Bangladesh University of Engineering & Technology
Department of Electrical & Electronic Engineering

Course No: 312

Course Title: Digital Signal Processing Lab

Date of Performance: 09.03.2021

Date of Submission: 16.03.2021

Istiak Hasan Arnab

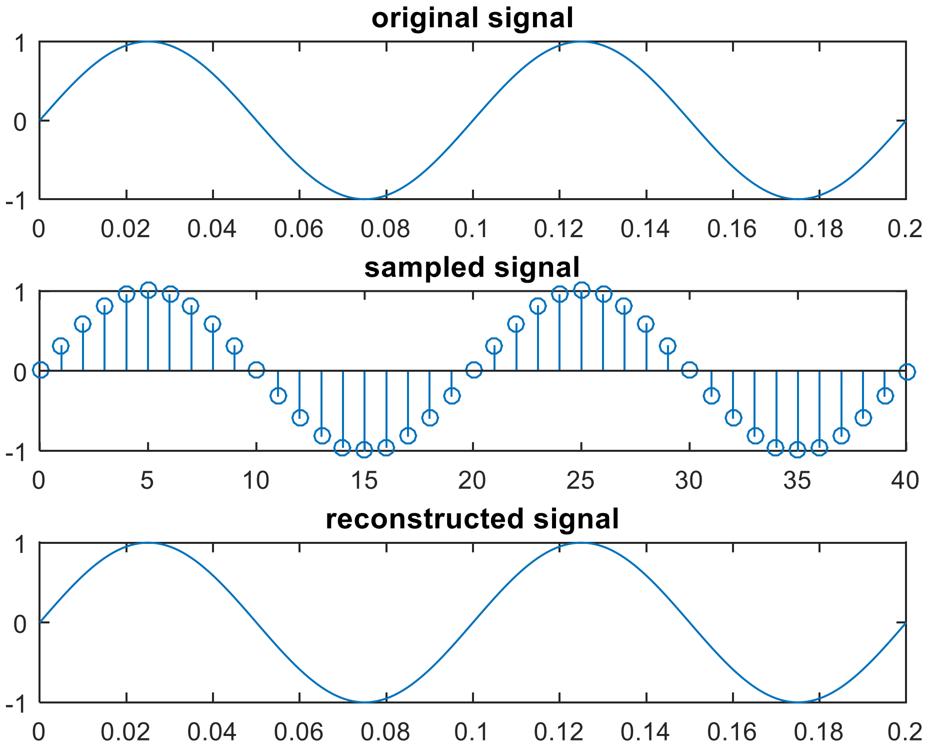
Student ID: 1706089

Section : B1

Level : 03   Term : 01

Part A:

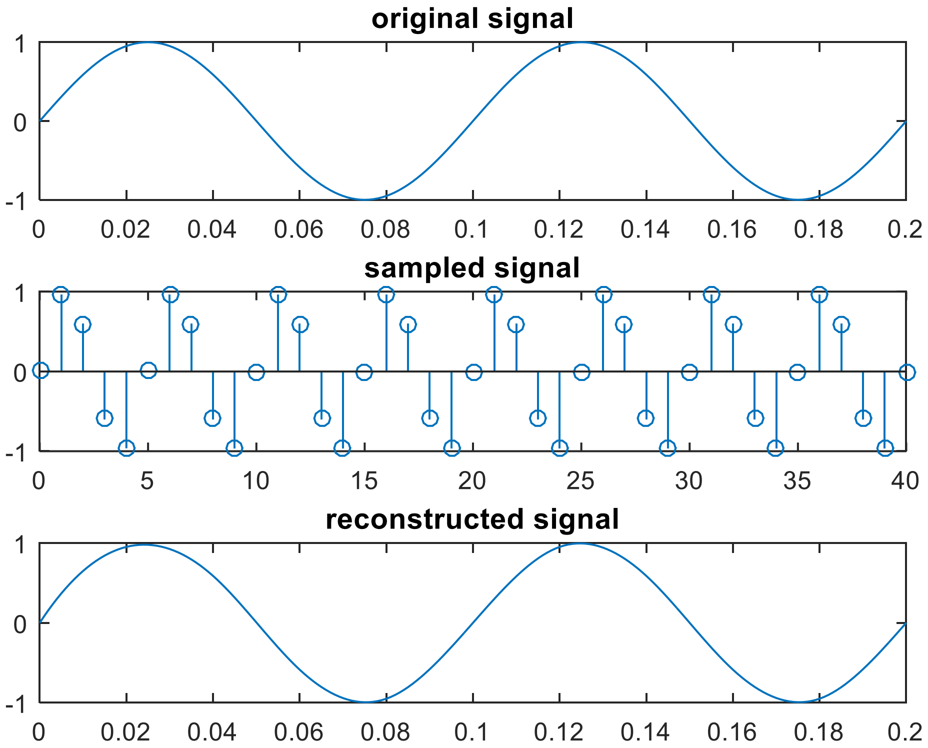
```
t=0:0.001:0.2;
n=0:1:40;
fs=200;
%original Signal
y=sin(2*pi*10*t+0);
subplot(3,1,1);
plot(t,y);title('original signal');
%sampling
y_samp=sin(2*pi*(10/fs)*n+0);
subplot(3,1,2);
stem(n,y_samp);title('sampled signal');
%reconstruction
y_rec=interp1(n/fs,y_samp,t,'spline');
subplot(3,1,3);
plot(t,y_rec);title('reconstructed signal');
```



