ECE3051 – Analog and Digital Signal Processing, Fall Semester 2022-2023

ELA DA – 1, Slot: L25-L26

By: Jonathan Rufus Samuel (20BCT0332) Date: 21.8.2022

ELA DA 1 – DOS: 21.08.2022

Task - 1: GENERATION OF ELEMENTARY SIGNALS AND SYSTEM ANALYSIS

Q1) Generate the elementary signals that are employed for characterization of random signals. Also, generate a sinusoidal signal and subject the same to the following basic signal processing operations. a. Time Shifting / Delaying (TD) b. Folding / Reflection (FD) c. Verify, whether TD[FD]=FD[TD] d. Convolution e. Correlation Illustrate the above operations with relevant waveforms.

CODE:

```
%Task - 1: GENERATION OF ELEMENTARY SIGNALS AND SYSTEM ANALYSIS
%Name: Jonathan Rufus Samuel (20BCT0332)
%Course: ECE3051 - ELA

%SubTask 1 - Generate the elementary signals that are employed for
characterization of random
%signals. Also, generate a sinusoidal signal and subject the same to the following
basic
%signal processing operations
%a. Time Shifting / Delaying (TD)
%b. Folding / Reflection (FD)
%c. Verify, whether TD[FD]=FD[TD]
%d. Convolution
%e. Correlation
%Illustrate the above operations with relevant waveforms.

%Elementary Functions:
%Zer0 Function
a = zeros(1 ,5);
b = a;
subplot(331),stem(b);
title('Zeros Function');
xlabel('Time');
ylabel('Magnitude');
grid;

%Ones Function
a = ones(1 ,5);
b = a;
subplot(332),stem(b);
title('Ones Function');
xlabel('Time');
```

```
ylabel('Magnitude');
grid;

%Impulse Function
a = zeros(1 ,4);
b = [a,1,a];
subplot(333),stem(b);
title('Impulse Function');
xlabel('time');
ylabel('Magnitude');
grid;

%Unit Step Function
a = zeros(1 , 4);
b = ones(1, 4);
c = [a,1,b];

t = (-1:0.01:5)';
% Start with all zeros:
unitstep = zeros(size(t));
% But make everything corresponding to t>=1 one:
unitstep(t>=0) = 1;
plot(t,unitstep,'b','linewidth',3)
% Repeat, with everything shifted to the right by 1 unit:
unitstep2 = zeros(size(t));
unitstep2(t>=2) = 1;
hold on
plot(t,unitstep2,'r:','linewidth',2)

subplot(334),stem(c);
title('Unit Step Function');
xlabel('time');
ylabel('Magnitude');
grid;

%Sine Function
a = 0:15:360;
b = sind(a);
subplot(335),stem(b);
title('Sine Function');
xlabel('time');
ylabel('Magnitude');
grid;

%Cosine Function
a = 0:15:360;
b = cosd(a);
subplot(336),stem(b);
title('Cosine Function');
xlabel('time');
ylabel('Magnitude');
grid;

%Exponential Signal
a = 1:20;
b = exp(a);
subplot(337),stem(b); %plot(b)
title('Exponential Function');
xlabel('time');
```

```
ylabel('Magnitude');
grid;

%Ramp Function
a = 1:10;
b = a;
subplot(338),stem(b);
title('Ramp Function');
xlabel('time');
ylabel('Magnitude');
grid;

%Parabolic Signal
a = linspace(-5,5,10);
b = 0.5*a.^2;
subplot(339),stem(a,b);
title('Parabolic Function');
xlabel('time');
ylabel('Magnitude');
grid;


%Generate a sinusoidal signal and subject the same to the following basic
a = 0:15:360;
b = sind(a);
subplot(331),stem(b);
title('Base Sinusoidal Function');
xlabel('time');
ylabel('Magnitude');
grid;

%signal processing operations:
%a. Time Shifting / Delaying (TD)
a = 0:15:360;
b = sind(a+30);
subplot(332),stem(b);
title('Time Delayed Sinusoidal Function');
xlabel('time');
ylabel('Magnitude');
grid;

%b. Folding / Reflection (FD)
a = 0:15:360;
b = sind(-a);
subplot(333),stem(b);
title('Folded Sinusoidal Function');
xlabel('time');
ylabel('Magnitude');
grid;

%c. Verify, whether TD[FD]=FD[TD]
a = 0:15:360;
x1 = sind(-sind(a+30));
y1 = sind(sind(-a-30));
subplot(211),stem(x1);
title('FD[TD] Function');
xlabel('time');
ylabel('Magnitude');
grid;
```

```
subplot(212),stem(y1);
title('TD[FD] Function');
xlabel('time');
ylabel('Magnitude');
grid;

%d. Convolution
Fs = 10000;
Ts = 1/Fs;
fc = 1000;
Tc = 1/fc;
t = 0:Ts:Tc;
% LTI impulse response h(t) = exp(-1000*t)
h = exp(-1000*t);
% angular frequency w = 2*pi*fc
w = 2*pi*fc;
% the signal x(t) = sin(wc*t)
x = sin(w*t);
% convolution of x(t) and h(t)
y = conv(x,h,'same');
subplot(3, 1, 1);
plot(t, h, 'LineWidth', 2);
grid on;
xlabel('t');
ylabel('h');
subplot(3, 1, 2);
plot(t, x, 'LineWidth', 2)
grid on;
xlabel('t');
ylabel('x');
subplot(3, 1, 3);
plot(t, y, 'LineWidth', 2)
grid on;
hold on
stem(t,y)
xlabel('t');
ylabel('y = x**h');

%e. Correlation
a = 0:15:360;
x = sind(a+30);
y = sind(-a);
R = corrcoef(x,y);
disp("The Result is: ");
disp(R);
```

OUTPUT:

>> %Elementary Functions:

%Zer0 Function

a = zeros(1 ,5);

b = a;

subplot(331),stem(b);

title('Zeros Function');

```
xlabel('Time');
ylabel('Magnitude');
grid;

%Ones Function
a = ones(1 ,5);
b = a;
subplot(332),stem(b);
title('Ones Function');
xlabel('Time');
ylabel('Magnitude');
grid;

%Impulse Function
a = zeros(1 ,4);
b = [a,1,a];
subplot(333),stem(b);
title('Impulse Function');
xlabel('time');
ylabel('Magnitude');
grid;

%Unit Step Function
a = zeros(1 , 4);
b = ones(1, 4);
c = [a,1,b];

t = (-1:0.01:5)';
% Start with all zeros:
unitstep = zeros(size(t));
% But make everything corresponding to t>=1 one:
```

```
unitstep(t>=0) = 1;
plot(t,unitstep,'b','linewidth',3)
% Repeat, with everything shifted to the right by 1 unit:
unitstep2 = zeros(size(t));
unitstep2(t>=2) = 1;
hold on
plot(t,unitstep2,'r:','linewidth',2)

subplot(334),stem(c);
title('Unit Step Function');
xlabel('time');
ylabel('Magnitude');
grid;

%Sine Function
a = 0:15:360;
b = sind(a);
subplot(335),stem(b);
title('Sine Function');
xlabel('time');
ylabel('Magnitude');
grid;

%Cosine Function
a = 0:15:360;
b = cosd(a);
subplot(336),stem(b);
title('Cosine Function');
xlabel('time');
ylabel('Magnitude');
grid;
```

