Homework 1: an Ultrasound Problem

Karl Marrett (kdmarrett@gmail.com)

Abstract

In this report, we use methods for analyzing high noise ultrasound recordings of a marble in a dog's intestine. More specifically, we will be implementing the Fast Fourier Transform (FFT), an algorithm that uses the insights of the similarly named mathematical technique that decomposes a time signal into a linear sum of periodic functions. By transforming the ultrasound data into a frequency space and finding the target frequency of the marble, we gain a unique way to denoise the volume data. We seek to show through these techniques:

1. The center frequency reflected by the marble
2. How designing and applying a filter around this frequency can reveal the trajectory of the marble,
3. Where in space an intense acoustic wave can be focused to breakup the marble at a specific time.

1 Introduction and Overview

Ultrasound is widely used in medical as well as veterinary imaging.[1] During an ultrasound, objects need to remain still in order to get a clear image. This is why sometimes animals and children are sedated in order to take imaging data. In the ultrasound, a transducer creates and receives high frequency echoed sound waves. Changes in the reflected sounds frequency and direction can be used to reconstruct the size, location, and consistency of whatever it is recording.

In this problem ultrasound is used to image the location of a highly dense object compared to the surrounding intestine: a marble. Due to movement of the dog and the intestine, the raw volume data given is too noisy to locate the marble. Although these movements produce enough noise in the time domain to render our volume data unreadable, the frequency signature of our ultrasound on the marble is constant in all time points and independent of the marble's location. Thus, although some noise remains in the frequency domain, averaged over the twenty time points, the frequency of our marble emerges as a strong peak when we view the absolute values of the frequency spectrum. Assuming that the volume data we are interested in is coming from the major frequency of the ultrasound, we can design a filter that attenuates other frequencies. With the new frequency representation, we can

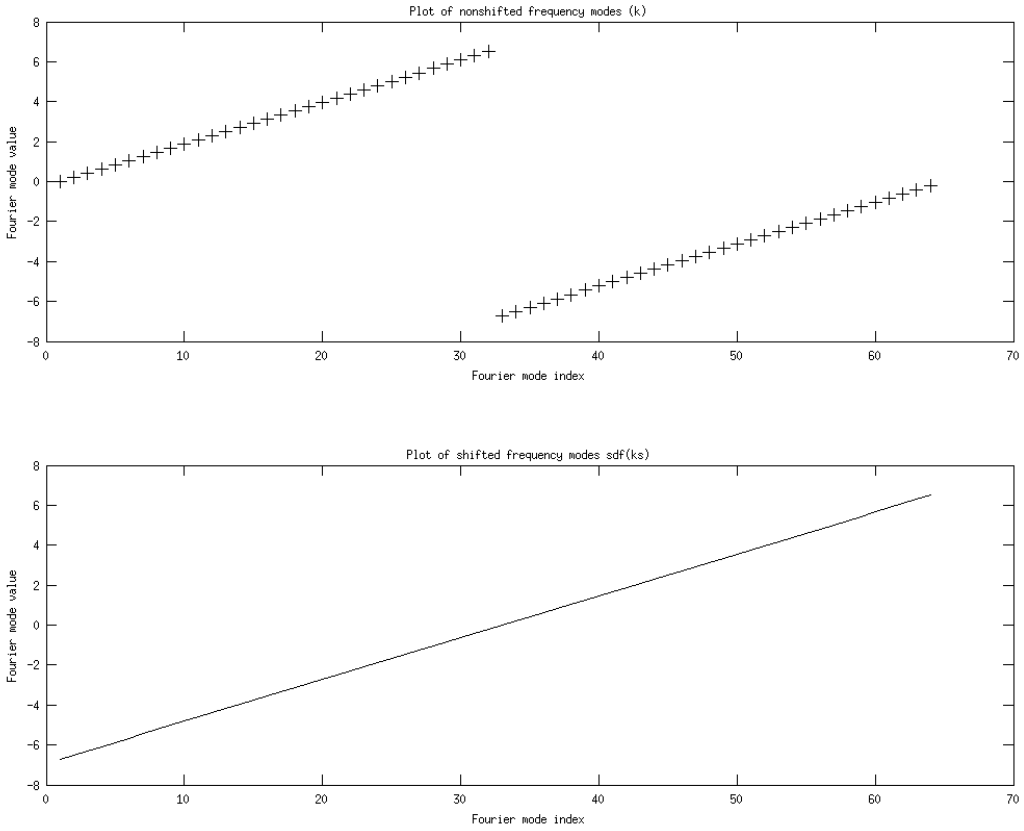

Figure 1: Shown is the frequency modes of the FFT shifted and nonshifted. Note that in the nonshifted case the interval 0 to L is switched with the interval of -L to 0. Without shifting the data returned from the FFT algorithm, the values would also be shifted in a similar way.