For 50Hz:

```
fs=50;
%original Signal
y=sin(2*pi*10*t+0);
subplot(3,1,1);
plot(t,y);title('original signal');
%sampling
y_samp=sin(2*pi*(10/fs)*n+0);
subplot(3,1,2);
stem(n,y_samp);title('sampled signal');
%reconstruction
y_rec=interp1(n/fs,y_samp,t,'spline');
subplot(3,1,3);
plot(t,y_rec);title('reconstructed signal');
```

Output:


original signal

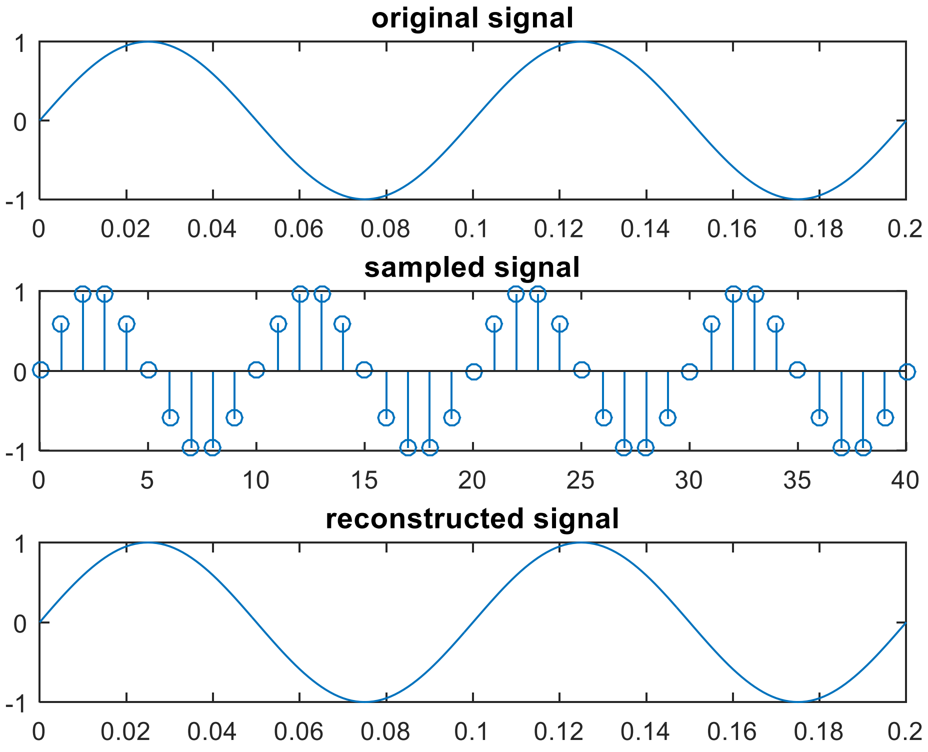
sampled signal

reconstructed signal

For 100Hz:

```
fs=100;
%original Signal
y=sin(2*pi*10*t+0);
subplot(3,1,1);
plot(t,y);title('original signal');
%sampling
y_samp=sin(2*pi*(10/fs)*n+0);
subplot(3,1,2);
stem(n,y_samp);title('sampled signal');
%reconstruction
y_rec=interp1(n/fs,y_samp,t,'spline');
subplot(3,1,3);
plot(t,y_rec);title('reconstructed signal');
```

