AMATH482 HW01

Joshua Borgman

University of Washington

Josh Borgman

Thursday, January 22, 2015

Contents

[1. Introduction and Overview 3](#_Toc409734417)

[2. Theoretical Background 3](#_Toc409734418)

[Fourier Transform 3](#_Toc409734419)

[Data Filtering 3](#_Toc409734420)

[Averaging: 3](#_Toc409734421)

[(Gaussian) Band pass filtering: 3](#_Toc409734422)

[3. Algorithm Implementation and Development 3](#_Toc409734423)

[(Inverse) Fast-Fourier Transform (IFFT/FFT): 3](#_Toc409734424)

[FFTShift: 3](#_Toc409734425)

[4. Computational Results 4](#_Toc409734426)

[5. Summary and Conclusions 4](#_Toc409734427)

[Appendix A: 5](#_Toc409734428)

[Appendix B: 5](#_Toc409734429)

# Introduction and Overview

The utilities of signal processing are far reaching and varied in their implementation and theory. The practicality of the associated techniques and algorithms are evident in this problem as we apply concepts of Fourier Transformation and Signal Filtering to the problem of a theoretical swallowed marble. These concepts are used to filter image data and locate the spatial location of the marble.

# Theoretical Background

## Fourier Transform

The Fourier Transform is an operation what decomposes a signal into an orthogonal basis of sinusoidal functions. The transform transports the data set between time and frequency domains. While traditionally based on a 2π interval, the transform can be extended to any arbitrary interval length by rescaling the wavenumbers by. Fourier Transforms allow for the simplification of more complicated operations into linear operations in frequency space.

## Data Filtering

This problem examines two main types of filtering, namely filtering by averaging and application of a (Gaussian) band pass filter.

### Averaging:

The idea behind averaging of a signal is centered on the belief that the noise in the image is truly random. With this idea averaging several realizations of a data series can effectively eliminate the random variation as the number of realizations grows.

### (Gaussian) Band pass filtering:

A band pass filler limits the frequency content of a signal to a specified range by weighting frequencies, favoring a particular range and effectively silencing the rest. While the shape of a band pass filter can be selected to fit a desired application, a common selection is the Gaussian curve which causes a quick but non-discontinuous fall off.

# Algorithm Implementation and Development

## (Inverse) Fast-Fourier Transform (IFFT/FFT):

The fast Fourier transform is an algorithm designed to compute the discrete Fourier transform (DFT). The algorithm computes the transform by factorizing the DFT matrix into several sparsely populated factors. The popularization of this algorithm comes in the quick operation time (O(Nlog(N))). The FFT takes our data set from a time domain to a frequency domain. The IFFT does the operation in reverse and returns us to the time domain.

## FFTShift:

The fftshift command centers the zero frequency component of the transformed data to the center of the array. This command is utilized largely for the purpose of visualizing a Fourier transform.

# Computational Results

The center frequency depicted below is obtained via the averaging of the data to remove the surrounding noise.

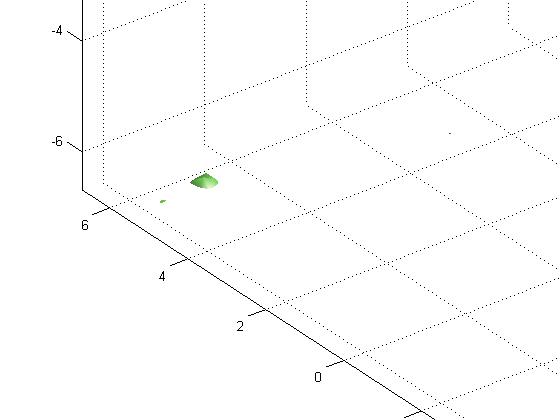


Figure 1: This is a visualization of the isosurface generated in the frequency space of the averaged data.

# Summary and Conclusions

The ubiquity of signal processing gives great utility to the application of filtering and associated algorithms. Highlighted in this problem we see the efficiency of the expressed transformations as well as detail needed in the implementation of these algorithms.

# Appendix A:

# Appendix B:

% Joshua Borgman, AMATH 482, 22 January 2015, Homework 1

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; %defines linear sample space

k=(2\*pi/(2\*L))\*[0:(n/2-1) -n/2:-1]; ks=fftshift(k); %Adjusts transform domain from 2pi -> 2L

%% Averaging Try One

close all

u = Undata;

uave = zeros(n,n,n);

for jj = 1:size(u,1)

ur = reshape(u(jj, :),n,n,n);

urt = fftn(ur);

uave = uave + urt;

end

ave = abs(uave)./20;

aven = ave/max(max(max(ave)));

[X,Y,Z]=meshgrid(x,y,z);

[Kx,Ky,Kz]=meshgrid(ks,ks,ks);

figure, isosurface(X,Y,Z,abs(aven),0.7)

axis([-20 20 -20 20 -20 20]),grid on, drawnow

figure, isosurface(Kx,Ky,Kz,abs(aven),0.7)

grid on, drawnow

[a, b, c] = ind2sub(size(aven), find(aven == max(max(max(aven)))));

cenX = Kx(a,b,c); cenY = Ky(a,b,c); cenZ = Kz(a,b,c);

%% Generate Filter...

filter = exp(-0.2\*((Kx - cenX).^2+(Ky - cenY).^2+(Kz - cenZ).^2));

ft = fftshift(fftn(filter));

%% Apply Filter and Plot in Space Domain

close all

Undata1(:,:,:) = reshape(Undata(1,:),n,n,n);

dataf = fftn(Undata1).\*ft;

datafin = ifftn(dataf);

figure, isosurface(X,Y,Z,abs(datafin),0.8)

axis([-20 20 -20 20 -20 20]),grid on, drawnow