AMATH 582 Homework 1

Leon Yan

January 24, 2020

Abstract

This report serves to illustrate how we denoise the data by averaging at frequency domain through multi-dimensional Fast-Fourier-Transform(FFT), given the pre-condition that there was only a center frequency in each measurement data. The relevant application is to break up the marble in our dog fluffy's intestines. The advantage of averaging over the frequency domain rather than the time domain, is that in the averaging frequency domain method we can solve the "moving object" detection successfully. For the cull materials including code and homework description, please go to url: `https://github.com/unitraveleryy/Amath582HW`

1 Introduction and Overview

In this homework, we are asked to detect the marble which was accidentally swallowed by our dog fluffy. Since the complex and dynamic environment in fluffy's intestines, the marble is moving all the time. Our goal is to target and break up the marble with the 20 different noisy ultrasound measurement data we have. What we presume is that the ideal response data should be a single bump in time series, the Fourier transform of which is a single frequency peak.

In this report, we solve the detection problem by denoising the data to track the marble's trajectory.

First, we averaging the measurement data over frequency domain. This approach not only enables us to cancel the white noise in each measurement, but also helps us to track the non-stationary object. The traditional approach of averaging over the time domain can only capture the stationary objects in the scenarios where we emit some specific frequency signals to detect things.

Second, based on the average values we have, we pick out the frequency corresponding the peak value as the frequency signature (center frequency). After that, we use a multi-dimension Gaussian filter around that frequency signature to filter out the signal, i.e, multiply out entry-wisely of the filter and the Fourier transform of each measurement signal.

In the last step, to track and plot the marble's trajectory, we choose the peak value of the data in our inverse Fourier Transform result to represent the position of the marble in our fluffy's 3d intestines' environment.

1.1 Data Format Description

The data is stored in **Testdata.mat**, in which each row represents a single measurement. The measurement is discretized 3d data, the inputs spans $[-15, 15]$ in x,y,z axes, with the same resolution, 65 sample points evenly distributed (including both end points). A demonstration of the raw data is plotted in Figure 1.

2 Theoretical Background

2.1 Multi-dimension Fast Fourier Transform (FFT)

As we learned from our textbook [2], Fourier introduced the concept of representing a given function $f(x)$ by a trigonometric series of sines and cosines:

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

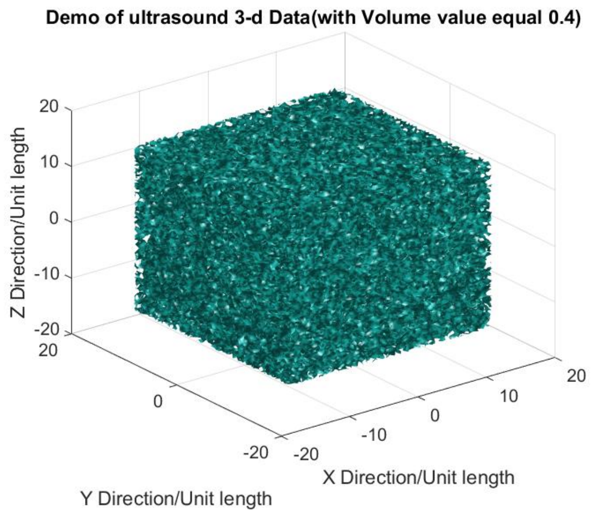

Figure 1: A demonstration of the raw ultrasound data in 3d spatial format.

By the orthogonality among the cos & sin basis, we can compute the coefficients of each term by doing the inner product of $f(x)$ with each basis respectively.

For the "Fast" feature mentioned before, is that we utilize the algorithm idea divide-and-conquer to let the computation be more efficient by making the number of the sample points to be the power of 2.

To promote the FFT in Multi-dimension case, we could do the FFT for each dimension separately, the sequence in doing so does not matter, an example of 2-D FFT for the image compression application can be found in Chapter 2.6 in [1].

2.2 Multi-dimension Gaussian Distribution Filter Design

The probability density distribution of the Multivariate normal distribution can be expressed in the equation below:

$$f(x) = (2\pi)^{-\frac{n}{2}} \det(\sum)^{\frac{1}{2}} e^{-\frac{1}{2}(x-\mu)^T \sum^{-1}(x-\mu)} \tag{2}$$

where $x \in \mathbb{R}^n$, $f(x)$ is the corresponding probability density value; the $\mu \in \mathbb{R}^n$ is the mean vector, and the $\sum \in \mathbb{R}^{n\times n}$ is the covariance matrix.

In our report, we simplified the computation of the probability density into a independent distribution among all the random variables, i.e., the covariance matrix $\sum$ is a diagonal matrix. Moreover, each variable has the same standard deviation. Then, specifically, our computation for the designed Gaussian like filter in 3d wave number spaces can be expressed in terms of below:

$$f(K_x, K_y, K_z) = e^{-\tau((K_x-f_x)^2+(K_y-f_y)^2+(K_z-f_z)^2)} \tag{3}$$

where K_x, K_y, K_z represent the 3d wave number; f_x, f_y, f_z is the coordinate of the center frequency of the Gaussian filter in 3d frequency domain. The τ (in the report $\tau = 1$) is the general standard deviation for K_x, K_y, K_z.