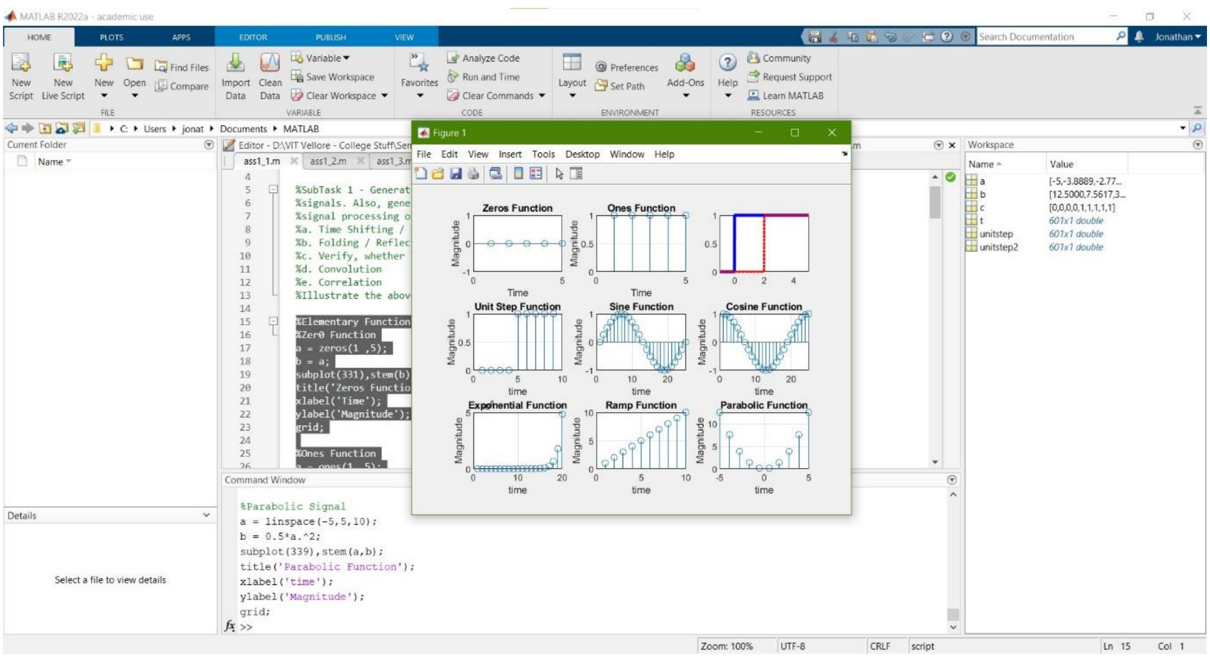
```
%Exponential Signal
a = 1:20;
b = exp(a);
subplot(337),stem(b); %plot(b)
title('Exponential Function');
xlabel('time');
ylabel('Magnitude');
grid;

%Ramp Function
a = 1:10;
b = a;
subplot(338),stem(b);
title('Ramp Function');
xlabel('time');
ylabel('Magnitude');
grid;

%Parabolic Signal
a = linspace(-5,5,10);
b = 0.5*a.^2;
subplot(339),stem(a,b);
title('Parabolic Function');
xlabel('time');
ylabel('Magnitude');
grid;
>> %Generate a sinusoidal signal and subject the same to the following basic
a = 0:15:360;
b = sind(a);
subplot(331),stem(b);
```

```
title('Base Sinusoidal Function');
xlabel('time');
ylabel('Magnitude');
grid;

%signal processing operations:
%a. Time Shifting / Delaying (TD)
a = 0:15:360;
b = sind(a+30);
subplot(332),stem(b);
title('Time Delayed Sinusoidal Function');
xlabel('time');
ylabel('Magnitude');
grid;

%b. Folding / Reflection (FD)
a = 0:15:360;
b = sind(-a);
subplot(333),stem(b);
title('Folded Sinusoidal Function');
xlabel('time');
ylabel('Magnitude');
grid;
>> %Generate a sinusoidal signal and subject the same to the following basic
a = 0:15:360;
b = sind(a);
subplot(311),stem(b);
title('Base Sinusoidal Function');
xlabel('time');
ylabel('Magnitude');
grid;
```

```
%signal processing operations:
%a. Time Shifting / Delaying (TD)
a = 0:15:360;
b = sind(a+30);
subplot(312),stem(b);
title('Time Delayed Sinusoidal Function');
xlabel('time');
ylabel('Magnitude');
grid;

%b. Folding / Reflection (FD)
a = 0:15:360;
b = sind(-a);
subplot(313),stem(b);
title('Folded Sinusoidal Function');
xlabel('time');
ylabel('Magnitude');
grid;
>> %c. Verify, whether TD[FD]=FD[TD]
a = 0:15:360;
x1 = sind(-sind(a+30));
y1 = sind(sind(-a-30));
subplot(211),stem(x1);
title('FD[TD] Function');
xlabel('time');
ylabel('Magnitude');
grid;
subplot(212),stem(y1);
title('TD[FD] Function');
xlabel('time');
```

```
ylabel('Magnitude');
grid;
>> %d. Convolution
Fs = 10000;
Ts = 1/Fs;
fc = 1000;
Tc = 1/fc;
t = 0:Ts:Tc;
% LTI impulse response h(t) = exp(-1000*t)
h = exp(-1000*t);
% angular frequency w = 2*pi*fc
w = 2*pi*fc;
% the signal x(t) = sin(wc*t)
x = sin(w*t);
% convolution of x(t) and h(t)
y = conv(x,h,'same');
subplot(3, 1, 1);
plot(t, h, 'LineWidth', 2);
grid on;
xlabel('t');
ylabel('h');
subplot(3, 1, 2);
plot(t, x, 'LineWidth', 2)
grid on;
xlabel('t');
ylabel('x');
subplot(3, 1, 3);
plot(t, y, 'LineWidth', 2)
grid on;
hold on
stem(t,y)
```

```
xlabel('t');
ylabel('y = x**h');

%e. Correlation
a = 0:15:360;
x = sind(a+30);
y = sind(-a);
R = corrcoef(x,y);
disp("The Result is: ");
disp(R);
The Result is:
   1.0000   -0.8575
  -0.8575    1.0000

>>
```



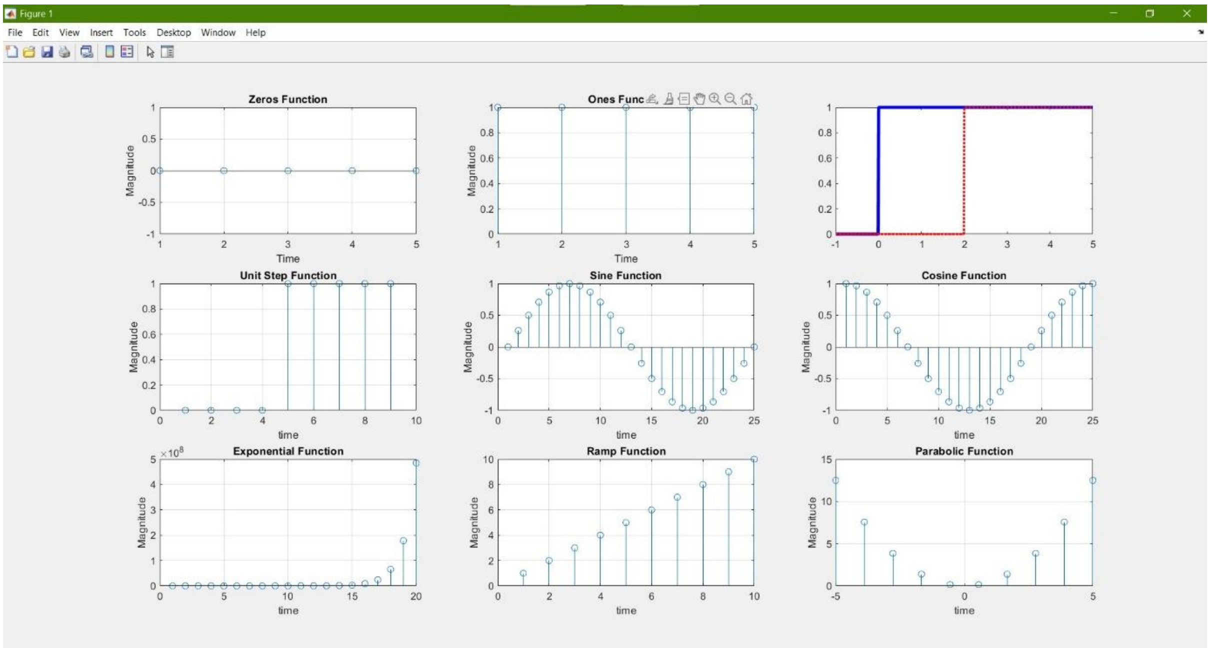
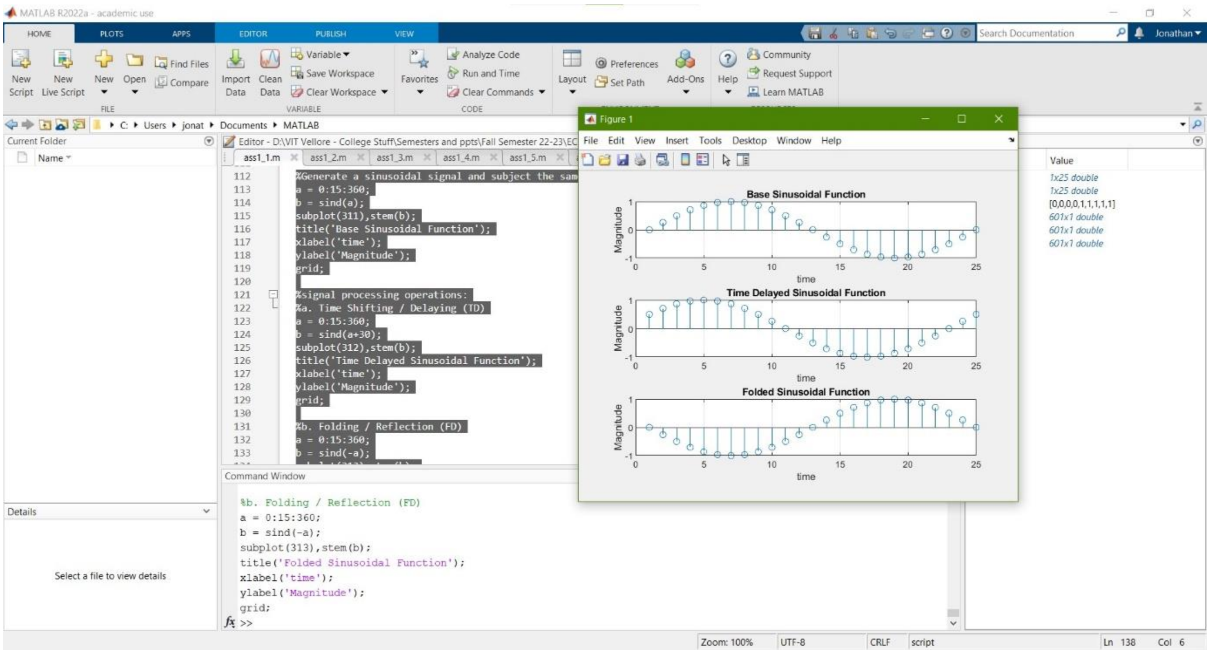


Figure 1
File Edit View Insert Tools Desktop Window Help
Zeros Function
Ones Function
Unit Step Function
Sine Function
Cosine Function
Exponential Function
Ramp Function
Parabolic Function
Magnitude
Time
time