reconstruct our spatial information to show a cleaner view of the intestine. Once this accurate image is formed in the volume domain the trajectory and location can be deduced as shown in the Computational Results section.

2 Theoretical Background

From our understanding that an ultrasound functions by recording an echoed signal around its signature frequency, we gain the intuition that the relevant part of our image can be represented with this frequency. In order to take advantage of this fact we need to transform our data into a representation in a set of periodic functions. The Fourier transform does this by transforming a vector from, for example, the space domain, into a sum of sines and cosines as shown in equation 1.[2]

$$f(x) = \frac{a_0}{2} + \sum_{n=1}^{\infty}(a_n \cos nx + b_n \sin nx) \qquad x \in (-\pi, \pi] \tag{1}$$

Similarly, the Fourier transform kernel can also be in the imaginary domain in continuous space as shown in equation 2. This is closer to the implementation that MATLAB uses.

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

Since this analysis is done with a computer, we will be implementing the fast Fourier transform (FFT), a discrete form of Fourier analysis. The FFT algorithm has a broad set of applications and implications but, for sake of brevity, we will only consider those that are directly relevant to the analysis of this report.

1. It has low runtime ($O(NlogN)$)
2. It runs on a finite interval $x \in$ some interval $[-L, L]$
3. The range $[-L, L]$ is discretized into 2^n points
4. It is distinctly accurate beyond most other discretization schemes

While implementing the FFT we must also be aware that it also:

- Switches the intervals and multiplies every other mode by -1
- Assumes the function is periodic on an interval of 2π.

3 Algorithm Implementation and Development

Throughout our discussion of implementation we will intermix a high level discussion of how we manipulated the data with their specific implementation with functions of MATLAB, a programming language. For further information, please refer to Appendix A on all function names italicized, otherwise for specific clarification on the script follow along in Appendix B.

We start with the assumption that the volume data occurs on a spatial domain of [-15,15] and with the assumption that the ultrasound has the spectral resolution of 64 Fourier modes (2^8). This means that we can construct our xyz coordinate system for the volume data by linearly spacing $n+1$ across our interval using *linspace*. Since the assumption of FFT is that it is periodic on the given interval, we know that all points on the boundaries (point 1 and $n+1$) are the same. Therefore we can discard the last data point so that we are operating on a space of n points. This creates a basis for plotting our data in three dimensions but we still need to create a similar basis in the Fourier domain, denoted by the variable k. To do so, we need to take into account that the FFT implemented shifts the data and assumes that the interval is 2π as discussed in the previous section. Adapting to these constraints,