3 Algorithm Implementation and Development

Below is my algorithm implementation and development:

1. Load 20 measurement data,
2. Iterate each measurement, get the multi-dimension FFT analysis of each,
3. Average the frequency domain components, and find the center frequency,

4. Design a Gaussian filter around the center frequency,
5. Iterate each measurement, filter out the nosie information in data,
6. Find the index of peak value in measurement data to represent the marble's position,
7. Plot the marble's trajectory.

Algorithm 1: Denoising Multi-dimension Signal by FFT Algorithm

```
Input: Ultrasound data from Testdata.mat
Output: Marble trajectory
  Initialisation, data preparation :
  $U_{nt}$ for storing the FFT results of all measurement data
  for $j = 1 : 20$ do
    Extract $j^{th}$ measurement from Undata
    Do the multi-FFT for this data and add to the $U_{nt}$
  end for
  Average the $U_{nt}$ to obtain the mean value of the measurement data
  Find the center frequency coordinate $(f_x, f_y, f_z)$ among the average data
  Design the Gaussian shape filter around the center frequency
  for $j = 1 : 20$ do
    Extract $j^{th}$ measurement from Undata
    Filter out this data to obtain $U$
    Find the coordinate $(x, y, z)$ corresponding the peak value in $U$
    Store the coordinate $(x, y, z)$ to represent the marble position in this measurement
  end for
  Plot the marble trajectory based on the coordinates $(x, y, z)$ gathered.
```

4 Computational Results

Below are the computation results for three questions in the homework.

1. Through averaging of the spectrum, determine the frequency signature (center frequency) generated by the marble.

See the center frequency data in Table 1.

2. Filter the data around the center frequency determined above in order to denoise the data and determine the path of the marble.

The marble trajectory is in Figure 2.

3. Where should an intense acoustic wave be focused to breakup the marble at the 20th data measurement.

See the marble position in 20th measurement data in Table 3.

Besides the required homework questions, I have plotted the normalization result of the accumulative frequency distribution. You can see in Figure 3, in which the peak value can be easily identified.

kx	ky	kz
1.8850	-1.0472	0

Table 1: center frequency coordinate

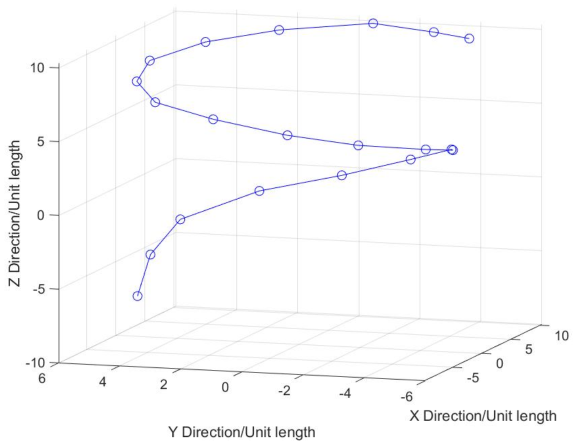

Figure 2: Marble trajectory plotted in 3d spatial format, which is obtained by the filtered ultrasound data.

index	x	y	z
1	4.6875	-4.6875	10.3125
2	8.4375	-2.8125	9.8438
3	10.3125	-0.4688	9.8438
4	8.90625	2.34375	9.3750
5	6.09375	4.21875	8.90625
6	1.40625	5.15625	8.4375
7	-3.28125	4.6875	7.96875
8	-7.5000	3.28125	7.5000
9	-9.84375	0.9375	7.03125
10	-9.3750	-1.40625	6.09375
11	-7.03125	-3.28125	5.15625
12	-2.8125	-4.6875	4.21875
13	1.8750	-4.6875	3.28125
14	6.5625	-3.7500	2.34375
15	9.3750	-1.8750	0.9375
16	9.84375	0.46875	-0.46875
17	7.96875	2.8125	-1.40625
18	4.21875	4.6875	-2.8125
19	-0.9375	4.6875	-4.21875
20	-5.6250	4.2188	-6.0938

Table 2: Marble's trajectory spatial coordinates.

x	y	z
-5.6250	4.2188	-6.0938

Table 3: marble position in 20th measurement

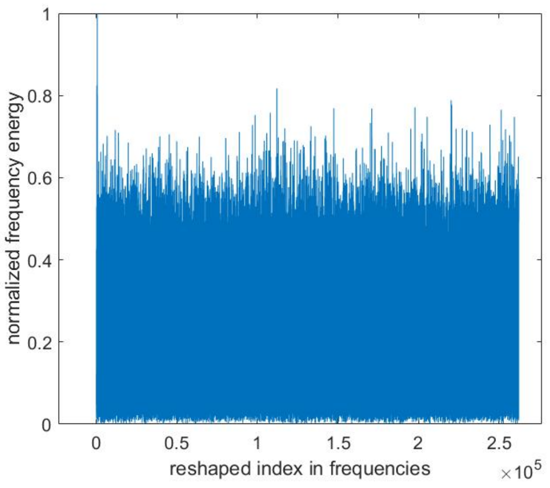

Figure 3: Averaged and normalized frequency energy distribution .

