Electrical & Computer Engineering Program

ECEN 314 - Fall 2019

Signals and Systems

MATLAB Project

Student's Name:

Hayfaa Al-Kuwari 925007133
Hissa Al-Darwish 625004390

Date of Submission: December 2nd 2019

Instructor: Dr. Khalid Qaraqe
Teaching Assistant: Ms. Takwa Tarhini

"On my honor, as an Aggie, I have neither given nor received unauthorized aid on this academic work"

Content

Introduction ……... **2**

Literature Review …….. **3**

Task one …… **4-7**

Question 1: Continuous Time Signal Representation ……**4**

Question 2: Discrete Time Signal Representation ……**4**

Question 3: Use of Vectors to Storing Discrete Signals …… **4**

Question 4: Exponential Decay Implementation ……..**5**

Question 5: Modified Signals with Echo ……**5**

Question 6: Reverberation Effect …….. **5**

Question 7: Computation of Time Frequency …… **6-7**

Task Two ……... **8-12**

Section 1: Complex Number Basics …… **8**

Section 2: Complex Number Continuation …….. **9**

Section 3: Power and Message Signals ……..…..**10-11**

Section 4: Convolution Sum ……**12**

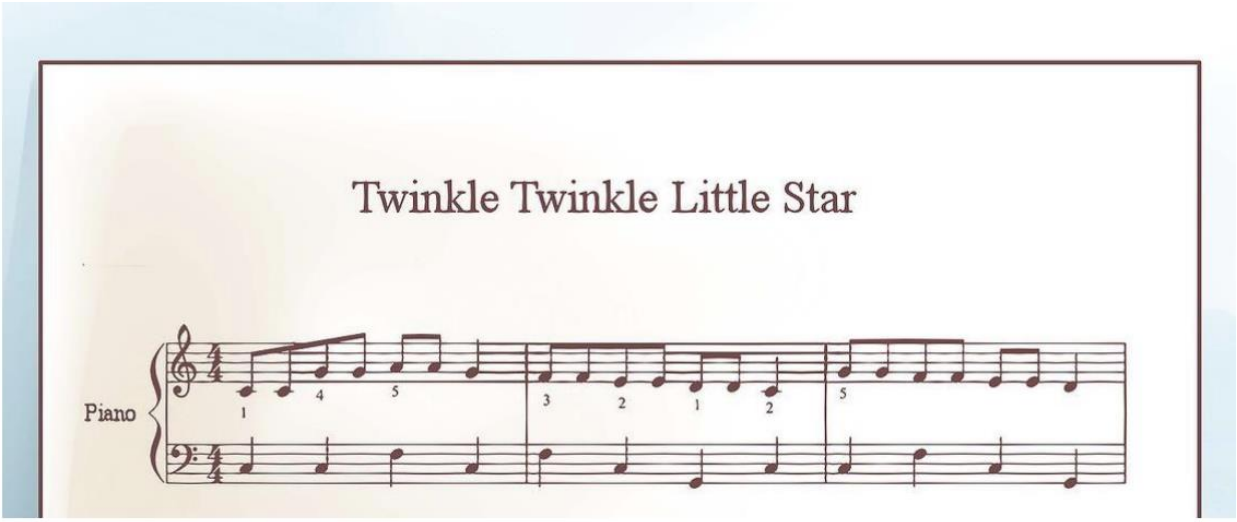
Discussion ……..**13**

Conclusion ……..…..**13**

References ……**14**

Appendix ……....**15-28**

Introduction



This paper focuses on how MATLAB software can be used in signals and systems. There are several MATLAB commands that helps users in understanding the behaviour of certain signals. The design of this project was divided into two different parts. The first task contains seven questions about the continuous time signals and discrete time signals as mathematical expressions which were one of the learning outcomes of our signals and systems in our electrical engineering course. The first task was about the conversion of an analogue audio signal that consists of musical notes into a digital representation by using certain frequencies and expressions. By understanding the instructions given to us we knew every musical note produces a sinusoidal wave, then by the code we created we had to make certain modifications in order to satisfy the given task. One of them was the need of adding an exponential part to the code to make the sound be more sensible and realistic. In the second task, each section was independent of the other, were we had to create certain functions, modify them and plot them. For the second task, each section is independent from one another. Which requires us to use certain functions to plot and test certain applications.

Literature review:

In order to understand how the musical notes, work and how musicians can read them, we watched a video that includes a brief explanation about each note and the location of it. Also, we learned what each symbol means and how can we label each line. By learning this we were able to understand the procedure and hence writing the mathematical expression for each note. The following table shows the given frequencies that we were required to use in order to create our musical piece (Table1).

Note	Frequency(Hz)
A	220
$A^{\#}, B^{b}$	$220*2^{1/12}$
B	$220*2^{2/12}$
C	$220*2^{3/12}$
$C^{\#}, D^{b}$	$220*2^{4/12}$
D	$220*2^{5/12}$
$D^{\#}, E^{b}$	$220*2^{6/12}$
E	$220*2^{7/12}$
F	$220*2^{8/12}$
$F^{\#}, G^{b}$	$220*2^{9/12}$
G	$220*2^{10/12}$
$G^{\#}, A^{b}$	$220*2^{11/12}$

Table 1: Notes in the 220-440 Hz octave.

Task 1

Question 1

Each musical note was presented as a mathematical expression with respect to time. Earlier we identified that the musical notes are sinusoidal waves so the expression is defined to be a sine function multiplied by heaviside function (unit step), and the resultant output should be a rectangular function which makes the splitting of notes easier and smoother. We decided to divide the full note into halves where the beat takes 0.25 seconds and the pause between each note is identified as the given instructions to be 0.25 seconds. In total the musical piece given to us contains 12 notes and 21 pauses in total.

Question 2

For this task, we were required to convert the continuous time signal to discrete time signal by sampling the continuous time signal at a frequency of 8KHz. The discrete time signal is defined as x[N]= x[N*T] where T is the fundamental time period which is equal to the inverse of the 8KHz frequency (1/8000 seconds).

Question 3

For this part, a row vector was constructed which represents the discrete time of the notes. First, we created an empty array or vector and named it y. Then, for loop was created to convert the notes into row vector form. The audiowrite and the audioread commands were used in this task. The empty array is then converted and saved as an audio file using the audiowrite command. However, the audioread command is used to read the selected range of audio samples in the file where the samples are in the form of vector. Finally, the sound command was used to play the musical notes.

Question 4

In this task, an exponential decay function was created to make the volume of the notes decay over time for more realistic output. The exponential decay function was multiplied with the discrete time signal. The exponential decay was represented as exp(-T.*N) with a time period of 1/8000 seconds.