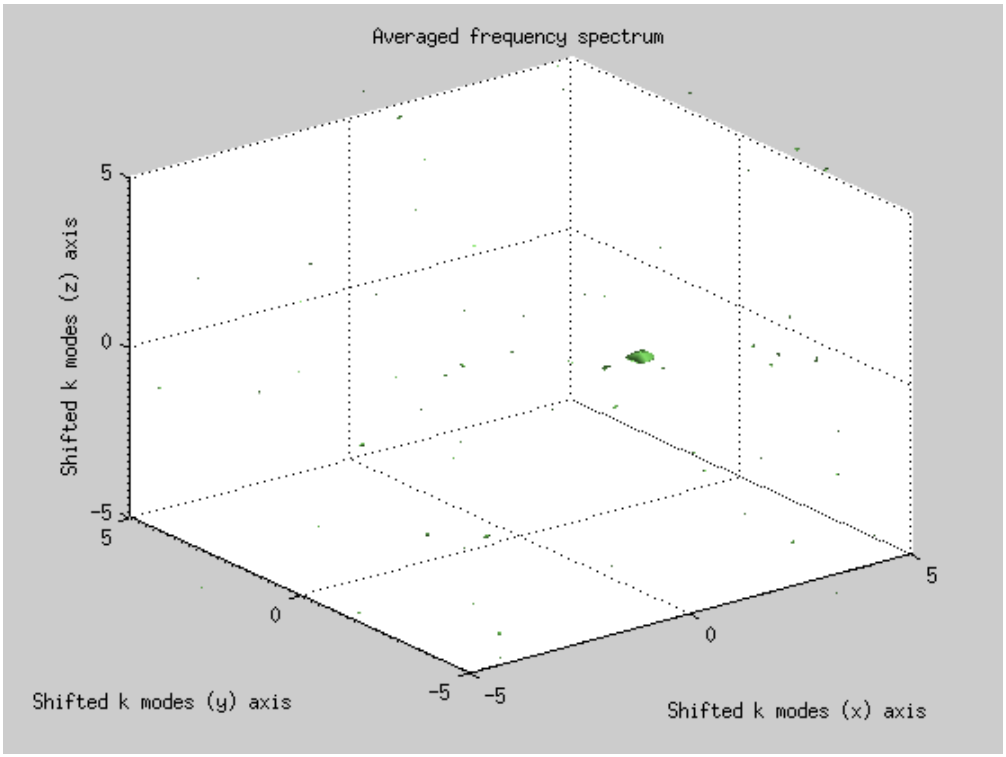

Figure 2: Shown is the *isosurface* plot of value .62 of the normalized average frequency from all the time slices. The cluster towards the center represents the peak of high values representing the frequency of the marble refracting the ultrasound frequency.

we build k, our Fourier modes scaled from now to $2L$ and concatenate two vectors to create our shifted k Fourier modes of length decided by our n Fourier modes as shown in 1. Finally we use *meshgrid* to create a three dimensional space to plot our volume and spectral data.

This problem begins from a 20 row matrix where each row represents one time slice. The volume data is saved to a vector so in order to analyze it in a 3 dimensional matrix we have to *reshape* it (see Appendix A). Given volume data in this matrix we take the fast Fourier transform along each dimension using *fftn* which returns a 3-dimensional Fourier transform for each time slice. Because ultrasound relies on a signature frequency, the volume data will be marked with high power (estimated by abs value) in some frequency modes as well as other random noise in other modes. The frequency modes of the ultrasound will remain regardless of the specific time points so we can sum all of the spectral contents at each slice. Since we are looking for some approximation of power we can take the absolute value of the frequency components, since, due to the implementation of FFT algorithms at this point, it has imaginary components.

The point of summing the spectral components is to have some averaging in order to find an accurate peak. Once we have denoised in this way finding the maximum across all three dimensions is a simple way to approximate our ultrasound frequency. To confirm that the maximum taken was not noise, we can normalize all our data between 0 and 1 then

increment an *isosurface* for higher values. We normalize as a means of averaging and to make plotting with *isosurface* easier.

Note that thus far all the operations on our data have been in a nonshifted space. So, in order to view a coherent image we must use the *meshgrid* Kx, Ky, Kz of our shifted ks and we must *fftnshift* our normalized data. Indeed, as we iterate over different values in *isosurface* a cluster forms around the maximum value that we recorded for the signature frequency as shown in figure 2, indicating that this is the actual ultrasound frequency.

Once we have an idea where the relevant information is in the frequency space we can take out all other frequencies by assuming they are noise. This process is formally known as band pass filtering, in other words, taking a window around some frequencies. A common formula for windowing is the Gaussian, which we implement in three dimensions around the peak values for each dimensions. The exact formula for the Gaussian is shown in the MATLAB code in Appendix B.

Having saved the original FFT data from the original summing step, for each time slice we do matrix multiplication of the our spectral data with our filter, then we take the inverse FFT *ifftn* and the absolute value since it is volume data to recover the filtered data for each point. Now that the volume data is filtered, the maximum values correspond to the position of the marble in each slice. Taking these maximum coordinates and projecting them onto our *meshgrid* vectors reveals the path of the marble as shown in 3. Although the goal was to find the location of the marble, the rescaling and averaging of the data was done in separate parts of the script to analyze the data. In other words, efforts were made to keep the original integrity of the data into the final filtered step.

