Table of Contents


```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%
% MATLAB Project 2
%   Author: Mary Oh
%   Date: 2016/10/13
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%
```

```
Error using evalin
Undefined function 'LastnameFirstname_Project2' for input arguments of
 type 'double'.
```

Initialization

```
close all;
clear all;
clc;

ASUID = 6; % Input the last digit of your ASUID number.
```

Question 1

Q1a

```
% Load image - DO NOT CHANGE THIS CODE
origIm = imread('rocky.jpg'); % Load the chosen image and save it to
 origIm
% DO NOT CHANGE THE ABOVE LINE OF CODE
```

```
origIm = rgb2gray(origIm);

% Plot
figure(1) % Make figure 1 the active figure.
imshow(origIm) % Show the image

title('Q1a: Original Image') % Entitling the plot
```

Q1b

```
% Initialize Variables
blurSize = 90; % Size of the square point-spread matrix
blurWidth = 4; % The width of the gaussian

% Create point-spread function to blur the original image
h = fspecial('gaussian', blurSize, blurWidth);
xs = imfilter(origIm, h,'replicate'); % use built in filter function

% Blur the original image
blurIm = imfilter(origIm, h, 'replicate');

% Plot figure
figure (2);
subplot(1,3,1)
imshow(origIm) % Show the original Image
title('Original Image');

subplot(1,3,2) % Show the point-spread function
imshow(h, [0,max(h(:))]);title('Point-Spread Function');

subplot(1,3,3)
imshow(xs); title('Blurred Image'); % Show the blurred image
```

Q1c

```
blurSize = 90;
width = 15 - (ASUID/2);

i = fspecial('gaussian', blurSize, width);
s = imfilter(origIm, i,'replicate'); % use built in filter function

% Blur the original image
blurIm1 = imfilter(origIm, i, 'replicate');

% Plot figure
figure (2);
subplot(1,3,1)
imshow(origIm) % Show the original Image
title('Original Image');

subplot(1,3,2) % Show the point-spread function
imshow(i, [0,max(i(:))]);title('Point-Spread Function');
```

```
subplot(1,3,3)
imshow(s); title('Blurred Image with ASUID 6'); % Show the blurred
 image
```

Q1d

```
figure(3);

resIm1 = deconvreg(blurIm, h);

subplot(2,2,1)
imshow(xs); title('Blurred Image')

subplot(2,2,2)
imshow(resIm1); title('Restored Image: Regularized deconvolution');

resIm2 = deconvblind(blurIm, ones(size(h)));

subplot(2,2,3)
imshow(xs); title('Blurred Image')

subplot(2,2,4)
imshow(resIm2); title('Restored Image: Blind deconvolution');

figure(4);

resIm3 = deconvreg(blurIm1, i);

subplot(2,2,1)
imshow(s); title('Blurred Image with ASUID 6')

subplot(2,2,2)
imshow(resIm3); title('Restored Image: Regularized deconvolution');

resIm4 = deconvblind(blurIm1, ones(size(i)));

subplot(2,2,3)
imshow(s); title('Blurred Image with ASUID 6')

subplot(2,2,4)
imshow(resIm4); title('Restored Image: Blind deconvolution');
```

Q1e

```
size = 20 + (ASUID*5);
width1 = 4;

j = fspecial('gaussian', size, width1);
t = imfilter(origIm, j,'replicate'); % use built in filter function

% Blur the original image
blurIm2 = imfilter(origIm, j, 'replicate');
```

```
% Plot figure
figure (5);
subplot(1,3,1)
imshow(origIm) % Show the original Image
title('Original Image');

subplot(1,3,2) % Show the point-spread function
imshow(j, [0,max(j(:))]);title('Point-Spread Function');

subplot(1,3,3)
imshow(t); title('Blurred Image with ASUID 6'); % Show the blurred
 image

figure (6);

resIm5 = deconvreg(blurIm2, j);
resIm6 = deconvblind(blurIm2, ones(size(j)));

subplot(1,3,1)
imshow(t)
title('Blurred Image with ASUID 6')

subplot(1,3,2)
imshow(resIm5)
title('Restored Image: Regularized deconvolution');

subplot(1,3,3)
imshow(resIm6)
title('Restored Image: Blind deconvolution');
```