MATLAB R2022a - academic use
HOME PLOTS APPS EDITOR PUBLISH VIEW
Current Folder
Details
Select a file to view details
Command Window
%Generate a sinusoidal signal and subject the sam
a = 0:15:360;
b = sind(a);
subplot(311),stem(b);
title('Base Sinusoidal Function');
xlabel('time');
ylabel('Magnitude');
grid;
%signal processing operations:
%a. Time Shifting / Delaying (TD)
a = 0:15:360;
b = sind(a+30);
subplot(312),stem(b);
title('Time Delayed Sinusoidal Function');
xlabel('time');
ylabel('Magnitude');
grid;
%b. Folding / Reflection (FD)
a = 0:15:360;
b = sind(-a);
subplot(313),stem(b);
title('Folded Sinusoidal Function');
xlabel('time');
ylabel('Magnitude');
grid;
Figure 1
Base Sinusoidal Function
Time Delayed Sinusoidal Function
Folded Sinusoidal Function
Magnitude
time

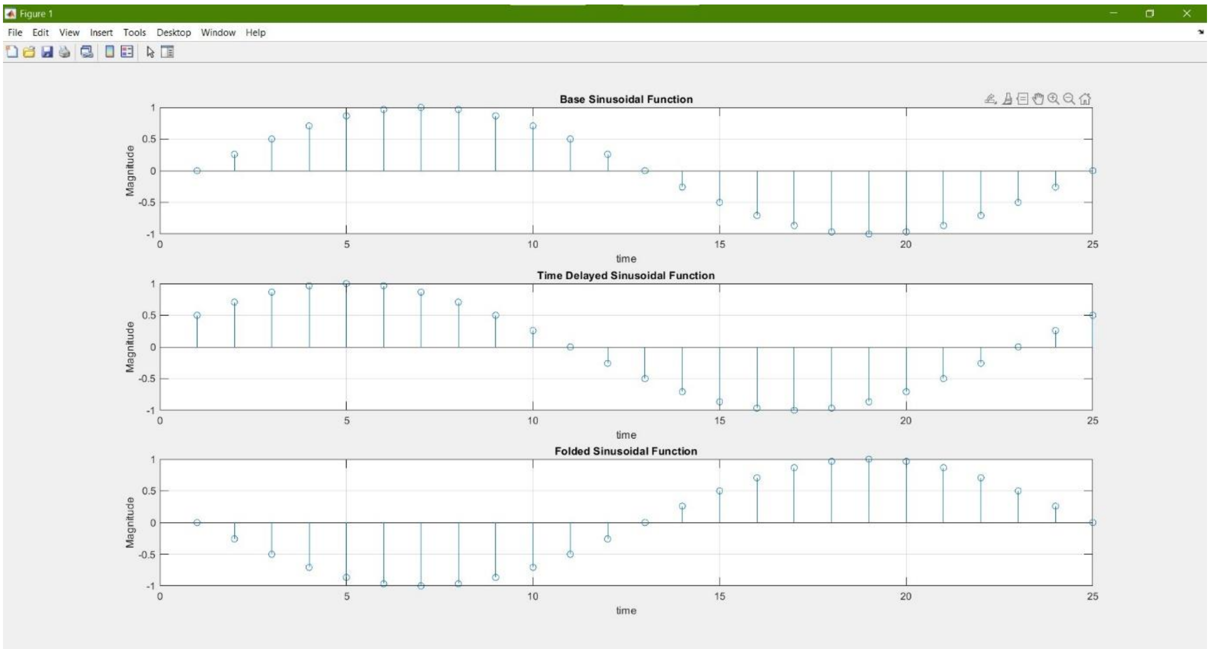
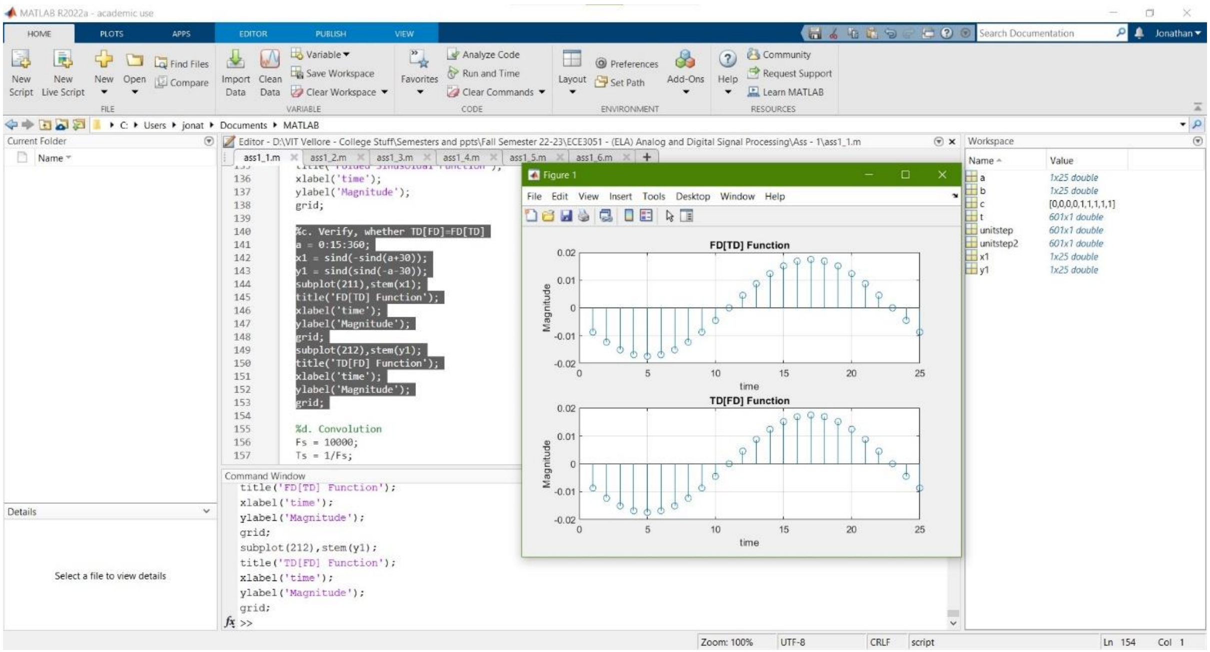
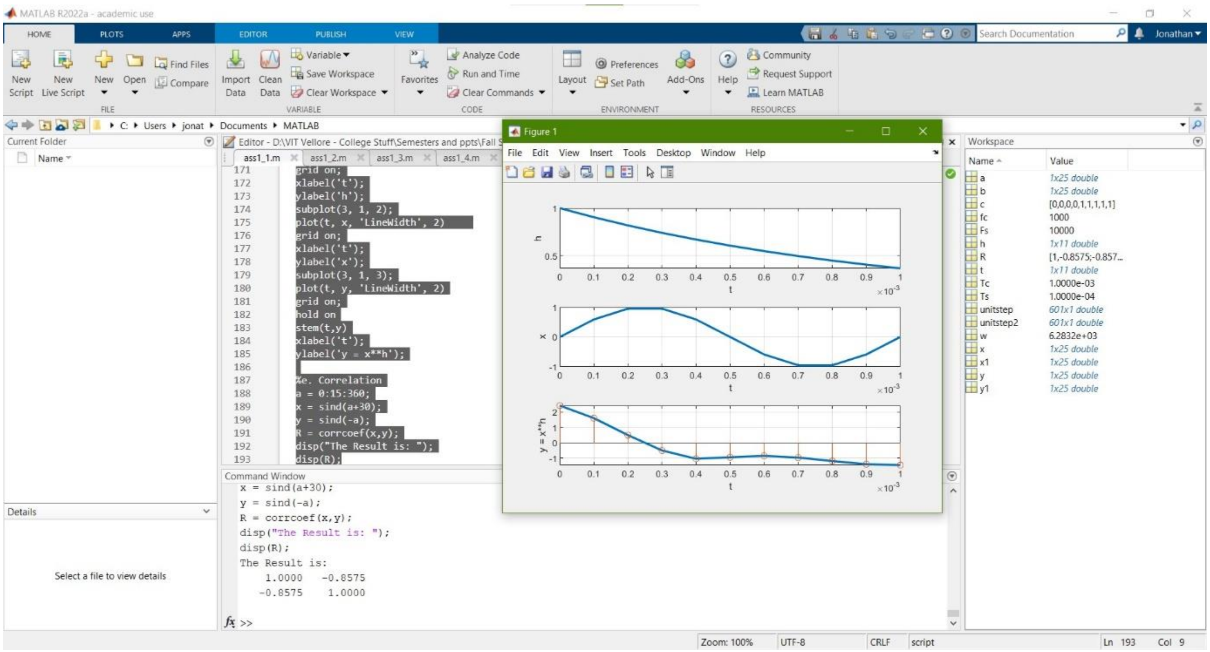


Figure 1
File Edit View Insert Tools Desktop Window Help
Base Sinusoidal Function
Magnitude
time
Time Delayed Sinusoidal Function
Magnitude
time
Folded Sinusoidal Function
Magnitude
time