4 Computational Results

By averaging the spectral components of the volume data we get a clear dominant frequency as shown in 2. Using the frequency modes of this peak to guide design of our Gaussian filter (shown in Table 2, and finding the max of each time slice for the marble location we can see the marbles trajectory 3. The marble's location at the twentieth data measurement can be approximated by the maximum of the volume data at that slice given in 1. This table gives the location that an intense acoustic wave must be focused in order to breakup the marble.

5 Summary and Conclusions

Analyzing data in a periodic space proves to be a powerful way to use key information about the nature of the recording to denoise data. With our approach, we were able to take

x	y	z
-5.6	4.2	-6.1

Table 1: Marble location coordinates at the twentieth time point on the interval [-15,15].

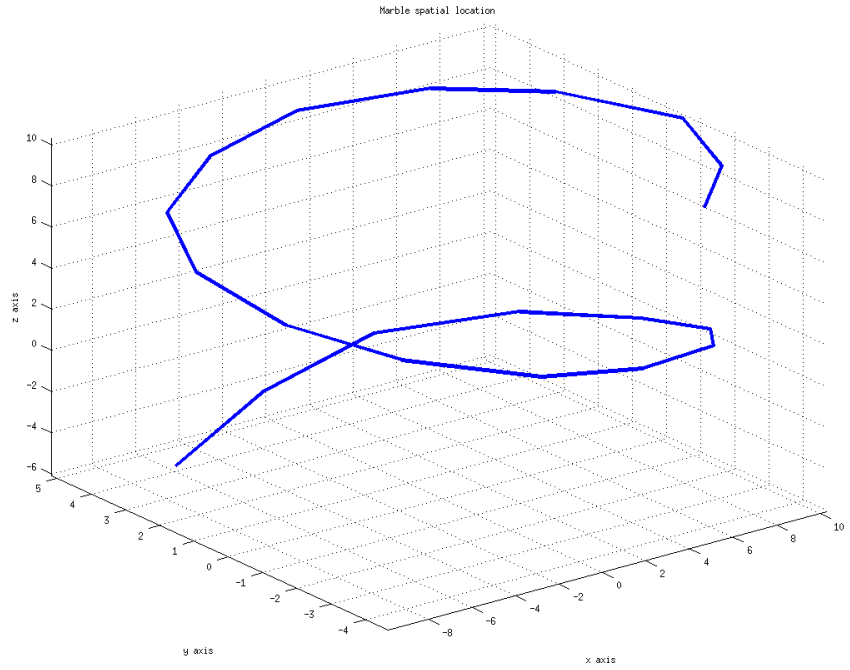

Figure 3: This tracks the trajectory of the marble through the volume space of the intestine. The marble's location is computed by taking the maximum of the volume data of each filtered time point.

otherwise unusably noisy volume data and represent it in the frequency domain. With the knowledge that our recording equipment, the ultrasound, constructs data from a specific frequency, we were able to average over all of our time points to recover information about the signature mode. Since the frequency information of the ultrasound was independent of location and constant throughout the measurements, by looking at measurements proportional to the power in the frequency domain, we were able to pick out the precise frequency of the ultrasound. Using only the frequency information of the data around the ultrasound's native frequency led to a clear picture of its location. This analysis sheds light on the power of choosing the right basis for representing data based upon prior knowledge of the problem.

K mode (x)	**K mode** (y)	**K mode** (z)
-1.04	1.88	0

Table 2: Shown are x, y, and z maximum frequency modes when averaging across the *fftn* of each time slice. This maximum frequency corresponds to the signature frequency of the ultrasound recording device. By designing a three dimensional band pass filter around these coordinates we significantly denoised our data.

A

meshgrid(A,B,C) takes the vectors A,B,C given and returns a 3 dimensional grid of each vector making it useful for plotting with isosurface.

linspace(A, B, C) creates C evenly spaced points on the interval [A,B]

fft 1-dimensional FFT

fftn take an N-dimensional FFT

fftnshift shift an N-dimensional FFT

ifftn inverse of an N-dimensional FFT

reshape(A, B) takes matrix B and shapes it into a vector of size A

isosurface(A,B,C,D,V) creates a plot of surface geometry of volume data D in the space of A,B,C of all values equal to V.