Question 2

Q2a

```
% Initialize Variables
startTime = -25-ASUID;
endTime = 45-ASUID;

n = startTime:endTime;

x = zeros(size(n));

% Define x
x(abs(n) == 5 ) = 0.5;
x(abs(n) == 4 ) = 2;
x(abs(n) == 3 ) = -1;
x(abs(n) == 2 ) = 0;
x(abs(n) == 1) = 1.5;
x(n == 0) = 3;

y = zeros(size(n));
```

```
tic;
for ii = 1:length(n)
    if 4 > ii > 2 % for i=1 cannot look back in time , i.e. there is
 no x(0)
        y(ii) = 4*x(ii) + 2*x(ii-2) - (1/3)*x(ii+2) + (3/4)*y(ii-2);
    end

    if (ii > 4)
        y(ii) = 4*x(ii) + 2*x(ii-2) - (7/3)*x(ii-4) - (1/3)*x(ii+2) +
 (3/4)*y(ii-2) - (1/2)*y(ii-4);
    else
        y(ii) = 4*x(ii);
    end
end

figure (7);

subplot(2,1,1)
stem(n,x)
xlabel('n')
xlim([startTime endTime])
ylabel('amplitude')
ylim([-1.5 3.5])
title('x[n]')

subplot(2,1,1)
stem(n,y)
xlabel('n')
xlim([startTime endTime])
ylabel('amplitude')
ylim([-1.5 3.5])
title('y[n]')
```

Q2b

Question 3

Q3a

Initialize

```
T = 7*pi/2;

t = -T/7:0.001:T/7;

cycles = 3;

x = zeros(size(t));

% Define x
x = 7*t.^2/(2*T);
```

```
sigx = repmat(x,1,cycles); % repeats the signal the assigned number of
 cycles;
t1 = linspace(-T*cycles/2,T*cycles/2,length(sigx)); % creates time
 vector to plot.

figure (8);
plot(t1,sigx)
title('Q3 Perodic Function')
xlabel('t, seconds')
ylabel('Amplitude')

T = T
amp = max(sigx) - min(sigx)
% ylim([0,1])
```

Q3b

```
T = (ASUID/2) + 7;

t = -T/7:0.001:T/7;

cycle = 5;

y = zeros(size(t));

% Define x
y = 7*t.^2/(2*T);

sigy = repmat(y,1,cycle); % repeats the signal the assigned number of
 cycles;
t2 = linspace(-T*cycle/2,T*cycle/2,length(sigy)); % creates time
 vector to plot.

figure (9);
plot(t2,sigy)
title('Q3 Perodic Function y(t)')
xlabel('t, seconds')
ylabel('Amplitude')
```

Q3c

```
ao = (1/T) * (7/(2*T)) * ((1/3)*((T/7)^3 - (-T/7)^3))
```

Q3d

```
N = 50;
k = -N:N; % vector of "k" coefficients.
ak = zeros(size(k));
omega = 2*pi/T; % frequency
dt = t(2) - t(1); % time delta

for n = 1:length(k)
```

```
    if k(n) == 0
        ak(n) = ao;
    else
        ak(n) = (1/T)*sum(y.*exp(-1i*k(n)*omega*t)*dt); % Simulates
 the integral of the analysis equation
    end
end

figure(10);

subplot(2,1,1);
real(ak)
plot(real(ak))
xlabel('N')
ylabel('ak')
xlim([0 100])
title('Real')

subplot(2,1,2)
imag(ak)
plot(imag(ak))
xlabel('N')
ylabel('ak')
xlim([0 100])
title('Imaginary')
```