MATLAB R2022a - academic use
Editor - D:\VIT Vellore - College Stuff\Semesters and ppts\Fall Semester 22-23\ECE3051 - (ELA) Analog and Digital Signal Processing\Ass - 1\ass1_1.m
xlabel('time');
ylabel('Magnitude');
grid;
%c. Verify, whether TD[FD]=FD[TD]
a = 0:15:360;
x1 = sind(-sind(a+30));
y1 = sind(sind(-a-30));
subplot(211),stem(x1);
title('FD[TD] Function');
xlabel('time');
ylabel('Magnitude');
grid;
subplot(212),stem(y1);
title('TD[FD] Function');
xlabel('time');
ylabel('Magnitude');
grid;
%d. Convolution
Fs = 10000;
Ts = 1/Fs;
Figure 1
FD[TD] Function
TD[FD] Function
Magnitude
time
Workspace
Select a file to view details


MATLAB R2022a - academic use
grid on;
xlabel('t');
ylabel('h');
subplot(3, 1, 2);
plot(t, x, 'LineWidth', 2)
grid on;
xlabel('t');
ylabel('x');
subplot(3, 1, 3);
plot(t, y, 'LineWidth', 2)
grid on;
hold on
stem(t,y)
xlabel('t');
ylabel('y = x**h');
%e. Correlation
a = 0:15:360;
x = sind(a+30);
y = sind(-a);
R = corrcoef(x,y);
disp("The Result is: ");
disp(R);
Command Window
x = sind(a+30);
y = sind(-a);
R = corrcoef(x,y);
disp("The Result is: ");
disp(R);
The Result is:
1.0000 -0.8575
-0.8575 1.0000
Figure 1

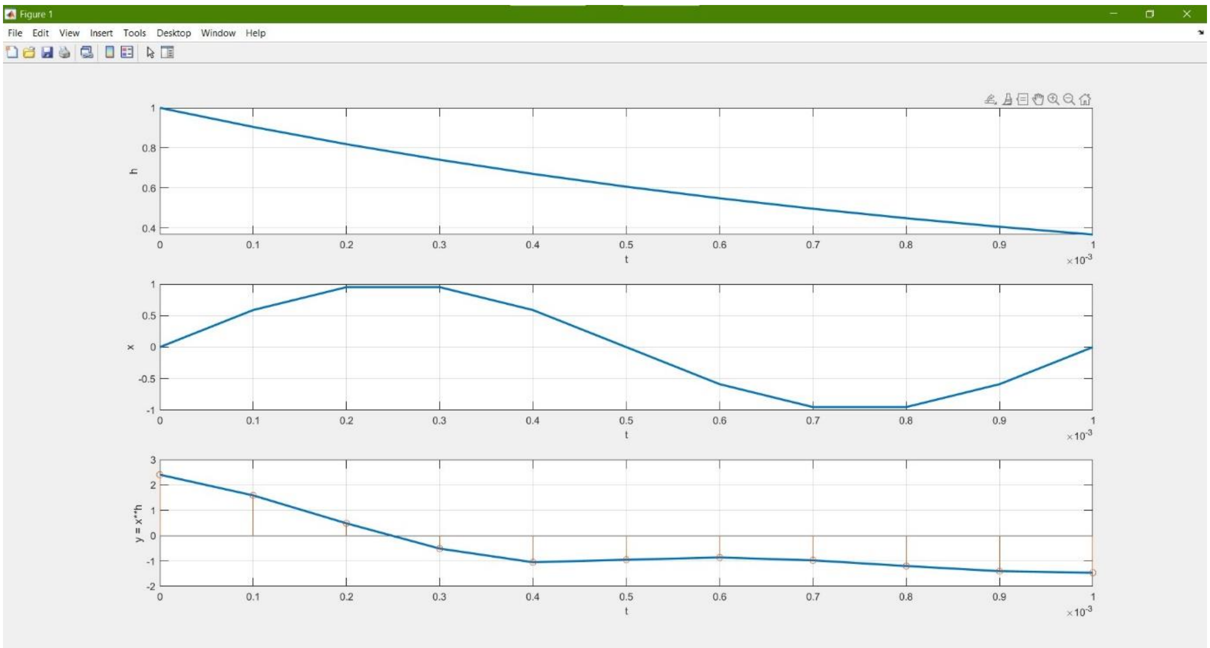


Figure 1
h
x
y = x**h
t
×10⁻³

Q2) Also, graphically verify whether the following system is linear / non-linear, Stable / unstable: $\boldsymbol{y}(\boldsymbol{n}) = \boldsymbol{y}\ \mathbf{2}\ (\boldsymbol{n} - \mathbf{1}) + \boldsymbol{x}(\boldsymbol{n})$, for the bounded input $\boldsymbol{x}(\boldsymbol{n}) = \boldsymbol{u}(\boldsymbol{n}) + \boldsymbol{u}(\boldsymbol{n} - \mathbf{2})$

CODE:

```
%SubTask 2 - graphically verify whether the following system is linear / non-
linear,
% Stable /unstable:

% 𝒚(𝒏) = 𝒚^2 (𝒏 − 𝟏) + 𝒙(𝒏), for the bounded input 𝒙(𝒏) = 𝒖(𝒏) + 𝒖(𝒏 − 𝟐)


n0 = -5:5; %range
syms x(n);
x(n) = heaviside(n)-heaviside(n-2);
syms y(n);
y(n) = ((n-1))^2 + x(n);
subplot(111),plot(y(n0),n0);
title('Sequence #1 - Stem');
xlabel('x(n)');
ylabel('n0');

%1) Condition for Linearity: Relationship between x & y is linear (straight
%line), and should cross the origin.
%Answer: NO, it is not Linear, as it does not pass through origin and does
%not satisfy superposition.

%2) Condition for Stability: Should Satisfy the BIBO stability condition.
B = isstable(y);
disp(B);
```

OUTPUT:

>> %SubTask 2 - graphically verify whether the following system is linear / non-linear,

% Stable /unstable:

% $\boldsymbol{y}(\boldsymbol{n}) = \boldsymbol{y}$^2 $(\boldsymbol{n} - \mathbf{1}) + \boldsymbol{x}(\boldsymbol{n})$, for the bounded input $\boldsymbol{x}(\boldsymbol{n}) = \boldsymbol{u}(\boldsymbol{n}) + \boldsymbol{u}(\boldsymbol{n} - \mathbf{2})$

n0 = -5:5; %range

syms x(n);

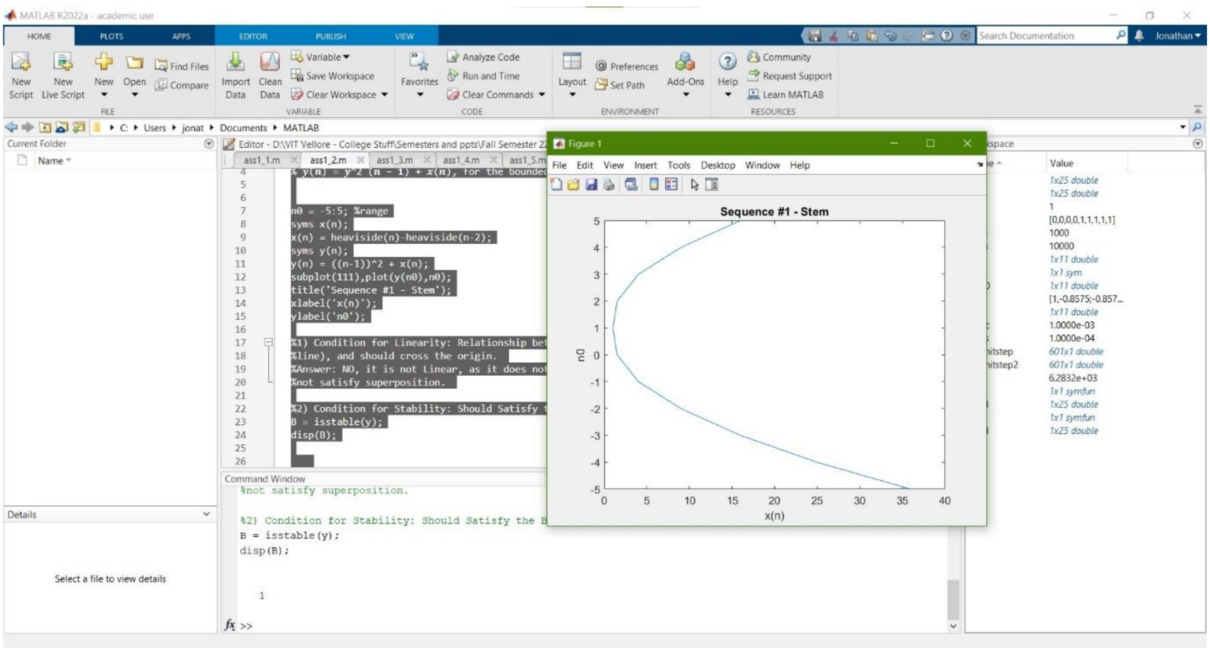
x(n) = heaviside(n)-heaviside(n-2);

syms y(n);

y(n) = ((n-1))^2 + x(n);

subplot(111),plot(y(n0),n0);

```
title('Sequence #1 - Stem');

xlabel('x(n)');

ylabel('n0');


%1) Condition for Linearity: Relationship between x & y is linear (straight
%line), and should cross the origin.
%Answer: NO, it is not Linear, as it does not pass through origin and does
%not satisfy superposition.


%2) Condition for Stability: Should Satisfy the BIBO stability condition.
B = isstable(y);
disp(B);



  1

>>
```