Question 5

In this part, a vector was generated which includes the original signal with the addition of an echo. This echo should occur at T seconds after the original signal starts and must have an amplitude as large the original signal at times. The function that is used to generate the vector was the echogenerator with an amplitude of A and a time delay of Td. The formula used in this part was $s_e = \alpha(t-T)s(t-T)$ which is provided in the project manual. $\alpha(t-T)$ is the attenuation function while $s(t-T)$ is the original signal shifted T units.

Question 6

In this task, an echo was generated to create a reverb effect. The code from question 5 was saved as a function file called echogenerator and used in this task. The attenuation factor and the time delay were given as 0.65 and 0.5 seconds respectively. The sound command was used to display the sound of the music with the effect of reverb.

Question 7

In this task, we were required to compute the time-frequency representation of the continuous time signal from Question 1. This was performed using the spectrogram MATLAB function. The spectrogram produces the musical notes in the frequency domain. The plot identifies the accuracy of the frequency values. The plot shows the relation between the normalized frequency and samples. The x-axis represents the samples while the y-axis represents the normalized frequency in radians per sample. The normalized equation is given as$Frequency\ (Hz) = \frac{1}{2} \times Normalized\ frequency \times sampling\ frequency$which is used to find out the frequency of different colors. The following table shows the frequencies of different colors compared with the musical notes. (Table 2)

Note	**Normalized Frequency**	**Measured Frequency (Hz)**	**Theoretical Frequency (Hz)**
C	0.0625	250	261.6
G	0.09375	375	391.9
A	0.1016	406.4	440
G	0.09375	375	391.9
F	0.07813	312.5	349.2
E	0.07813	312.5	329.6
D	0.07031	281.24	293.66
C	0.07031	281.24	261.6
G	0.0937	374.8	391.9
F	0.08594	343.76	349.2
E	0.07813	312.52	329.6
D	0.07031	281.24	293.66

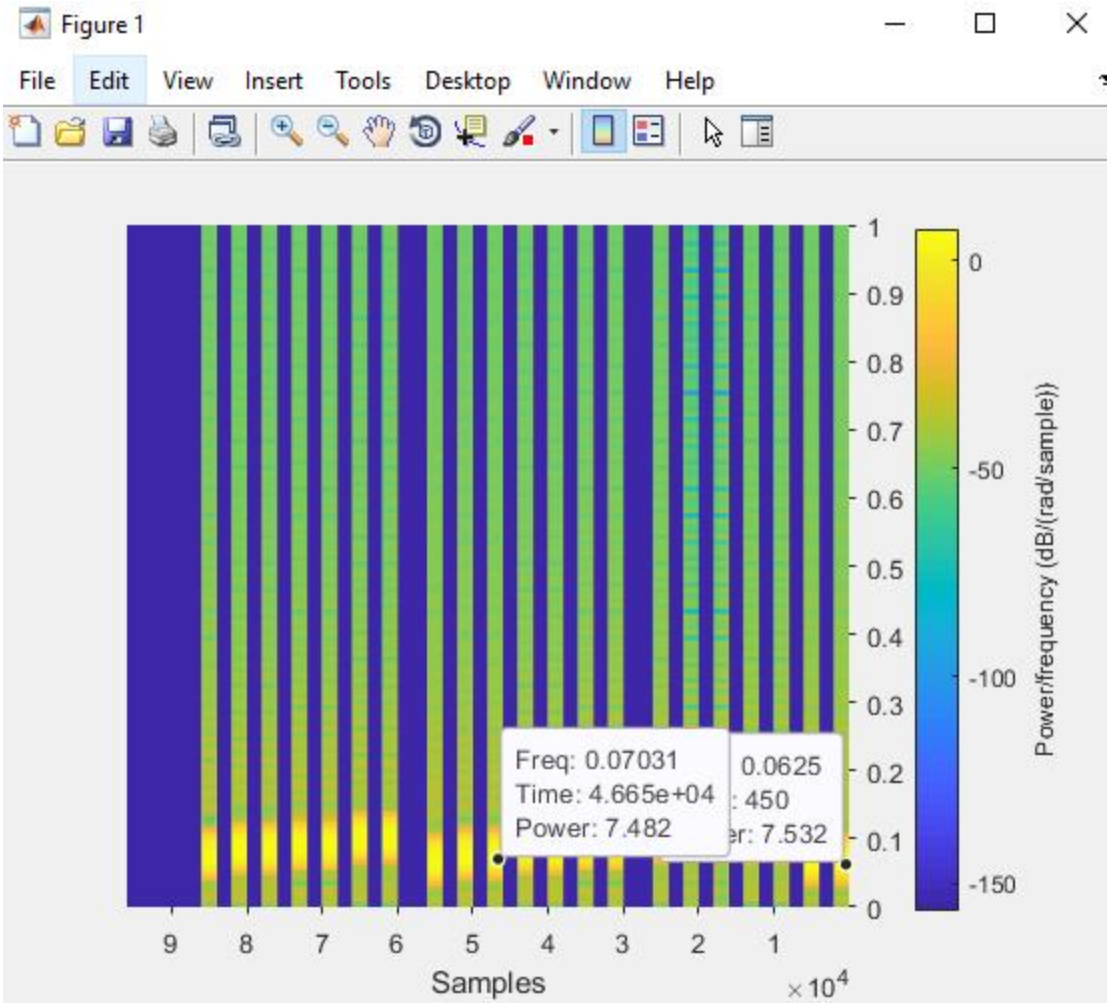
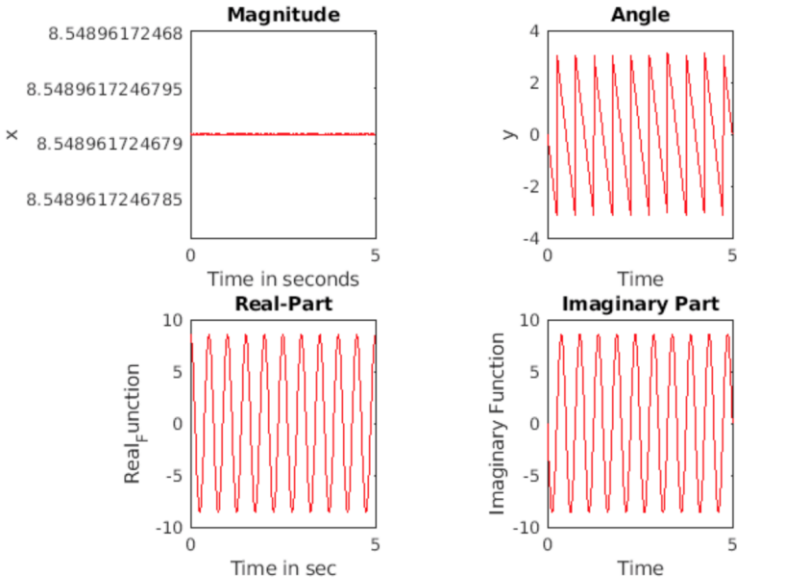