Q3e

```
% The following code is the core of your synthesis equation to
% reconstruct your yN(t) signal.
for n = 1:length(t)
    yN(n) = sum(ak.*exp(1i*k*omega*t(n))); % Summation of the
 synthesis equation

    if n == 10
        y10 = yN(10)
    end
    if n == 25
        y25 = yN(25)
    end
    if n == 50
        y50 = yN(50)
    end
    if n == 100
        y100 = yN(100)
    end
    if n == 200
        y200 = yN(200)
    else
        y = 0
    end
end
```

```
figure (11);

subplot(5,2,1)
real(y10)
stem(10,real(y10))
xlabel('n')
ylabel('yN')
title('yN(10) - real')

subplot(5,2,2)
imag(y10)
stem(10,imag(y10))
xlabel('n')
ylabel('yN')
title('yN(10) - imaginary')

subplot(5,2,3)
real(y25)
stem(25,real(y25))
xlabel('n')
ylabel('yN')
title('yN(25) - real')

subplot(5,2,4)
imag(y25)
stem(25,imag(y25))
xlabel('n')
ylabel('yN')
title('yN(25) - imaginary')

subplot(5,2,5)
real(y50)
stem(50,real(y50))
xlabel('n')
ylabel('yN')
title('yN(50) - real')

subplot(5,2,6)
imag(y50)
stem(50,imag(y50))
xlabel('n')
ylabel('yN')
title('yN(50) - imaginary')

subplot(5,2,7)
real(y100)
stem(100,real(y100))
xlabel('n')
ylabel('yN')
title('yN(100) - real')

subplot(5,2,8)
imag(y100)
stem(100,imag(y100))
```

```
xlabel('n')
ylabel('yN')
title('yN(100) - imaginary')

subplot(5,2,9)
real(y200)
stem(200,real(y200))
xlabel('n')
ylabel('yN')
title('yN(200) - real')

subplot(5,2,10)
imag(y200)
stem(200,imag(y200))
xlabel('n')
ylabel('yN')
title('yN(200) - imaginary')
```

Q3f

The following code is the calculation for MSE.

```
MSE = sum(abs(err.^2))/(dt);

% End of File
```

Published with MATLAB® R2016b

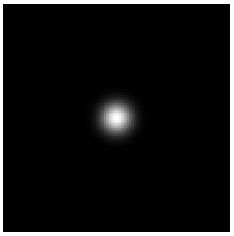
Q1A

Q1a: Original Image

Q1B

Original Image

Point-Spread Function

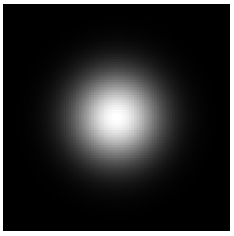


Blurred Image

Q1C

Original Image

Point-Spread Function



Blurred Image with ASUID 6

Q1D

Blurred Image

Restored Image: Regularized deconvolution

Blurred Image

Restored Image: Blind deconvolution

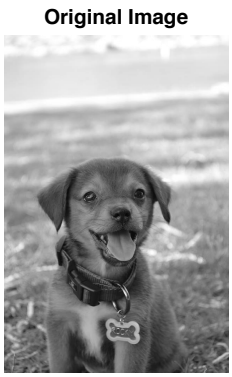
Blurred Image with ASUID 6

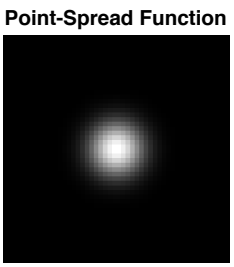
Restored Image: Regularized deconvolution

Blurred Image with ASUID 6

Restored Image: Blind deconvolution

Q1E



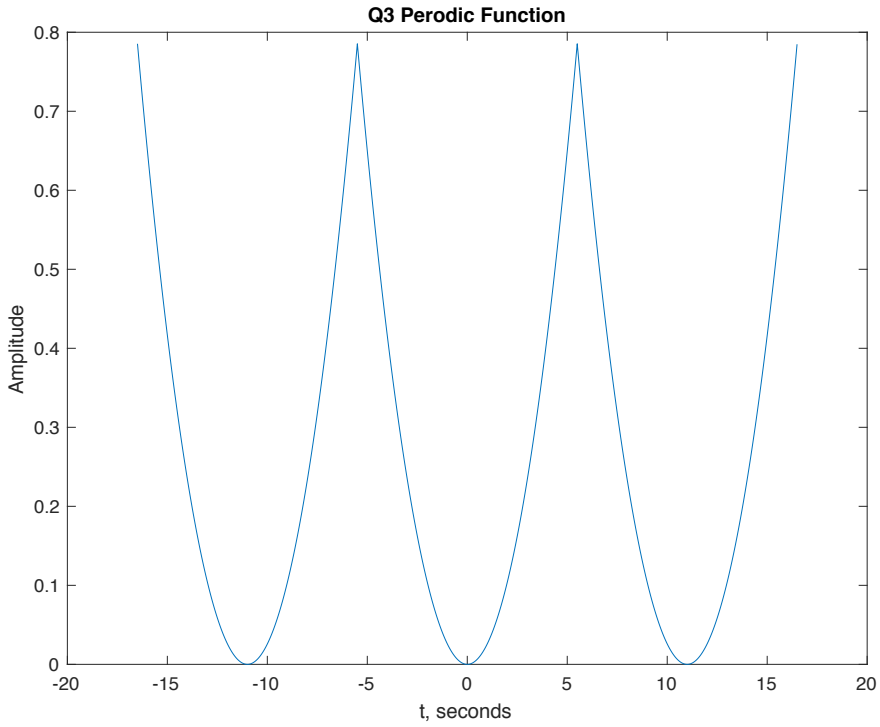


2nd plot error – not showing any plots/images

Q2A
Error - Subscript indices must either be real positive integers or logicals.