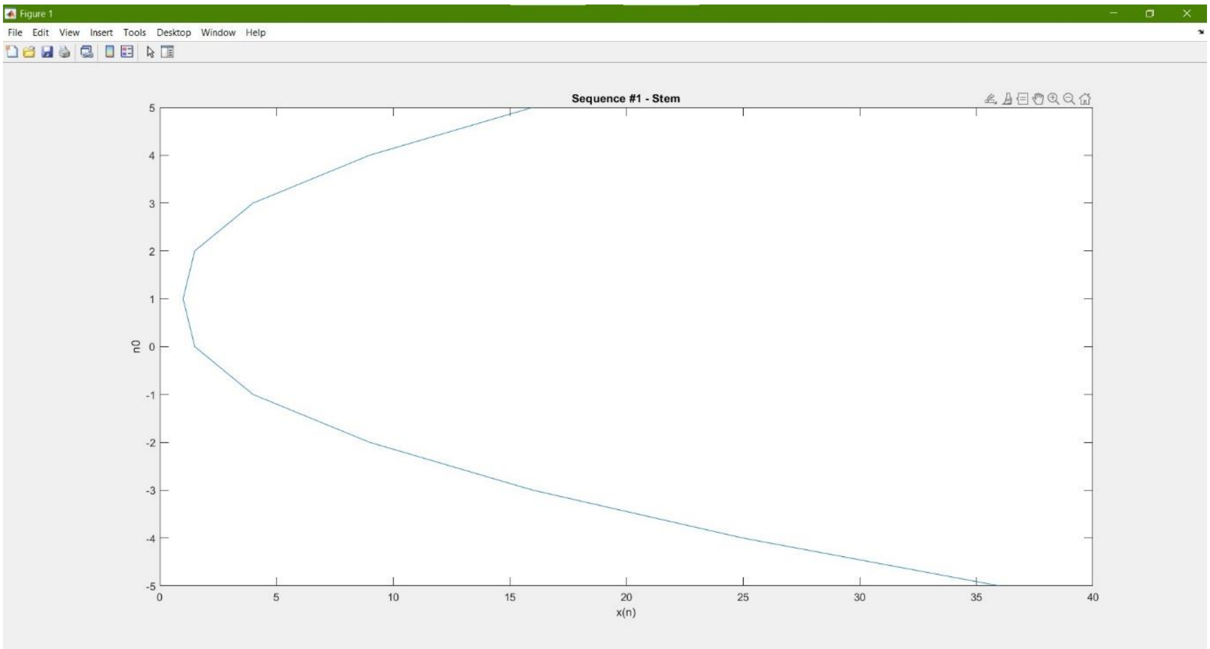


Figure 1
File Edit View Insert Tools Desktop Window Help
Sequence #1 - Stem
n0
x(n)
5
4
3
2
1
0
-1
-2
-3
-4
-5
0
5
10
15
20
25
30
35
40

Q3) Generate and plot each of the following sequences over the indicated interval.

a) x(n) = 2delta(n+2) - delta(n-4), -5<=n<=5

b) x(n) = n[u(n)-u(n-10)] + 10e^-0.3(n-10) [u(n-10) - u(n-20)]

c) x(n) = cos(0.04pi*n) + 0.2 * w(n), 0<=n<=50, where w(n) is a Gaussian Random Sequence with zero limit and unit variance.

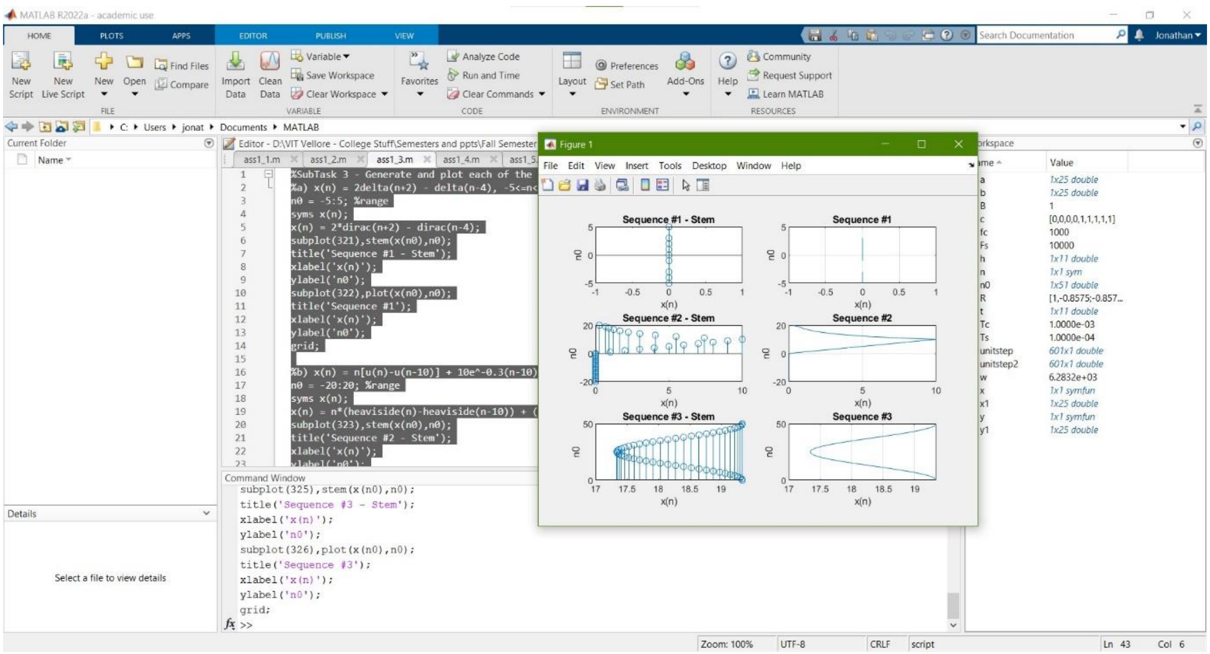
CODE:

```
%SubTask 3 - Generate and plot each of the following sequences over the indicated
interval.
%a) x(n) = 2delta(n+2) - delta(n-4), -5<=n<=5
n0 = -5:5; %range
syms x(n);
x(n) = 2*dirac(n+2) - dirac(n-4);
subplot(321),stem(x(n0),n0);
title('Sequence #1 - Stem');
xlabel('x(n)');
ylabel('n0');
subplot(322),plot(x(n0),n0);
title('Sequence #1');
xlabel('x(n)');
ylabel('n0');
grid;

%b) x(n) = n[u(n)-u(n-10)] + 10e^-0.3(n-10) [u(n-10) - u(n-20)]
n0 = -20:20; %range
syms x(n);
x(n) = n*(heaviside(n)-heaviside(n-10)) + (10*exp(-0.3*(n-10)) * (heaviside(n-10)
- heaviside(n-20)));
subplot(323),stem(x(n0),n0);
title('Sequence #2 - Stem');
xlabel('x(n)');
ylabel('n0');
subplot(324),plot(x(n0),n0);
title('Sequence #2');
xlabel('x(n)');
ylabel('n0');
grid;

%c) x(n) = cos(0.04pi*n) + 0.2 * w(n), 0<=n<=50, where w(n) is a Gaussian
%Random Sequence with zero limit and unit variance.
n0 = 0:50; %range
syms x(n);
x(n) = cos(0.04*pi*n) + 0.2*normrnd(0,50);%normrnd(n);
subplot(325),stem(x(n0),n0);
title('Sequence #3 - Stem');
xlabel('x(n)');
ylabel('n0');
subplot(326),plot(x(n0),n0);
title('Sequence #3');
xlabel('x(n)');
ylabel('n0');
grid;
```

OUTPUT:

```
>> %SubTask 3 - Generate and plot each of the following sequences over the indicated interval.
%a) x(n) = 2delta(n+2) - delta(n-4), -5<=n<=5
n0 = -5:5; %range
syms x(n);
x(n) = 2*dirac(n+2) - dirac(n-4);
subplot(321),stem(x(n0),n0);
title('Sequence #1 - Stem');
xlabel('x(n)');
ylabel('n0');
subplot(322),plot(x(n0),n0);
title('Sequence #1');
xlabel('x(n)');
ylabel('n0');
grid;

%b) x(n) = n[u(n)-u(n-10)] + 10e^-0.3(n-10) [u(n-10) - u(n-20)]
n0 = -20:20; %range
syms x(n);
x(n) = n*(heaviside(n)-heaviside(n-10)) + (10*exp(-0.3*(n-10)) * (heaviside(n-10) - heaviside(n-20)));
subplot(323),stem(x(n0),n0);
title('Sequence #2 - Stem');
xlabel('x(n)');
ylabel('n0');
subplot(324),plot(x(n0),n0);
title('Sequence #2');
xlabel('x(n)');
ylabel('n0');
grid;

%c) x(n) = cos(0.04pi*n) + 0.2 * w(n), 0<=n<=50, where w(n) is a Gaussian
```

```
%Random Sequence with zero limit and unit variance.

n0 = 0:50; %range

syms x(n);

x(n) = cos(0.04*pi*n) + 0.2*normrnd(0,50);%normrnd(n);

subplot(325),stem(x(n0),n0);

title('Sequence #3 - Stem');

xlabel('x(n)');

ylabel('n0');

subplot(326),plot(x(n0),n0);

title('Sequence #3');

xlabel('x(n)');

ylabel('n0');

grid;

>>
```