Figure 1
File
Edit
View
Insert
Tools
Desktop
Window
Help
Freq: 0.07031
Time: 4.665e+04
Power: 7.482
0.0625
: 450
er: 7.532
Samples
×10^4
Power/frequency (dB/(rad/sample))

Task 2

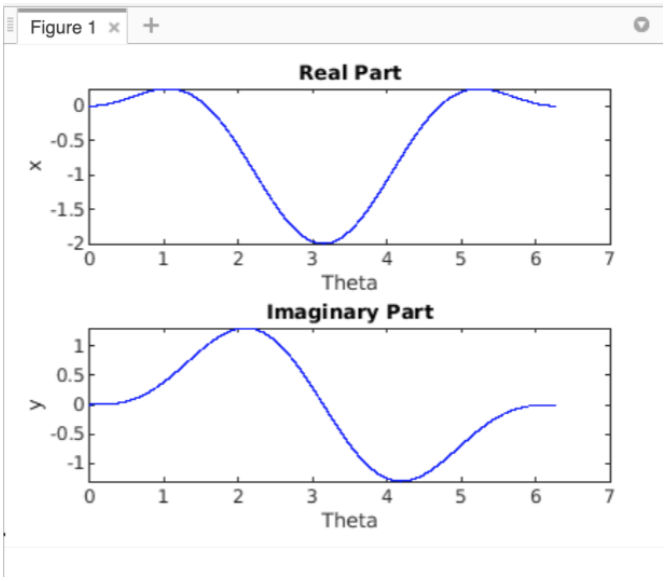
Section 1

In this question, we were required to plot the real and imaginary parts of a complex function, and be familiar with the complex numbers magnitude, real and imaginary parts and the phase angle. In order to compute them we used Angle(ft) to find the phase angle of the given expression. In order to get the magnitude abs was used to get the absolute value. However, in order to distinguish between the real and imaginary parts imag and real were used.



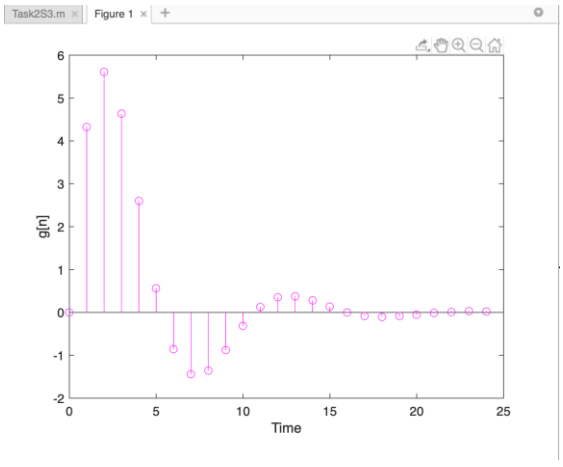
Section 2

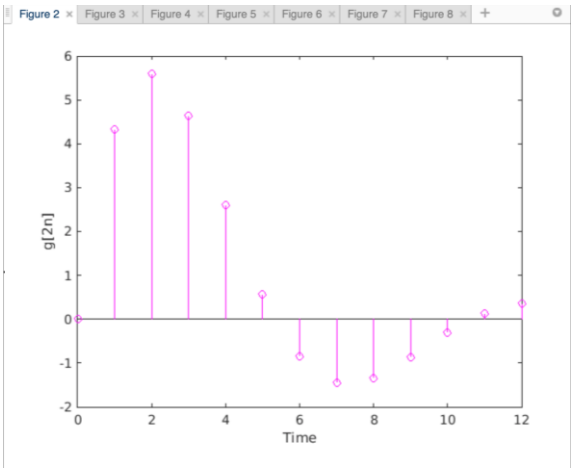
In this question, method used in question 1 was used again to get the imaginary and real parts. In addition to that by using plot and subplots commands in MATLAB. The graphs were plotted.

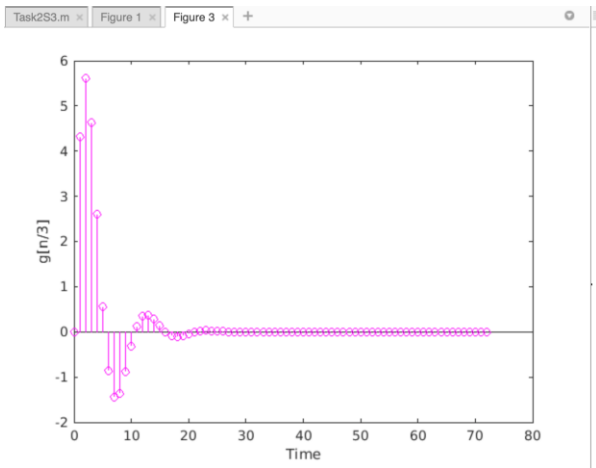


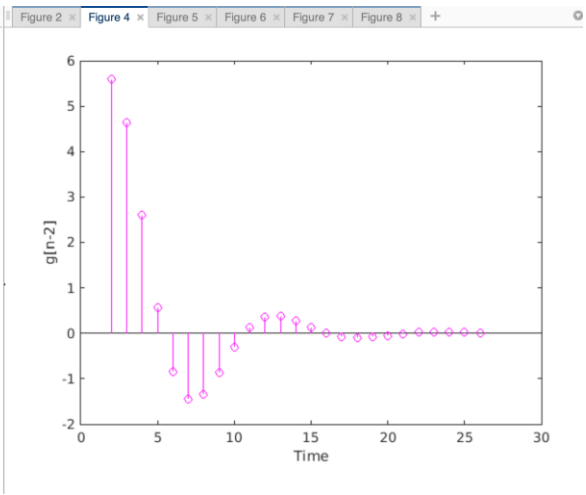
Section 3

In this section, stem function was used, the purpose of this function is used to display data as line starting from the x-axis. A circle by default is inserted in order to represent the value that terminates at each ends of stem. A function was written in MATLAB using the regular script. Then multiple graphs were plotted using stem function to plot discrete time graphs.