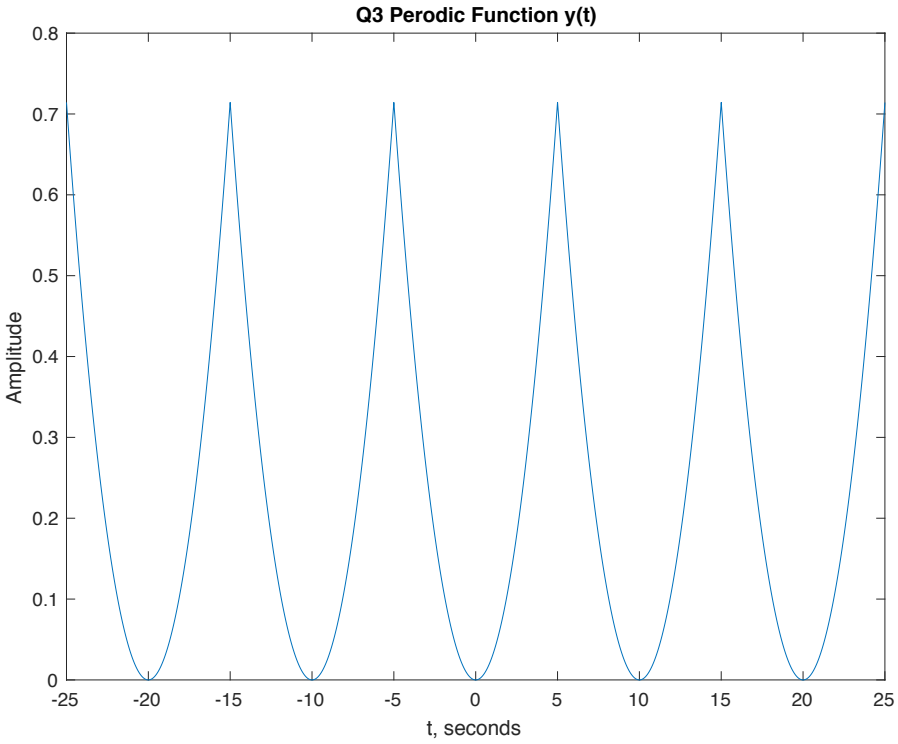
Q2B
System is acasual because it has non-zero value when t>0.

Q3A



T = 10.9956
amp = 0.7854

Q3B



Q3C

$$ao = \frac{1}{T}\int_{-T/7}^{T/7}\frac{7t^2}{2T}dt$$

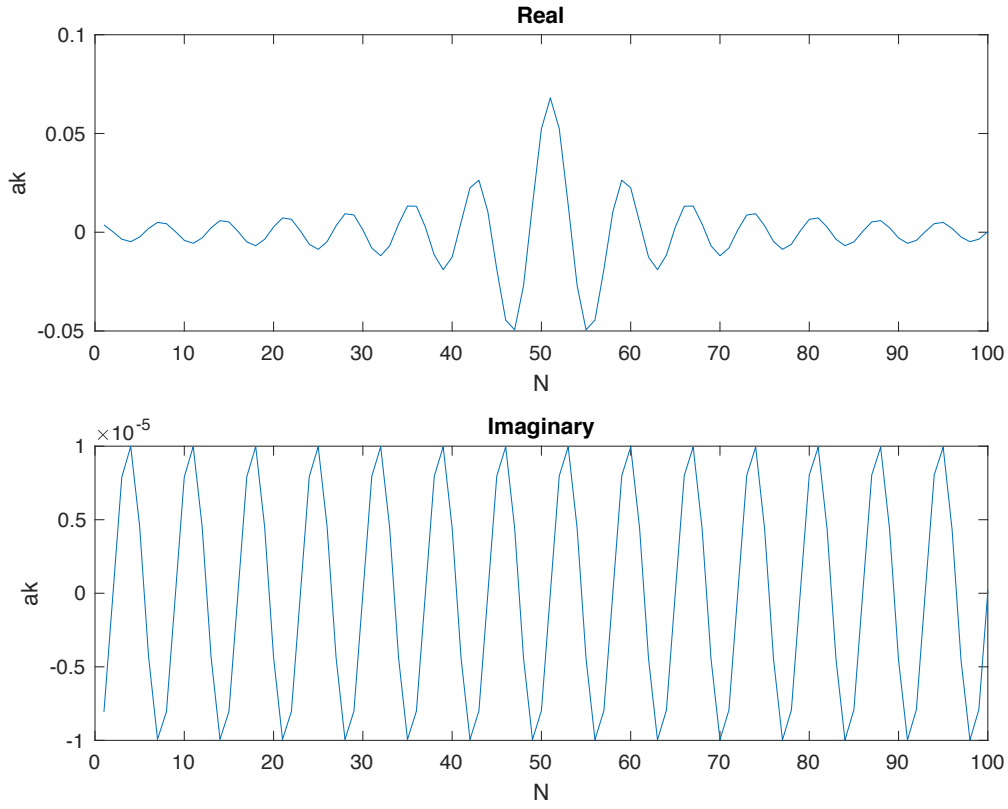
$$ao = \frac{7}{2T^2}\int_{-T/7}^{T/7}t^2dt$$