5 Summary and Conclusions

By choosing the peak value in our measurement data, whcih is filtered out by our designed Gaussian filter around the center frequency, we can find that the marble is spirally going down inside our dog fluffy's intestines. The averaging action on the measurement frequency domain prove to make sure the white noise gradually cancel out as the measurement times increase. Further, the averaging action on the frequency domain proves to be eligible to detect the moving object.

References

[1] Steven L Brunton and J Nathan Kutz. *Data-driven science and engineering: Machine learning, dynamical systems, and control.* Cambridge University Press, 2019.

[2] Jose Nathan Kutz. *Data-driven modeling & scientific computation: methods for complex systems & big data.* Oxford University Press, 2013.

Appendix A MATLAB Functions

Add your important MATLAB functions here with a brief implementation explanation. This is how to make an **unordered** list:

- `k=(2*pi/(2*L))*[0:(n/2-1) -n/2:-1]` rescales the wavenumbers by $2 * \pi/L$ since the FFT assumes $2 * \pi$ periodic signals.
- `[X,Y] = meshgrid(x,y)` returns 2-D grid coordinates based on the coordinates contained in the vectors `x` and `y`. `X` is a matrix where each row is a copy of `x`, and `Y` is a matrix where each column is a copy of `y`. The grid represented by the coordinates `X` and `Y` has `length(y)` rows and `length(x)` columns.
- `Y = ffn(X)` returns the multidimensional Fourier transform of an N-D array using a fast Fourier transform algorithm. The N-D transform is equivalent to computing the 1-D transform along each dimension of X. The output Y is the same size as X.
- `X = ifftn(Y)` returns the multidimensional discrete inverse Fourier transform of an N-D array using a fast Fourier transform algorithm. The N-D inverse transform is equivalent to computing the 1-D inverse transform along each dimension of Y. The output X is the same size as Y.

- `Y = abs(X)` returns the absolute value of each element in array X.
- `[x,y]=max(_)` finds the indices of the maximum values and returns them in array y, using any of the input arguments in the previous syntaxes. If the largest value occurs multiple times, the index of the first occurrence is returned.
- `fftshift(),ifftshift()` were not used due to the reason I prefer in that way I can write less code and may have the less probability to make the wavenumbers' sequence consistent.
- I do not use the ks, i.e, shifted wavenumbers to generate the meshgrid, but with k instead, this will be consistent with the wavenumbers in the Gaussian filter sequence.

Appendix B MATLAB Code

```
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);
[Kx,Ky,Kz]=meshgrid(k,k,k);

% frequency spectrum averaging
Unt_accumulate = zeros(n,n,n);
for j=1:20
    Un(:,:,:)=reshape(Undata(j,:),n,n,n);
    Unt = fftn(Un);
    Unt_accumulate = Unt_accumulate + Unt;
end

Unt_average = Unt_accumulate./20;


% build the frequency filter
[~, idx] = max(abs(Unt_average(:)));
Kx_tmp = Kx(:); fx = Kx_tmp(idx);
Ky_tmp = Ky(:); fy = Ky_tmp(idx);
Kz_tmp = Kz(:); fz = Kz_tmp(idx);
tau = 1;
filter = exp(-tau*((Kx - fx).^2 + (Ky - fy).^2 + (Kz - fz).^2));

% initializing the memory
X_tmp = X(:); Y_tmp = Y(:); Z_tmp = Z(:);
X_record = zeros(1,20); Y_record = zeros(1,20); Z_record = zeros(1,20);

% filtering each meausrement
for j =1:20
    Un(:,:,:)=reshape(Undata(j,:),n,n,n);
    Unt = fftn(Un);

    Unt_filter = Unt .* filter;
    Un_filter = ifftn(Unt_filter);
    [~, idx] = max(abs(Un_filter(:)));
    x = X_tmp(idx); y = Y_tmp(idx); z = Z_tmp(idx);
    X_record(j) = x; Y_record(j) = y; Z_record(j) = z;
end

figure(); plot3(X_record,Y_record,Z_record,'-o','Color','b','MarkerSize',10);
grid on
set(gca,'FontSize',15)
xlabel('X Direction/Unit length');
ylabel('Y Direction/Unit length');
zlabel('Z Direction/Unit length');
```

Listing 1: MATLAB code for signal denoising and plotting.