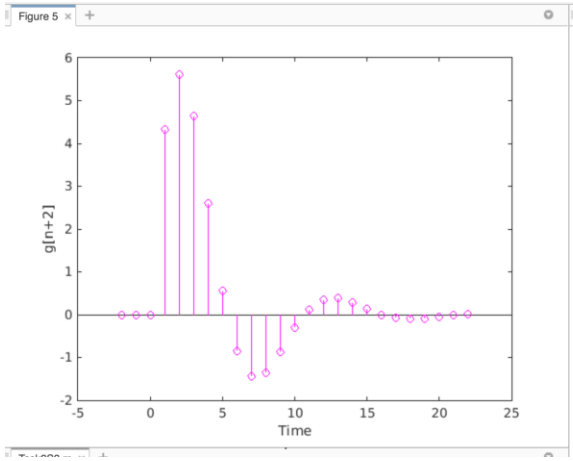


Figure 5
g[n+2]
Time

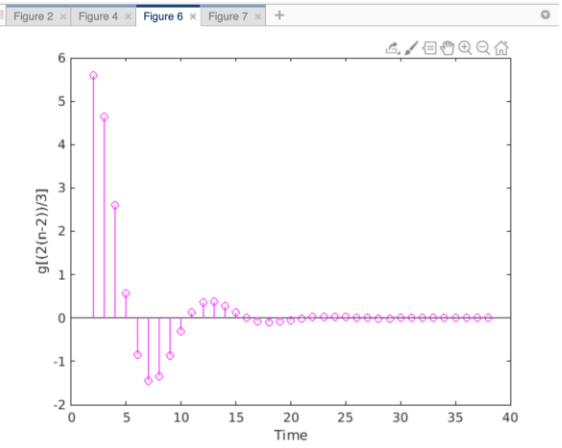
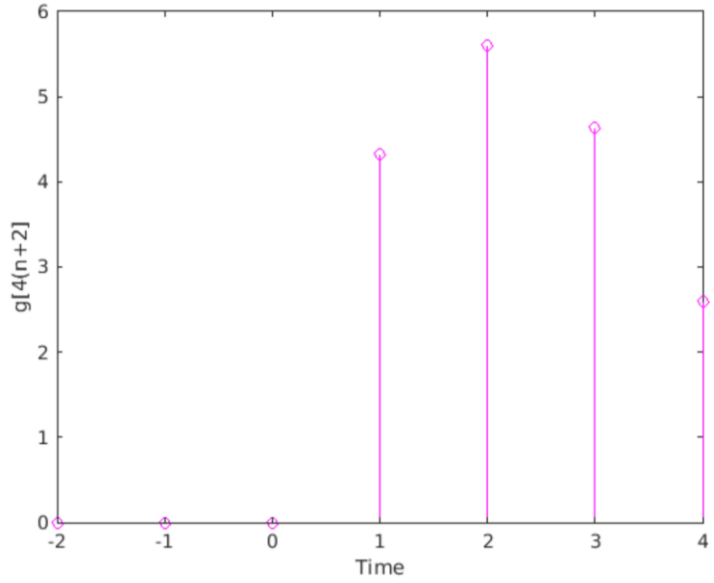
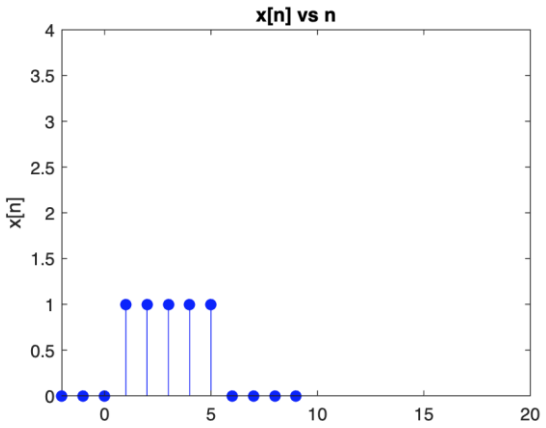


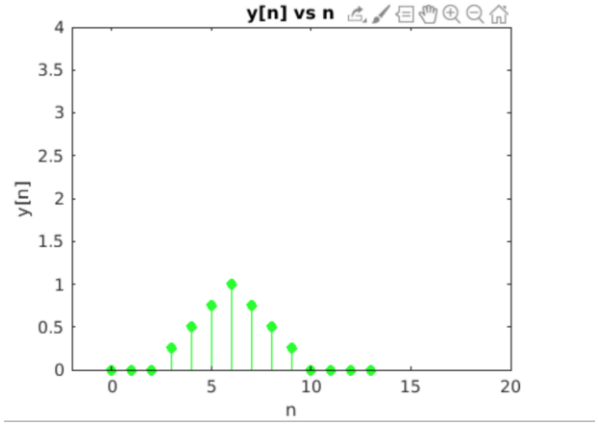
Figure 2
Figure 4
Figure 6
Figure 7
g[(2(n-2))/3]
Time

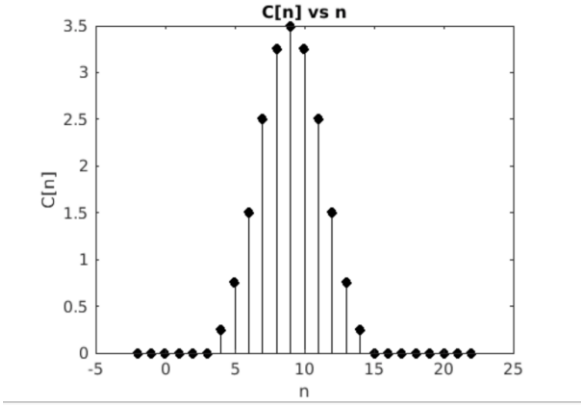

g[4(n+2]
Time

Section 4

The last part of this task is to compute the convolution sum using rectangular pulse and triangular pulse. Using MATLAB functions, triangular pulse was generated by using trangularPulse function. And the rectangular pulse was generated by using rectangularPulse. Finally, the convolution between the rectangular and triangular pulses was generated by using the conv(x,y) function, then the results were plotted using Stem to generate a discrete time vertical lines graphs.







Discussion and Conclusion

This task familiarized us with MATLAB commands that turns out to be useful when solving certain questions in signals and systems, we observed the behavior of signals in each section. The first two sections were dealing with complex numbers.

Finally, this project covered different aspects that we learned in our signals and systems course, we learned theoretically about signals and how they behave. But with this project it gave us an insight about how signals can actually be represented as sine or cosine functions and how the duration of each signal can be controlled using the unit step function. By using the knowledge, we learned from ECEN 314, we were able to express a musical note into a mathematical expression. This mathematical expression was turned into a code in MATLAB to generate an audio, and then implementing certain modifications to make it sound smoother and interesting. Also, the second task we learned how to plot functions and used conv, stem commands. By using these commands, we were able to plot functions in discrete time domain.