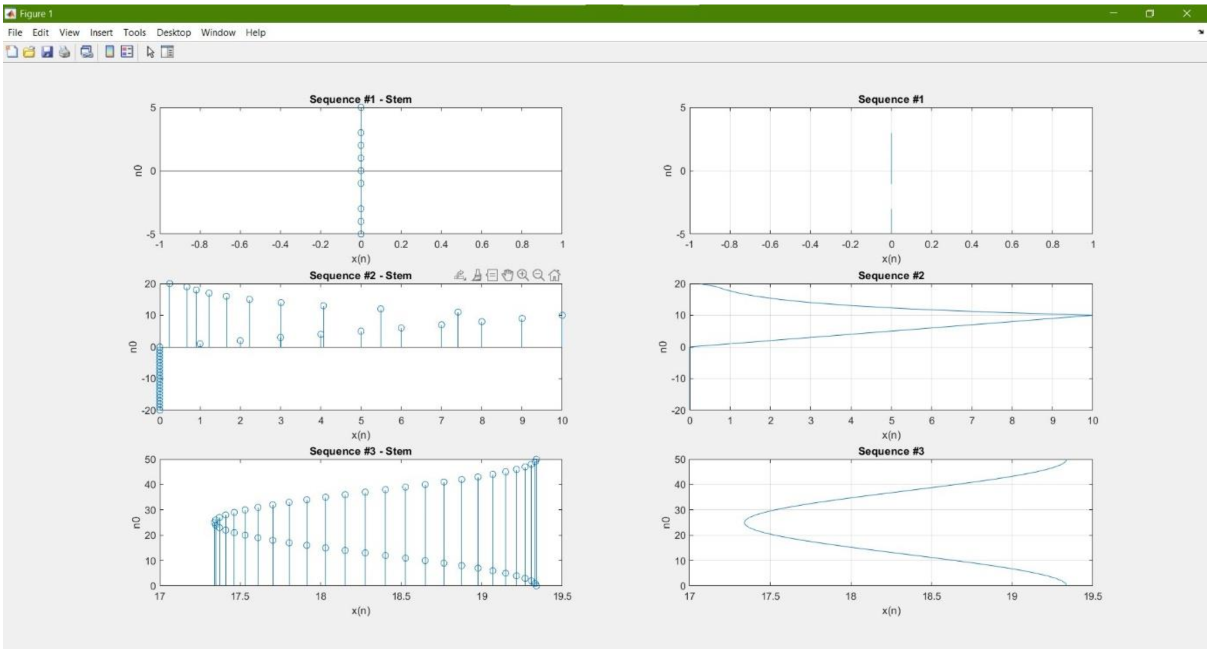


Figure 1
File Edit View Insert Tools Desktop Window Help
Sequence #1 - Stem
Sequence #1
Sequence #2 - Stem
Sequence #2
Sequence #3 - Stem
Sequence #3
n0
x(n)

Q4) Given the following difference equation:

y(n) - y(n-1) + 0.9y(n-2) = x(n); for all n

a) Calculate and plot the impulse response h(n) at n = -20,. . . ,100.

b) Calculate and plot the unit step response s(n) at n = -20, ..., 100.

c) Is the system specified by h(n) stable?

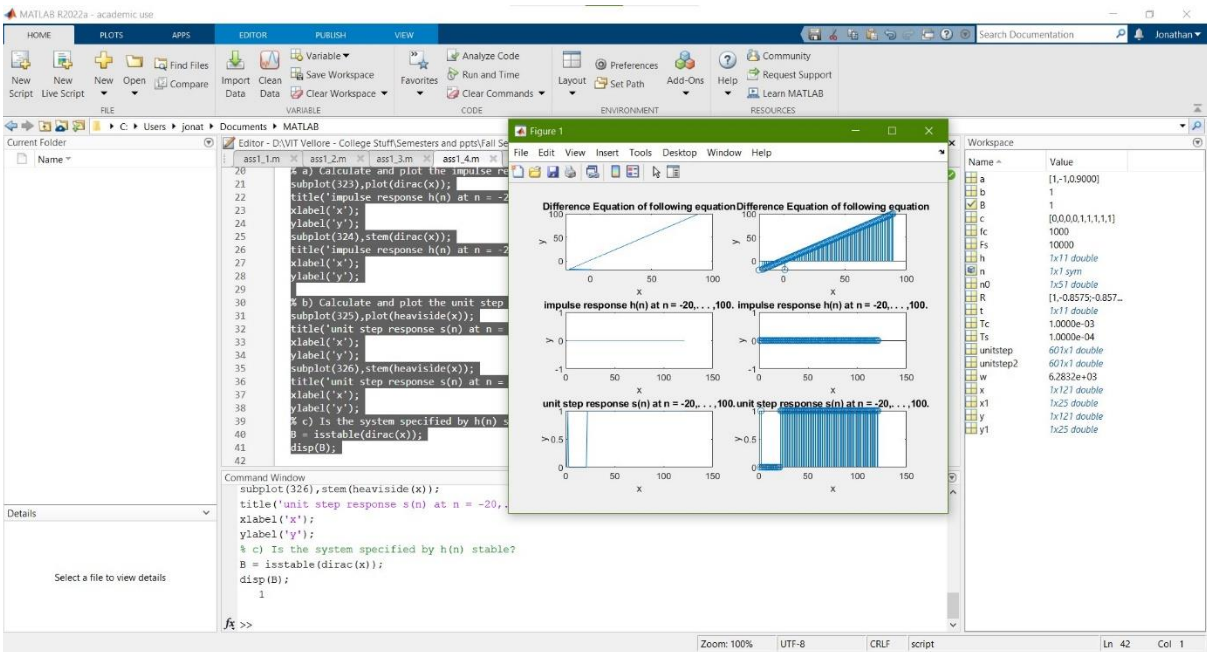
CODE:

```
%Given the following difference equation:
% y(n) - y(n-1) + 0.9y(n-2) = x(n); for all n

%Therefore: We need to use the filter function
%Let us take coefficients of both y (here a) and x (here b)

a = [1 -1 9/10];
b = 1;
y = -20:100;
x = filter(a,b,y);
subplot(321),plot(x,y);
title('Difference Equation of following equation');
xlabel('x');
ylabel('y');
subplot(322),stem(x,y);
title('Difference Equation of following equation');
xlabel('x');
ylabel('y');

% a) Calculate and plot the impulse response h(n) at n = -20,. . . ,100.
subplot(323),plot(dirac(x));
title('impulse response h(n) at n = -20,. . . ,100.');
xlabel('x');
ylabel('y');
subplot(324),stem(dirac(x));
title('impulse response h(n) at n = -20,. . . ,100.');
xlabel('x');
ylabel('y');

% b) Calculate and plot the unit step response s(n) at n = -20, ....., 100.
subplot(325),plot(heaviside(x));
title('unit step response s(n) at n = -20,. . . ,100.');
xlabel('x');
ylabel('y');
subplot(326),stem(heaviside(x));
title('unit step response s(n) at n = -20,. . . ,100.');
xlabel('x');
ylabel('y');
% c) Is the system specified by h(n) stable?
B = isstable(dirac(x));
disp(B);
```

OUTPUT:

```
>> %Given the following difference equation:
% y(n) - y(n-1) + 0.9y(n-2) = x(n); for all n

%Therefore: We need to use the filter function
%Let us take coefficients of both y (here a) and x (here b)

a = [1 -1 9/10];
b = 1;
y = -20:100;
x = filter(a,b,y);
subplot(321),plot(x,y);
title('Difference Equation of following equation');
xlabel('x');
ylabel('y');
subplot(322),stem(x,y);
title('Difference Equation of following equation');
xlabel('x');
ylabel('y');

% a) Calculate and plot the impulse response h(n) at n = -20,. . . ,100.
subplot(323),plot(dirac(x));
title('impulse response h(n) at n = -20,. . . ,100.');
xlabel('x');
ylabel('y');
subplot(324),stem(dirac(x));
title('impulse response h(n) at n = -20,. . . ,100.');
xlabel('x');
ylabel('y');

% b) Calculate and plot the unit step response s(n) at n = -20, ....., 100.
```

```
subplot(325),plot(heaviside(x));
title('unit step response s(n) at n = -20,. . . ,100.');
xlabel('x');
ylabel('y');
subplot(326),stem(heaviside(x));
title('unit step response s(n) at n = -20,. . . ,100.');
xlabel('x');
ylabel('y');
% c) Is the system specified by h(n) stable?
B = isstable(dirac(x));
disp(B);
  1


>>
```



