0.9375
16	0.46875	9.84375	0
17	2.8125	7.96875	-1.875
18	4.6875	4.21875	-2.8125
19	5.15625	-0.9375	-4.21875
20	4.21875	-5.625	-6.09375

Table 2: Location of the marble over 20 time steps.

To destroy the marble in the 20th timestep, an intense acoustic wave should target at position (42, 24, 20).

Section V: Summary and Conclusions

This paper gives an overview and intuition of Fourier transform and its properties to be used widely in various engineering fields, including application in time-series data analysis and feature extraction. Many natural phenomenons exhibits wave properties such as sound, light or even quantum particles, therefore, Fourier transform revolutionizes the way we observe the data in a more natural way of how things work. Generally, it is often extremely useful to know what components is the data consists of so that it will be easier to understand its behaviour and properties when combined the elements together. Furthermore, understanding the features will give us a chance of creating similar datasets which is really helpful when generating training data for machine learning models or image processing techniques.

Appendix A: MATLAB functions used and brief implementation explanation

[X,Y,Z] = meshgrid(x,y,z) returns 3-D grid coordinates defined by the vectors x, y, and z. The grid represented by X, Y, and Z has size length(y)-by-length(x)-by-length(z). In the code, it is used to generate the grid for X,Y,Z spatial location and Kx,Ky, Kz as frequency domain for plotting.

Y = fftshift(X) rearranges a Fourier transform X by shifting the zero-frequency component to the center of the array. This method is used to shift the Fourier transform to its proper representation.

fv = isosurface(X,Y,Z,V,isovalue) .This function is used for plotting purposes and visualize high dimensional data.

fftn and ifftn is used for multi-dimensional Fourier transform and inverse Fourier transform.

Definitions of functions are taken from official Matlab docs[4]

Appendix B: MATLAB Codes

```
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);
[X,Y,Z]=meshgrid(x,y,z);
[Kx,Ky,Kz]=meshgrid(ks,ks,ks); %frequency
%matrix to store average frequency signal, which will contain the
%central signal since the noise get canceled out
Uave = zeros(n,n,n);

for j=1:20
   %Average out for each time step
   Un(:,:,:)=reshape(Undata(j,:),n,n,n); %this is the data we received each time
   Ut(:,:,:) = fftn(Un); %convert to frequency space
   Uave = Uave + Ut;
end
Uave = fftshift(abs(Uave)) / 20; %take average after each time step
Umax = max(max(max(abs(Uave)))); %spectral signature
close all, isosurface(Kx,Ky,Kz, Uave / Umax,0.4);

axis([-20 20 -20 20 -20 20]), grid on, drawnow
title('Fig 1: Average spectral profiles for the 20 realizations')
xlabel('Frequency x direction')
ylabel('Frequency y direction')
zlabel('Frequency z direction')
[freq_max_x, freq_max_y, freq_max_z] = ind2sub([n,n,n], find(Uave == Umax));
```

[4] "MATLAB Documentation - MathWorks." https://www.mathworks.com/help/. Accessed 22 Jan. 2019.

```
width = 0.2;
filter = exp(-width * (Kx - Kx(freq_max_x,freq_max_y,freq_max_z)).^2 - width * (Ky
-Ky(freq_max_x,freq_max_y,freq_max_z)).^2 - width * (Kz
-Kz(freq_max_x,freq_max_y,freq_max_z)).^2);
filter = fftshift(filter);
position_over_time = zeros(20,3);
%%
for k = 1:20
    Un(:,:,:)=reshape(Undata(k,:),n,n,n); %this is the data we received each time
    Ut(:,:,:) = fftn(Un);
    Utf(:,:,:) = Ut(:,:,:) .* filter;
    Utif(:,:,:) = ifftn(Utf);
    isosurface(X,Y,Z,(abs(Utif)),0.4)
    [index_x, index_y, index_z] = ind2sub([n,n,n], find(abs(Utif) == max(max(max(abs(Utif))))));
    position_x = x(index_x);
    position_y = y(index_y);
    position_z = z(index_z);
    position_over_time(k, 1) = position_x;
    position_over_time(k, 2) = position_y;
    position_over_time(k, 3) = position_z;

    title('Fig 2: Spatial position of object in animal body for the 20 realizations')
    xlabel('x ')
    ylabel('y ')
    zlabel('z')
     axis([-20 20 -20 20 -20 20]), grid on,
     drawnow
     pause(.1)
end

%%
close all, isosurface(X,Y,Z,abs(Un),0.4)
axis([-20 20 -20 20 -20 20]), grid on, drawnow
title('Figure 1: Noisy ultrasound data of small area in intestines where the marble is expected to
be at 20th realization')
xlabel('x ')
ylabel('y ')
zlabel('z')
%%
figure
plot3(position_over_time(:,1), position_over_time(:, 2), position_over_time(:,3));
axis([-15 15 -15 15 -15 15]);
grid on;
xlabel('x ')
```

```
ylabel('y ')
zlabel('z ')
```