References:

Pianote. (2019, march 19).How to read Notes (Beginner piano lesson). Retrieved from https://www.youtube.com/watch?v=gEI7uYOCQXo&t=496s

Appendix

Question 1:

```
clc; clear; close all;
% For task 1 question 1, we have to write sin functions multiplied with a
% rect function that will represents the frequency and the duration of each
% note. The notes will be used are Middle C,G,A,E,F and D. The order will
% be pause,Cm,G,G,pause,A,A,pause,G.
% G, E, pause, F, F, D


Cm=220*2^(3/12);
D=220*2^(5/12);
E=220*2^(7/12);
F=220*2^(8/12);
G=220*2^(10/12);
A=220*2;
t=0:1/8000:12; % defining the time scale for the notes.

note1=sin(2*pi*Cm.*t).*(heaviside(t)-heaviside(t-0.25)); %the middle c note
with duration of 0.25.
note1a=sin(2*pi*Cm.*t).*(heaviside(t-0.5)-heaviside(t-0.75));%middle c note
with a pause of 0.25 and a duration of 0.25.

note2=sin(2*pi*G.*t).*(heaviside(t-1)-heaviside(t-1.25));
note2a=sin(2*pi*G.*t).*(heaviside(t-1.5)-heaviside(t-1.75));

note3=sin(2*pi*A.*t).*(heaviside(t-2)-heaviside(t-2.25));
note3a=sin(2*pi*A.*t).*(heaviside(t-2.5)-heaviside(t-2.75));

note4=sin(2*pi*G.*t).*(heaviside(t-3)-heaviside(t-3.25));

note5=sin(2*pi*F.*t).*(heaviside(t-3.75)-heaviside(t-4));
note5a=sin(2*pi*F.*t).*(heaviside(t-4.25)-heaviside(t-4.5));

note6=sin(2*pi*E.*t).*(heaviside(t-4.75)-heaviside(t-5));
note6a=sin(2*pi*E.*t).*(heaviside(t-5.25)-heaviside(t-5.5));

note7=sin(2*pi*D.*t).*(heaviside(t-5.75)-heaviside(t-6));
note7a=sin(2*pi*D.*t).*(heaviside(t-6.25)-heaviside(t-6.5));

note8=sin(2*pi*Cm.*t).*(heaviside(t-6.75)-heaviside(t-7));

note9=sin(2*pi*G.*t).*(heaviside(t-7.5)-heaviside(t-7.75));
note9a=sin(2*pi*G.*t).*(heaviside(t-8)-heaviside(t-8.25));


note10=sin(2*pi*F.*t).*(heaviside(t-8.5)-heaviside(t-8.75));
```

```
note10a=sin(2*pi*F.*t).*(heaviside(t-9)-heaviside(t-9.25));


note11=sin(2*pi*E.*t).*(heaviside(t-9.5)-heaviside(t-9.75));
note11a=sin(2*pi*E.*t).*(heaviside(t-10)-heaviside(t-10.25));


note12=sin(2*pi*D.*t).*(heaviside(t-10.5)-heaviside(t-10.75));

NOTE=note1+note1a+note2+note2a+note3+note3a+note4+note5+note5a+note6+note6a+n
ote7+note7a+note8+note9+note9a+note10+note10a+note11+note11a+note12;%the sum
of all notes to be viewed as a music

sound(NOTE); % sound function to play the musical note.
```

Question 2:

```
%%QUESTION 2


clc; clear; close all;

Cm=220*2^(3/12);
D=220*2^(5/12);
E=220*2^(7/12);
F=220*2^(8/12);
G=220*2^(10/12);
A=220*2;
N=0:1:11*8000; % defining the time scale for the notes.
T=1/8000;


 %first question code was converted into a discrete time signal.

note1=sin(2*pi*Cm.*(T*N)).*(heaviside((T*N))-heaviside((T*N)-0.25));
note1a=sin(2*pi*Cm.*(T*N)).*(heaviside((T*N)-0.5)-heaviside((T*N)-0.75));

note2=sin(2*pi*G.*(T*N)).*(heaviside((T*N)-1)-heaviside((T*N)-1.25));
note2a=sin(2*pi*G.*(T*N)).*(heaviside((T*N)-1.5)-heaviside((T*N)-1.75));


note3=sin(2*pi*A.*(T*N)).*(heaviside((T*N)-2)-heaviside((T*N)-2.25));
note3a=sin(2*pi*A.*(T*N)).*(heaviside((T*N)-2.5)-heaviside((T*N)-2.75));


note4=sin(2*pi*G.*(T*N)).*(heaviside((T*N)-3)-heaviside((T*N)-3.25));


note5=sin(2*pi*F.*(T*N)).*(heaviside((T*N)-3.75)-heaviside((T*N)-4));
```

```
note5a=sin(2*pi*F.*(T*N)).*(heaviside((T*N)-4.25)-heaviside((T*N)-4.5));


note6=sin(2*pi*E.*(T*N)).*(heaviside((T*N)-4.75)-heaviside((T*N)-5));
note6a=sin(2*pi*E.*(T*N)).*(heaviside((T*N)-5.25)-heaviside((T*N)-5.5));

note7=sin(2*pi*D.*(T*N)).*(heaviside((T*N)-5.75)-heaviside((T*N)-6));
note7a=sin(2*pi*D.*(T*N)).*(heaviside((T*N)-6.25)-heaviside((T*N)-6.5));


note8=sin(2*pi*Cm.*(T*N)).*(heaviside((T*N)-6.75)-heaviside((T*N)-7));


note9=sin(2*pi*G.*(T*N)).*(heaviside((T*N)-7.5)-heaviside((T*N)-7.75));
note9a=sin(2*pi*G.*(T*N)).*(heaviside((T*N)-8)-heaviside((T*N)-8.25));


note10=sin(2*pi*F.*(T*N)).*(heaviside((T*N)-8.5)-heaviside((T*N)-8.75));
note10a=sin(2*pi*F.*(T*N)).*(heaviside((T*N)-9)-heaviside((T*N)-9.25));


note11=sin(2*pi*E.*(T*N)).*(heaviside((T*N)-9.5)-heaviside((T*N)-9.75));
note11a=sin(2*pi*E.*(T*N)).*(heaviside((T*N)-10)-heaviside((T*N)-10.25));


note12=sin(2*pi*D.*(T*N)).*(heaviside((T*N)-10.5)-heaviside((T*N)-10.75));

NOTE=note1+note1a+note2+note2a+note3+note3a+note4+note5+note5a+note6+note6a+n
ote7+note7a+note8+note9+note9a+note10+note10a+note11+note11a+note12;%the sum
of all notes to be viewed as a music

sound(NOTE);
```