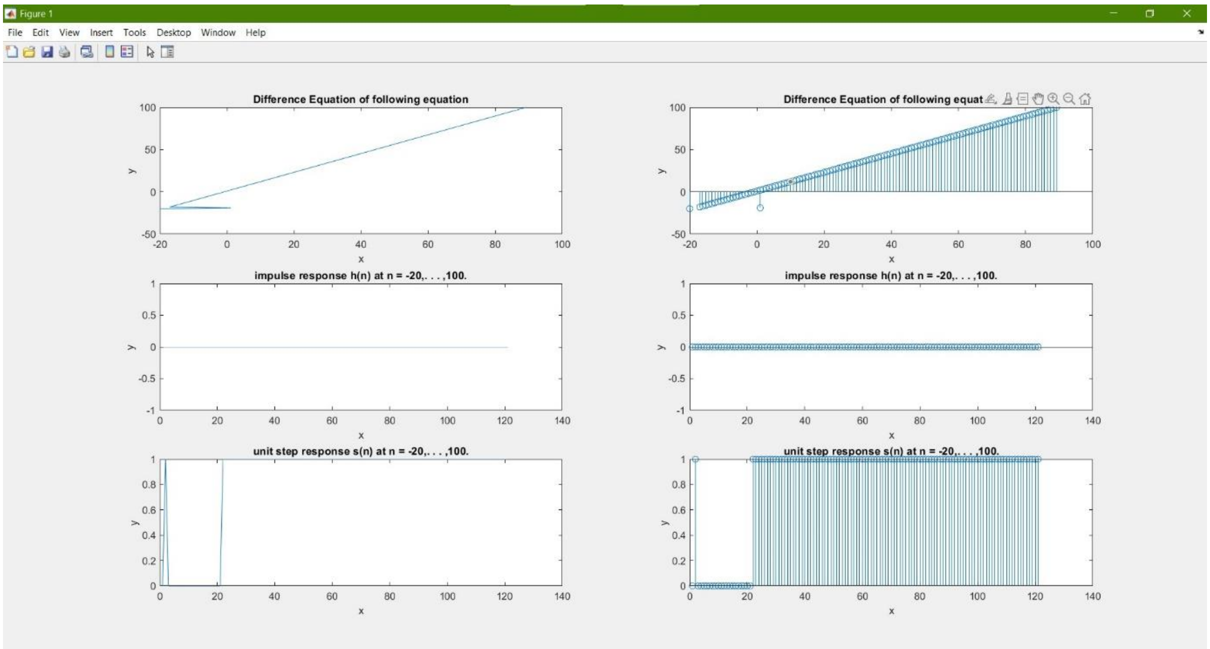


Figure 1
File Edit View Insert Tools Desktop Window Help
Difference Equation of following equation
Difference Equation of following equat
impulse response h(n) at n = -20,. . . ,100.
impulse response h(n) at n = -20,. . . ,100.
unit step response s(n) at n = -20,. . . ,100.
unit step response s(n) at n = -20,. . . ,100.
x
y

Q5) A particular linear and time-invariant system is described by the difference equation:

y(n) - 0.5y(n-1) + 0.25y(n-2) = x(n) + 2x(n-1) + x(n-3)

a) **Determine the stability of the system.**
b) **Determine and plot the impulse response of the system over 0<=n<=100.**
c) **Determine the stability from this impulse response**

CODE:

```
%A particular linear and time-invariant system is described by the
%difference equation:

% y(n) - 0.5y(n-1) + 0.25y(n-2) = x(n) + 2x(n-1) + x(n-3)

% a) Determine the stability of the system.
%Therefore: We need to use the filter function
%Let us take coefficients of both y (here a) and x (here b)

a = [1 -5/10 25/100];
b = [1 2 1];
y = 0:100;
x = filter(a,b,y);
subplot(211),plot(x,y);
title('Difference Equation of following equation');
xlabel('x');
ylabel('y');
B = isstable(x);
disp(B);
% b) Determine and plot the impulse response of the system over 0<=n<=100.
z = dirac(x);
subplot(212),stem(z,y);
title('Impulse Response of Difference Equation of following equation');
xlabel('z');
ylabel('y');
% c) Determine the stability from this impulse response
B2 = isstable(z);
disp(B2);
```

OUTPUT:

>> %A particular linear and time-invariant system is described by the

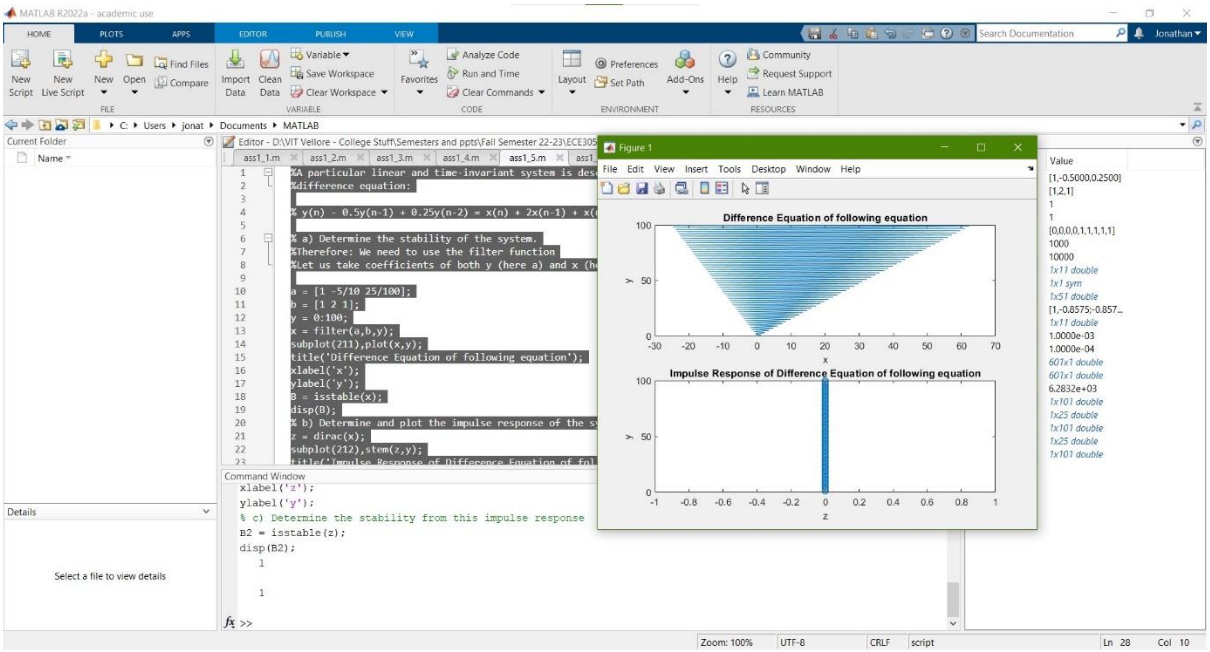
%difference equation:

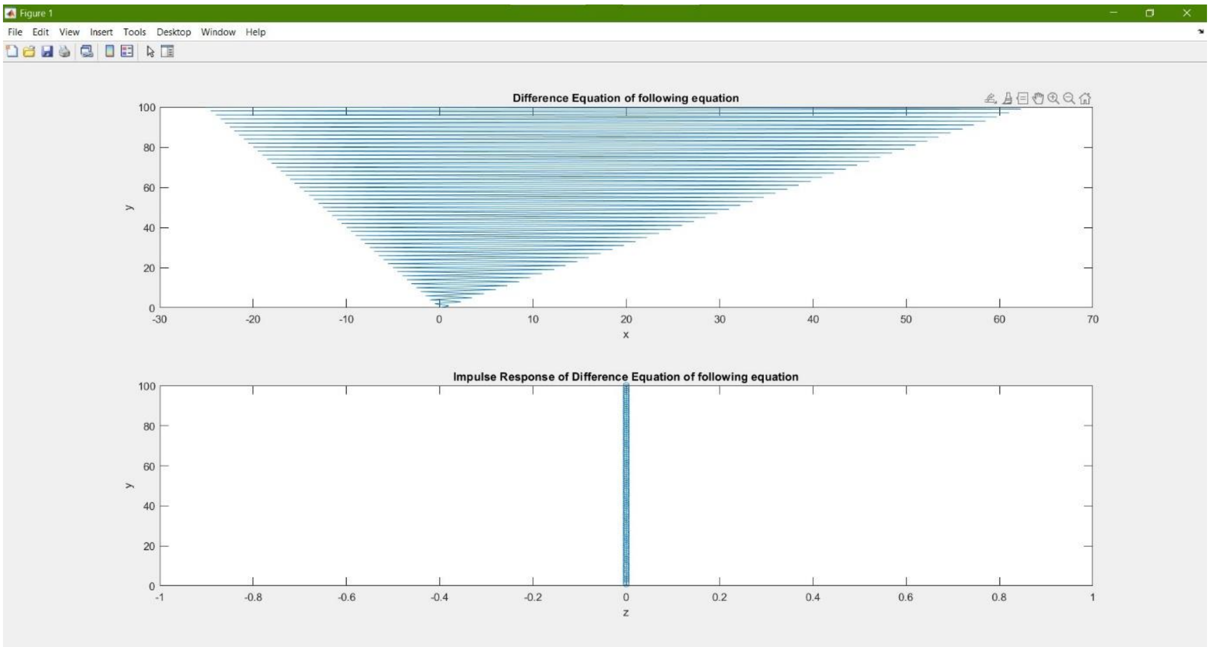
% y(n) - 0.5y(n-1) + 0.25y(n-2) = x(n) + 2x(n-1) + x(n-3)

% a) Determine the stability of the system.