$$ao = \frac{7}{2T^2}\left(\frac{t^3}{3}\right)_{-T/7}^{T/7}$$

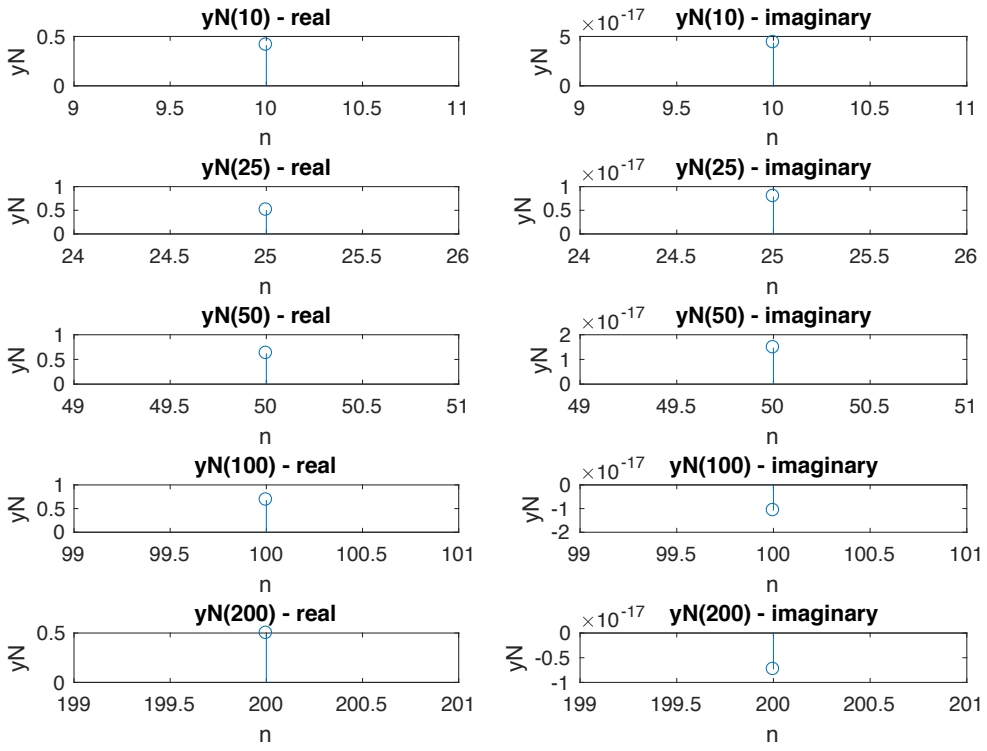
$$ao = \frac{7}{2T^2}\left(\frac{\left(\frac{T}{7}\right)^3}{3} - \frac{\left(\frac{-T}{7}\right)^3}{3}\right)$$

ao = 0.0680

Q3D



Q3E


yN(10) - real
yN(10) - imaginary
yN(25) - real
yN(25) - imaginary
yN(50) - real
yN(50) - imaginary
yN(100) - real
yN(100) - imaginary
yN(200) - real
yN(200) - imaginary
yN
n
×10^-17

Q3F