Question 3

```
%%QUESTION 3
clc; clear; close all;
```

```
Cm=220*2^(3/12);
D=220*2^(5/12);
E=220*2^(7/12);
F=220*2^(8/12);
G=220*2^(10/12);
A=220*2;
N=0:1:11*8000;
T=1/8000;
y=zeros(1,88001); %Empty row vector created to save musical notes.

note1=sin(2*pi*Cm.*(T*N)).*(heaviside((T*N))-heaviside((T*N)-0.25));
note1a=sin(2*pi*Cm.*(T*N)).*(heaviside((T*N)-0.5)-heaviside((T*N)-0.75));

note2=sin(2*pi*G.*(T*N)).*(heaviside((T*N)-1)-heaviside((T*N)-1.25));
note2a=sin(2*pi*G.*(T*N)).*(heaviside((T*N)-1.5)-heaviside((T*N)-1.75));

note3=sin(2*pi*A.*(T*N)).*(heaviside((T*N)-2)-heaviside((T*N)-2.25));
note3a=sin(2*pi*A.*(T*N)).*(heaviside((T*N)-2.5)-heaviside((T*N)-2.75));

note4=sin(2*pi*G.*(T*N)).*(heaviside((T*N)-3)-heaviside((T*N)-3.25));

note5=sin(2*pi*F.*(T*N)).*(heaviside((T*N)-3.75)-heaviside((T*N)-4));
note5a=sin(2*pi*F.*(T*N)).*(heaviside((T*N)-4.25)-heaviside((T*N)-4.5));

note6=sin(2*pi*E.*(T*N)).*(heaviside((T*N)-4.75)-heaviside((T*N)-5));
note6a=sin(2*pi*E.*(T*N)).*(heaviside((T*N)-5.25)-heaviside((T*N)-5.5));

note7=sin(2*pi*D.*(T*N)).*(heaviside((T*N)-5.75)-heaviside((T*N)-6));
note7a=sin(2*pi*D.*(T*N)).*(heaviside((T*N)-6.25)-heaviside((T*N)-6.5));

note8=sin(2*pi*Cm.*(T*N)).*(heaviside((T*N)-6.75)-heaviside((T*N)-7));

note9=sin(2*pi*G.*(T*N)).*(heaviside((T*N)-7.5)-heaviside((T*N)-7.75));
note9a=sin(2*pi*G.*(T*N)).*(heaviside((T*N)-8)-heaviside((T*N)-8.25));

note10=sin(2*pi*F.*(T*N)).*(heaviside((T*N)-8.5)-heaviside((T*N)-8.75));
note10a=sin(2*pi*F.*(T*N)).*(heaviside((T*N)-9)-heaviside((T*N)-9.25));

note11=sin(2*pi*E.*(T*N)).*(heaviside((T*N)-9.5)-heaviside((T*N)-9.75));
note11a=sin(2*pi*E.*(T*N)).*(heaviside((T*N)-10)-heaviside((T*N)-10.25));

note12=sin(2*pi*D.*(T*N)).*(heaviside((T*N)-10.5)-heaviside((T*N)-10.75));

y_n=note1+note1a+note2+note2a+note3+note3a+note4+note5+note5a+note6+note6a+no
te7+note7a+note8+note9+note9a+note10+note10a+note11+note11a+note12;

for N=1:1:88001 %A loop was created to save the discrete signals within the
vector.
```

```
row_vector(N)=y_n(N);
end

audiowrite('Twinkle Twinkle.wav',y_n,8000); %creates a file for the musical
piece.
R=audioread('Twinkle Twinkle.wav'); %reads the musical piece stored in the
file created by the audiowrite.
sound(R); %Sound function to play the music.
```

Question 4

```
%%QUESTION 4

clc; clear; close all;
%frequencies

Cm=220*2^(3/12);
D=220*2^(5/12);
E=220*2^(7/12);
F=220*2^(8/12);
G=220*2^(10/12);
A=220*2;
N=0:1:11*8000;
T=1/8000;
y=zeros(1,88001);%Empty row vector created to save musical notes.
%Exponential function was added to create a decay which makes the  music
sounds better.
note1=sin(2*pi*Cm.*(T*N)).*(heaviside((T*N))-heaviside((T*N)-0.25)).*exp(-
T.*N);
note1a=sin(2*pi*Cm.*(T*N)).*(heaviside((T*N)-0.5)-heaviside((T*N)-
0.75)).*exp(-T.*N+0.5);

note2=sin(2*pi*G.*(T*N)).*(heaviside((T*N)-1)-heaviside((T*N)-1.25)).*exp(-
T.*N+1);
note2a=sin(2*pi*G.*(T*N)).*(heaviside((T*N)-1.5)-heaviside((T*N)-
1.75)).*exp(-T.*N+1.5);

note3=sin(2*pi*A.*(T*N)).*(heaviside((T*N)-2)-heaviside((T*N)-2.25)).*exp(-
T.*N+2);
note3a=sin(2*pi*A.*(T*N)).*(heaviside((T*N)-2.5)-heaviside((T*N)-
2.75)).*exp(-T.*N+2.5);

note4=sin(2*pi*G.*(T*N)).*(heaviside((T*N)-3)-heaviside((T*N)-3.25)).*exp(-
T.*N+3);

note5=sin(2*pi*F.*(T*N)).*(heaviside((T*N)-3.75)-heaviside((T*N)-4)).*exp(-
T.*N+3.75);
note5a=sin(2*pi*F.*(T*N)).*(heaviside((T*N)-4.25)-heaviside((T*N)-
4.5)).*exp(-T.*N+4.25);
```

```
note6=sin(2*pi*E.*(T*N)).*(heaviside((T*N)-4.75)-heaviside((T*N)-5)).*exp(-
T.*N+4.75);
note6a=sin(2*pi*E.*(T*N)).*(heaviside((T*N)-5.25)-heaviside((T*N)-
5.5)).*exp(-T.*N+5.25);

note7=sin(2*pi*D.*(T*N)).*(heaviside((T*N)-5.75)-heaviside((T*N)-6)).*exp(-
T.*N+5.75);
note7a=sin(2*pi*D.*(T*N)).*(heaviside((T*N)-6.25)-heaviside((T*N)-
6.5)).*exp(-T.*N+6.25);

note8=sin(2*pi*Cm.*(T*N)).*(heaviside((T*N)-6.75)-heaviside((T*N)-7)).*exp(-
T.*N+6.75);

note9=sin(2*pi*G.*(T*N)).*(heaviside((T*N)-7.5)-heaviside((T*N)-7.75)).*exp(-
T.*N+7.5);
note9a=sin(2*pi*G.*(T*N)).*(heaviside((T*N)-8)-heaviside((T*N)-8.25)).*exp(-
T.*N+8);

note10=sin(2*pi*F.*(T*N)).*(heaviside((T*N)-8.5)-heaviside((T*N)-
8.75)).*exp(-T.*N+8.5);
note10a=sin(2*pi*F.*(T*N)).*(heaviside((T*N)-9)-heaviside((T*N)-9.25)).*exp(-
T.*9);

note11=sin(2*pi*E.*(T*N)).*(heaviside((T*N)-9.5)-heaviside((T*N)-
9.75)).*exp(-T.*N+9.5);
note11a=sin(2*pi*E.*(T*N)).*(heaviside((T*N)-10)-heaviside((T*N)-
10.25)).*exp(-T.*N+10);

note12=sin(2*pi*D.*(T*N)).*(heaviside((T*N)-10.5)-heaviside((T*N)-
10.75)).*exp(-T.*N+10.5);

y_n=note1+note1a+note2+note2a+note3+note3a+note4+note5+note5a+note6+note6a+no
te7+note7a+note8+note9+note9a+note10+note10a+note11+note11a+note12;
for N=1:1:88001
row_vector(N)=y_n(N);
end
audiowrite('Twinkle Twinkle.wav',y_n,8000);
R=audioread('Twinkle Twinkle.wav');
sound(R);
```