B

```
% Author: Karl Marrett
% HW1 AMATH 482
% DUE: Jan 22

clear all; close all; clc;
load Testdata;  %loads Undata
[row,col]=size(Undata);
slices = 20;
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);

% visualize k/ks space
figure(1)
subplot(2,1,1);
plot(1:length(k), k, 'k+');
xlabel('Fourier mode index');
ylabel('Fourier mode value');
title('Plot of nonshifted frequency modes (k)', 'fontsize', 70);
subplot(2,1,2);
plot(1:length(ks), ks, 'k-');
xlabel('Fourier mode index');
ylabel('Fourier mode value');
title('Plot of shifted frequency modes sdf(ks)', 'fontsize', 70);
saveas(1,'kPlot', 'png');
combinedFreq = zeros(n, n, n);

% create grids
[X, Y, Z] = meshgrid(x, y, z);
[Kx,Ky,Kz]=meshgrid(k,k,k);

for j=1:slices % each time point
raw(:, :, :, j)=reshape(Undata(j,:),n,n,n);
% n-dim fft for each time slice
rawFFT(:, :, :, j) = fftn(raw(:, :, :, j));
 % sum FFT for each slice
combinedFreq = combinedFreq + rawFFT(:, :, :, j);
end
```

```
sumFreq = abs(combinedFreq);
% max over all three dims
[maxVal, linearIndex] = max(sumFreq(:));
 % get index of max
[Kx_0, Ky_0, Kz_0] = ind2sub(size(sumFreq), linearIndex);
maxFreqModes = k([Kx_0, Ky_0 Kz_0]) % get value of max
normFreq = sumFreq/maxVal;  % normalize all values to 1

% Visualize signature frequency
[Kxs,Kys,Kzs]=meshgrid(ks,ks,ks); % for plotting in shifted space
% iterate over some values for isosurface
% thresh = linspace(.5, .93, 6);
% for j = 1 : length(thresh)
%     pause(3); close all;
%     isosurface(Kxs, Kys, Kzs, fftshift(normFreq), thresh(j))
%     title(strcat('value: ', int2str(thresh(j))));
%     axis([-5 5 -5 5 -5 5]), grid on, rotate3d, drawnow
% end

figure(2)
isosurface(Kxs, Kys, Kzs, fftshift(normFreq),.62, 'k+');
axis([-5 5 -5 5 -5 5]), grid on, rotate3d, drawnow
title('Averaged frequency spectrum');
xlabel('Shifted k modes (x) axis');
ylabel('Shifted k modes (y) axis');
zlabel('Shifted k modes (z) axis');
saveas(2,'freqCluster', 'png');

% build a meshgrid gaussian filter
bandwidth = 1;
% centered at max frequency components
meshFilter = exp(-bandwidth * ((Kx-Kx(Kx_0, Ky_0, Kz_0)).^2 + ...
(Ky-Ky(Kx_0, Ky_0, Kz_0)).^2 + (Kz-Kz(Kx_0, Ky_0, Kz_0)).^2));

% Filter out unwanted frequencies and recreate volume data
for j = 1:slices
filteredFFT(:, :, :, j) = rawFFT(:, :, :, j) .* meshFilter ;
%recreate in volume space must be positive
filtered(:, :, :, j) = abs(ifftn(filteredFFT(:, :, :, j)));
slice = filtered(:, :, :, j);
% max of volume data gives marble location
[M, linearIndex] = max(slice(:));
```

```
% convert to matrix subscripts
meshCoords = ind2sub(size(slice), linearIndex);
marbleX(j) = X(meshCoords);
marbleY(j) = Y(meshCoords);
marbleZ(j) = Z(meshCoords);
end

figure(3);
plot3(marbleX, marbleY, marbleZ, 'Linewidth', 2);
rotate3d, grid on, axis tight
title('Marble spatial location'); xlabel('x axis'); ylabel('y axis');
zlabel('z axis');
saveas(3,'trajectory', 'png');

% find location at time index
timeIndex = 20;
marbleLocation = [marbleX(timeIndex) marbleY(timeIndex) marbleZ(timeIndex)];
```

References

[1] A Fenster and D B Downey. Three-dimensional ultrasound imaging. *Annual review of biomedical engineering*, 2:457–475, 2000.

[2] JN Kutz. AMATH 301 Beginning Scientific Computing. *Department of Applied Mathematics, Box*, 2003, 2005.