%Therefore: We need to use the filter function

```
%Let us take coefficients of both y (here a) and x (here b)

a = [1 -5/10 25/100];
b = [1 2 1];
y = 0:100;
x = filter(a,b,y);
subplot(211),plot(x,y);
title('Difference Equation of following equation');
xlabel('x');
ylabel('y');
B = isstable(x);
disp(B);
% b) Determine and plot the impulse response of the system over 0<=n<=100.
z = dirac(x);
subplot(212),stem(z,y);
title('Impulse Response of Difference Equation of following equation');
xlabel('z');
ylabel('y');
% c) Determine the stability from this impulse response
B2 = isstable(z);
disp(B2);
  1

  1

>>
```




Difference Equation of following equation
Impulse Response of Difference Equation of following equation

Q6) A simple digital differentiator is given by:

y(n) = x(n) - x(n-1)

which computes a backward first-order difference in the input sequence. Implement this differentiator on the following sequences and plot the results. Comment on the appropriateness of this simple differentiator.

a) x(n) = 5[u(n)-u(n-20)]: a rectangular pose
b) x(n) = n[u(n) - u(n-10)] + (20-n)[u(n-10)-u(n-20)]: a triangular pose.
c) x(n) = sin(pi*n/25) * [u(n) - u(n-100)]: a sinusoidal pulse

CODE:

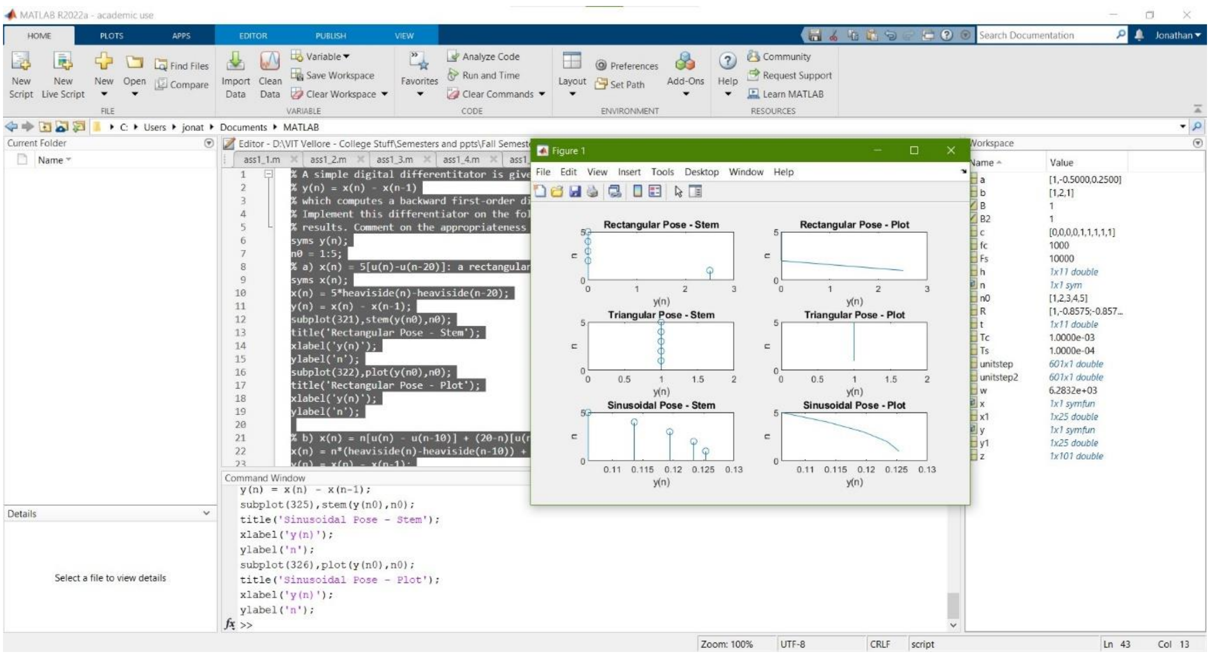
```
% A simple digital differentitator is given by:
% y(n) = x(n) - x(n-1)
% which computes a backward first-order difference in the input sequence.
% Implement this differentiator on the following sequences and plot the
% results. Comment on the appropriateness of this simple differentiator.
syms y(n);
n0 = 1:5;
% a) x(n) = 5[u(n)-u(n-20)]: a rectangular pose
syms x(n);
x(n) = 5*heaviside(n)-heaviside(n-20);
y(n) = x(n) - x(n-1);
subplot(321),stem(y(n0),n0);
title('Rectangular Pose - Stem');
xlabel('y(n)');
ylabel('n');
subplot(322),plot(y(n0),n0);
title('Rectangular Pose - Plot');
xlabel('y(n)');
ylabel('n');

% b) x(n) = n[u(n) - u(n-10)] + (20-n)[u(n-10)-u(n-20)]: a triangular pose.
x(n) = n*(heaviside(n)-heaviside(n-10)) + ((20-n)*(heaviside(n-10) - heaviside(n-
20)));
y(n) = x(n) - x(n-1);
subplot(323),stem(y(n0),n0);
title('Triangular Pose - Stem');
xlabel('y(n)');
ylabel('n');
subplot(324),plot(y(n0),n0);
title('Triangular Pose - Plot');
xlabel('y(n)');
ylabel('n');

% c) x(n) = sin(pi*n/25) * [u(n) - u(n-100)]: a sinusoidal pulse
x(n) = sin((pi*n)/25) * (heaviside(n) - heaviside(n-100));
y(n) = x(n) - x(n-1);
subplot(325),stem(y(n0),n0);
title('Sinusoidal Pose - Stem');
xlabel('y(n)');
ylabel('n');
subplot(326),plot(y(n0),n0);
title('Sinusoidal Pose - Plot');
xlabel('y(n)');
```

```
ylabel('n');
```

OUTPUT:

```
>> % A simple digital differentitator is given by:
% y(n) = x(n) - x(n-1)
% which computes a backward first-order difference in the input sequence.
% Implement this differentiator on the following sequences and plot the
% results. Comment on the appropriateness of this simple differentiator.
syms y(n);
n0 = 1:5;
% a) x(n) = 5[u(n)-u(n-20)]: a rectangular pose
syms x(n);
x(n) = 5*heaviside(n)-heaviside(n-20);
y(n) = x(n) - x(n-1);
subplot(321),stem(y(n0),n0);
title('Rectangular Pose - Stem');
xlabel('y(n)');
ylabel('n');
subplot(322),plot(y(n0),n0);
title('Rectangular Pose - Plot');
xlabel('y(n)');
ylabel('n');

% b) x(n) = n[u(n) - u(n-10)] + (20-n)[u(n-10)-u(n-20)]: a triangular pose.
x(n) = n*(heaviside(n)-heaviside(n-10)) + ((20-n)*(heaviside(n-10) - heaviside(n-20)));
y(n) = x(n) - x(n-1);
subplot(323),stem(y(n0),n0);
title('Triangular Pose - Stem');
xlabel('y(n)');
ylabel('n');
subplot(324),plot(y(n0),n0);
```

```
title('Triangular Pose - Plot');
xlabel('y(n)');
ylabel('n');

% c) x(n) = sin(pi*n/25) * [u(n) - u(n-100)]: a sinusoidal pulse
x(n) = sin((pi*n)/25) * (heaviside(n) - heaviside(n-100));
y(n) = x(n) - x(n-1);
subplot(325),stem(y(n0),n0);
title('Sinusoidal Pose - Stem');
xlabel('y(n)');
ylabel('n');
subplot(326),plot(y(n0),n0);
title('Sinusoidal Pose - Plot');
xlabel('y(n)');
ylabel('n');
>>
```



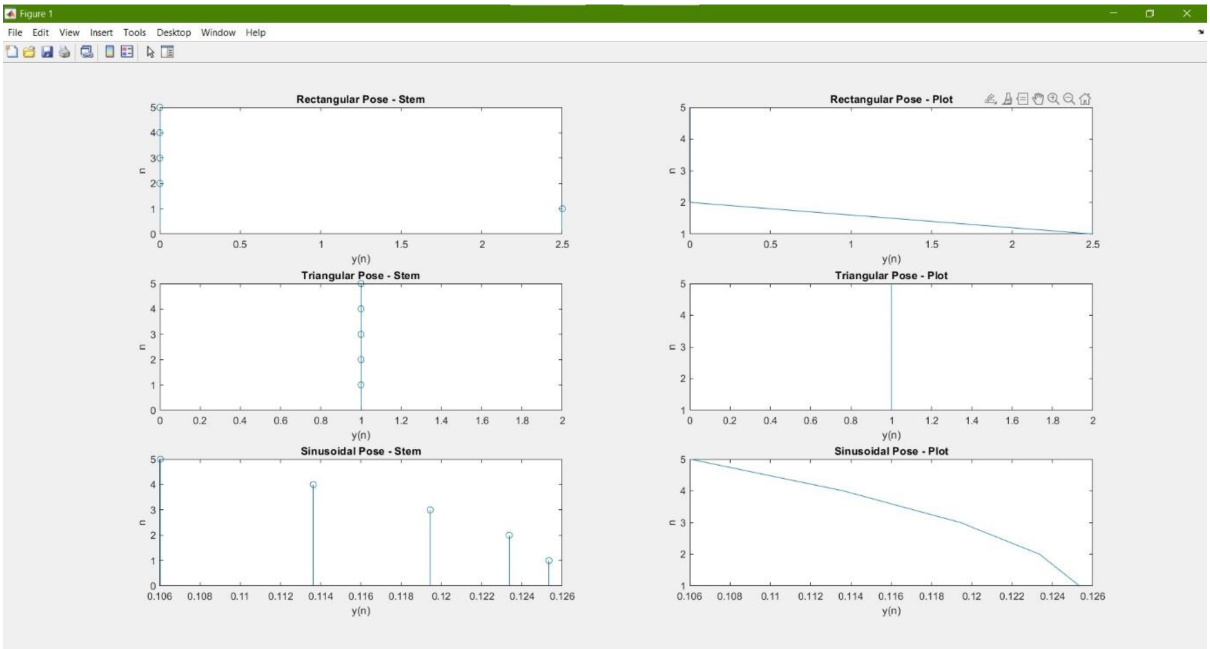


Figure 1
File Edit View Insert Tools Desktop Window Help
Rectangular Pose - Stem
Rectangular Pose - Plot
Triangular Pose - Stem
Triangular Pose - Plot
Sinusoidal Pose - Stem
Sinusoidal Pose - Plot
y(n)
n