Question 5

```
%%QUESTION 5

clc; clear; close all;
% Echo generator was used to generate echo sound.
%Where A is the Amplitude and Td is the time delay.
function [echo_music]= echogenerator(A,Td)
```

```
Cm=220*2^(3/12);
D=220*2^(5/12);
E=220*2^(7/12);
F=220*2^(8/12);
G=220*2^(10/12);
A=220*2;
N=0:1:11*8000;
T=1/8000;
y=zeros(1,88001);

note1=sin(2*pi*Cm.*(T*N)).*(heaviside((T*N))-heaviside((T*N)-0.25)).*exp(-
T.*N);
note1a=sin(2*pi*Cm.*(T*N)).*(heaviside((T*N)-0.5)-heaviside((T*N)-
0.75)).*exp(-T.*N+0.5);

note2=sin(2*pi*G.*(T*N)).*(heaviside((T*N)-1)-heaviside((T*N)-1.25)).*exp(-
T.*N+1);
note2a=sin(2*pi*G.*(T*N)).*(heaviside((T*N)-1.5)-heaviside((T*N)-
1.75)).*exp(-T.*N+1.5);

note3=sin(2*pi*A.*(T*N)).*(heaviside((T*N)-2)-heaviside((T*N)-2.25)).*exp(-
T.*N+2);
note3a=sin(2*pi*A.*(T*N)).*(heaviside((T*N)-2.5)-heaviside((T*N)-
2.75)).*exp(-T.*N+2.5);

note4=sin(2*pi*G.*(T*N)).*(heaviside((T*N)-3)-heaviside((T*N)-3.25)).*exp(-
T.*N+3);

note5=sin(2*pi*F.*(T*N)).*(heaviside((T*N)-3.75)-heaviside((T*N)-4)).*exp(-
T.*N+3.75);
note5a=sin(2*pi*F.*(T*N)).*(heaviside((T*N)-4.25)-heaviside((T*N)-
4.5)).*exp(-T.*N+4.25);

note6=sin(2*pi*E.*(T*N)).*(heaviside((T*N)-4.75)-heaviside((T*N)-5)).*exp(-
T.*N+4.75);
note6a=sin(2*pi*E.*(T*N)).*(heaviside((T*N)-5.25)-heaviside((T*N)-
5.5)).*exp(-T.*N+5.25);

note7=sin(2*pi*D.*(T*N)).*(heaviside((T*N)-5.75)-heaviside((T*N)-6)).*exp(-
T.*N+5.75);
note7a=sin(2*pi*D.*(T*N)).*(heaviside((T*N)-6.25)-heaviside((T*N)-
6.5)).*exp(-T.*N+6.25);

note8=sin(2*pi*Cm.*(T*N)).*(heaviside((T*N)-6.75)-heaviside((T*N)-7)).*exp(-
T.*N+6.75);

note9=sin(2*pi*G.*(T*N)).*(heaviside((T*N)-7.5)-heaviside((T*N)-7.75)).*exp(-
T.*N+7.5);
```

```
note9a=sin(2*pi*G.*(T*N)).*(heaviside((T*N)-8)-heaviside((T*N)-8.25)).*exp(-
T.*N+8);

note10=sin(2*pi*F.*(T*N)).*(heaviside((T*N)-8.5)-heaviside((T*N)-
8.75)).*exp(-T.*N+8.5);
note10a=sin(2*pi*F.*(T*N)).*(heaviside((T*N)-9)-heaviside((T*N)-9.25)).*exp(-
T.*9);

note11=sin(2*pi*E.*(T*N)).*(heaviside((T*N)-9.5)-heaviside((T*N)-
9.75)).*exp(-T.*N+9.5);
note11a=sin(2*pi*E.*(T*N)).*(heaviside((T*N)-10)-heaviside((T*N)-
10.25)).*exp(-T.*N+10);

note12=sin(2*pi*D.*(T*N)).*(heaviside((T*N)-10.5)-heaviside((T*N)-
10.75)).*exp(-T.*N+10.5);

y_n=note1+note1a+note2+note2a+note3+note3a+note4+note5+note5a+note6+note6a+no
te7+note7a+note8+note9+note9a+note10+note10a+note11+note11a+note12;

for N=1:1:88001 % creats a loop to save the discrete signals in the empty
vector.
row_vector(N)=y_n(N);
end
audiowrite('Musicwithecho.wav', Music,8000);
[M,F]= audioread('MUSICWITHECHO.wav', 'Native');

Time_delay=Td.*F;

Length= size(M);
echo_music= zeros(Length);

for i=1+Time_delay:1:Length %For loop was used again to add the echoed signal
to the orignal one. And then multiplying by the amplitude and delayed by the
Td.
    echo_music(i)=row_vector(i)+A*row_vector(i-Time_delay);
end

end
```

Question 6:

```
clc; clear; close all;
% In this question we are supposed to call the function created in question 5
Ax=0.65;
Td=0.5;
b=echogenerator(Ax,Td);% the function is assigned to variable b
sound(b)%plays the signal generated from the echogenerator.
```

Question 7:

```
clc; clear; close all;

%Use the exact from code question 1
Cm=220*2^(3/12);
D=220*2^(5/12);
E=220*2^(7/12);
F=220*2^(8/12);
G=220*2^(10/12);
A=220*2;
t=0:1/8000:12; % defining the time scale for the notes.

note1=sin(2*pi*Cm.*t).*(heaviside(t)-heaviside(t-0.25));
note1a=sin(2*pi*Cm.*t).*(heaviside(t-0.5)-heaviside(t-0.75));

note2=sin(2*pi*G.*t).*(heaviside(t-1)-heaviside(t-1.25));
note2a=sin(2*pi*G.*t).*(heaviside(t-1.5)-heaviside(t-1.75));

note3=sin(2*pi*A.*t).*(heaviside(t-2)-heaviside(t-2.25));
note3a=sin(2*pi*A.*t).*(heaviside(t-2.5)-heaviside(t-2.75));

note4=sin(2*pi*G.*t).*(heaviside(t-3)-heaviside(t-3.25));

note5=sin(2*pi*F.*t).*(heaviside(t-3.75)-heaviside(t-4));
note5a=sin(2*pi*F.*t).*(heaviside(t-4.25)-heaviside(t-4.5));

note6=sin(2*pi*E.*t).*(heaviside(t-4.75)-heaviside(t-5));
note6a=sin(2*pi*E.*t).*(heaviside(t-5.25)-heaviside(t-5.5));

note7=sin(2*pi*D.*t).*(heaviside(t-5.75)-heaviside(t-6));
note7a=sin(2*pi*D.*t).*(heaviside(t-6.25)-heaviside(t-6.5));

note8=sin(2*pi*Cm.*t).*(heaviside(t-6.75)-heaviside(t-7));

note9=sin(2*pi*G.*t).*(heaviside(t-7.5)-heaviside(t-7.75));
note9a=sin(2*pi*G.*t).*(heaviside(t-8)-heaviside(t-8.25));


note10=sin(2*pi*F.*t).*(heaviside(t-8.5)-heaviside(t-8.75));
note10a=sin(2*pi*F.*t).*(heaviside(t-9)-heaviside(t-9.25));


note11=sin(2*pi*E.*t).*(heaviside(t-9.5)-heaviside(t-9.75));
note11a=sin(2*pi*E.*t).*(heaviside(t-10)-heaviside(t-10.25));


note12=sin(2*pi*D.*t).*(heaviside(t-10.5)-heaviside(t-10.75));
```

```
NOTE=note1+note1a+note2+note2a+note3+note3a+note4+note5+note5a+note6+note6a+n
ote7+note7a+note8+note9+note9a+note10+note10a+note11+note11a+note12;%the sum
of all notes to be viewed as a music

sound(NOTE);
% using spectrogram function to view the time-frequency representation of
% the music notes, dividing the window into 100 segments
spectrogram(NOTE,100)

view(-100,100)
```

Section 1

```
% time interval
time = 0:0.01:5;

% complex function f(t)
ft = 3*exp((-i*4*pi.*time) + (pi/3));

% magnitude of complex function
magnitude = abs(ft);

% angle of complex
Angle = angle(ft);

% plotting complex magnitude
A = subplot(2,2,1);
plot(A,time,magnitude,'-r')
title(A,'Magnitude');
ylabel(A,'x');
xlabel(A,'Time in seconds');

% plotting phase angle
Phase = subplot(2,2,2);
plot(Phase,time,Angle,'-r')
title(Phase,'Angle');
ylabel(Phase,'y');
xlabel(Phase,'Time');
% real part of f(t)
RealPart = real(ft);
% imaginary part of f(t)
imag_part = imag(ft);

% plotting imaginary part
Phase = subplot(2,2,4);
plot(Phase,time,imag_part,'-r')
title(Phase,'Imaginary Part');
ylabel(Phase,'Imaginary Function');
xlabel(Phase,'Time');
```

```
% plotting real part
Phase = subplot(2,2,3);
plot(Phase,time,RealPart,'-r')
title(Phase,'Real-Part');
ylabel(Phase,'Real_Function');
xlabel(Phase,'Time in sec');
```

Section 2

```
% ANGLE
ANGLE = 0:0.001:2*pi;

% Radius
r = 1-cos(ANGLE);

% complex function z
z = r.*exp(i.*ANGLE );

% real part of z
Zreal = real(z);

% plot of  Real Part
A = subplot(2,1,1);
plot(A,ANGLE,Zreal,'-b')
title(A,'Real Part');
ylabel(A,'x');
xlabel(A,'Theta');

% imaginary part of z
ImaginryPart = imag(z);

% plot of imaginary part
b = subplot(2,1,2);
plot(b,ANGLE,ImaginryPart,'-b')
title(b,'Imaginary Part');
ylabel(b,'y');
xlabel(b,'Theta');
```

Section 3

```
help stem % help stem, this does a discrete stem plot
n1=0:24;% setting the interval of n
% given function g[n]
```

```
gn=10.*exp(-n1./4).*sin(3*pi.*n1./16).*heaviside(n1);
%stem function to plot discrete time ( its like a vertical lines going
from
%the x-axis)
stem(n1,gn,'-m');
xlabel('Time'); % x and y axis labels.
ylabel('g[n]');
figure

n2=0:12;
%given function g[2n]
gn2=10.*exp(-n2./4).*sin(3*pi.*n2./16).*heaviside(n2);
%stem function
stem(n2,gn2,'-m');
ylabel('g[2n]');
xlabel('Time');
figure

n3=0:72;
%function g[n/3]
gn3=10.*exp(-n3./4).*sin(3*pi.*n3./16).*heaviside(n3);
%stem function to plot discrete time
stem(n3,gn3,'-m');
ylabel('g[n/3]');
xlabel('Time');
figure

n4=2:26;
%function g[n-2]
gn4=10.*exp(-n4./4).*sin(3*pi.*n4./16).*heaviside(n4);
%stem function to plot discrete time
stem(n4,gn4,'-m');
ylabel('g[n-2]');
xlabel('Time');
figure

n5=-2:22;
%function g[n+2]
gn5=10.*exp(-n5./4).*sin(3*pi.*n5./16).*heaviside(n5);
%stem function to plot discrete time
stem(n5,gn5,'-m');
ylabel('g[n+2]');
xlabel('Time');
Figure

n6=2:38;
%function g[(2(n-2))/3]
```

```
gn6=10.*exp(-n6./4).*sin(3*pi.*n6./16).*heaviside(n6);
%stem function to plot discrete time
stem(n6,gn6,'-m');
ylabel('g[(2(n-2))/3]');
xlabel('Time');
Figure

n7=-2:4;
%function g[4(n+2]
gn7=10.*exp(-n7./4).*sin(3*pi.*n7./16).*heaviside(n7);
%stem function to plot discrete time
stem(n7,gn7,'-m');
ylabel('g[4(n+2]');xlabel('Time');
figure
```

Section 4

```
help conv %help conv, convolves vectors X and Y
xn=-2:9; % Setting time vectors for x[n] and y[n]
yn=0:13;
%Computing Values of x[n]
rect=rectangularPulse(-2.5,2.5,xn-3);
%Computing Values of y[n]
tri=triangularPulse((yn-6)/4);
%convolution between the given functions
convolution=conv(rect,tri);
%discrete-time vector
DTC=(xn(1)+yn(1))+(0:(length(xn)+length(yn)-2));
% plotting rectangular function
stem(xn,rect,'b','filled'); %  discrete graph
xlabel('n');
ylabel('x[n]');
axis([-2,20,0,4]);
title('x[n] vs n');
Figure

% plotting triangular function
stem(yn,tri,'g','filled');% discrete graph
title('y[n] vs n');
xlabel('n');
ylabel('y[n]');
axis([-2,20,0,4]);
figure

% plotting the convolution of the two functions
stem(DTC,convolution,'k','filled');
```

```
title('C[n] vs n');
xlabel('n');
ylabel('C[n]');
```