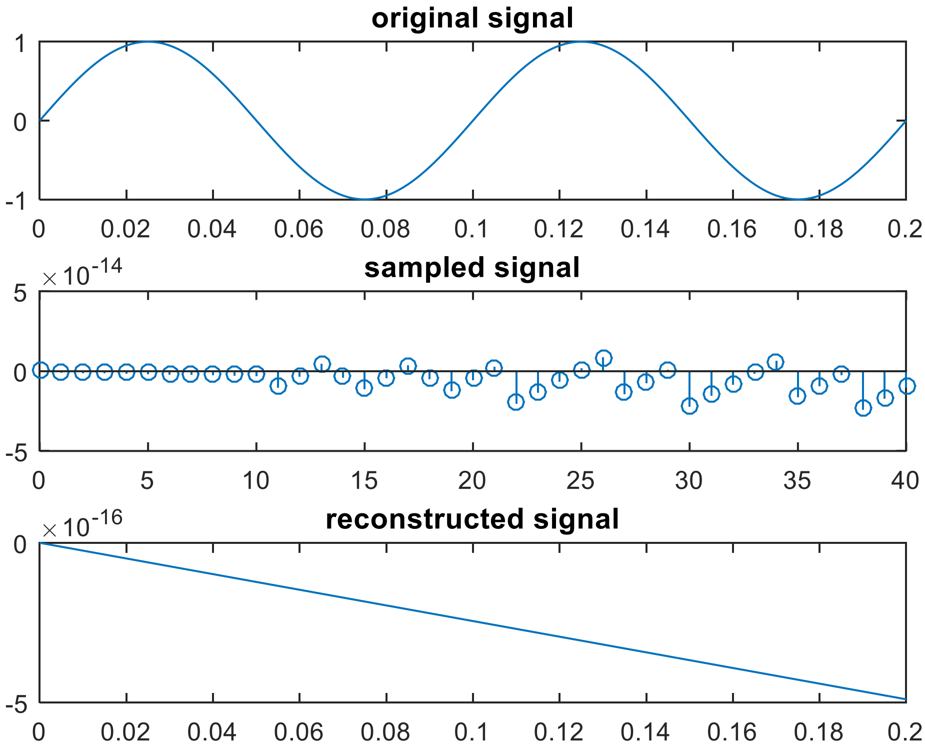
Output:



For 10Hz:

```
fs=10;
%original Signal
y=sin(2*pi*10*t+0);
subplot(3,1,1);
plot(t,y);title('original signal');
%sampling
y_samp=sin(2*pi*(10/fs)*n+0);
subplot(3,1,2);
stem(n,y_samp);title('sampled signal');
%reconstruction
y_rec=interp1(n/fs,y_samp,t,'spline');
subplot(3,1,3);
plot(t,y_rec);title('reconstructed signal');
```

Output:

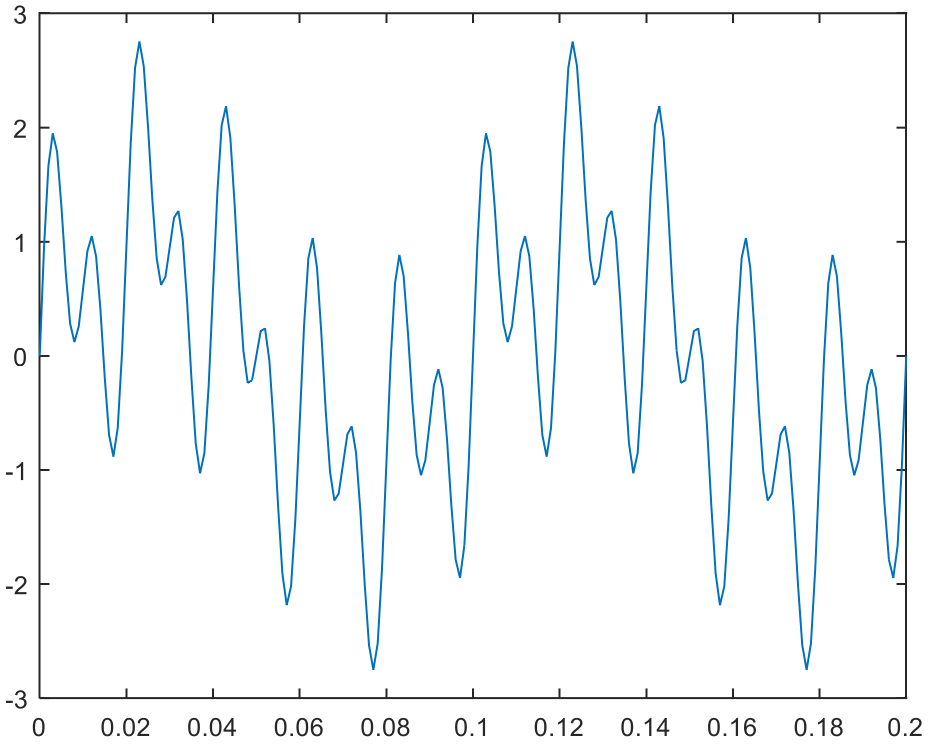


Comments: In step 4, if we take higher sampling frequency, the output signal will be more accurate. For successful reconstruction of the signal, the sampling must be greater than the twice of the original signal frequency.

Step 5:

```
x=sin(2*pi*10*t)+sin(2*pi*50*t)+sin(2*pi*100*t);
plot(t,x);
```

Output:



Part B:

Uniform 3-bit Quantizer:

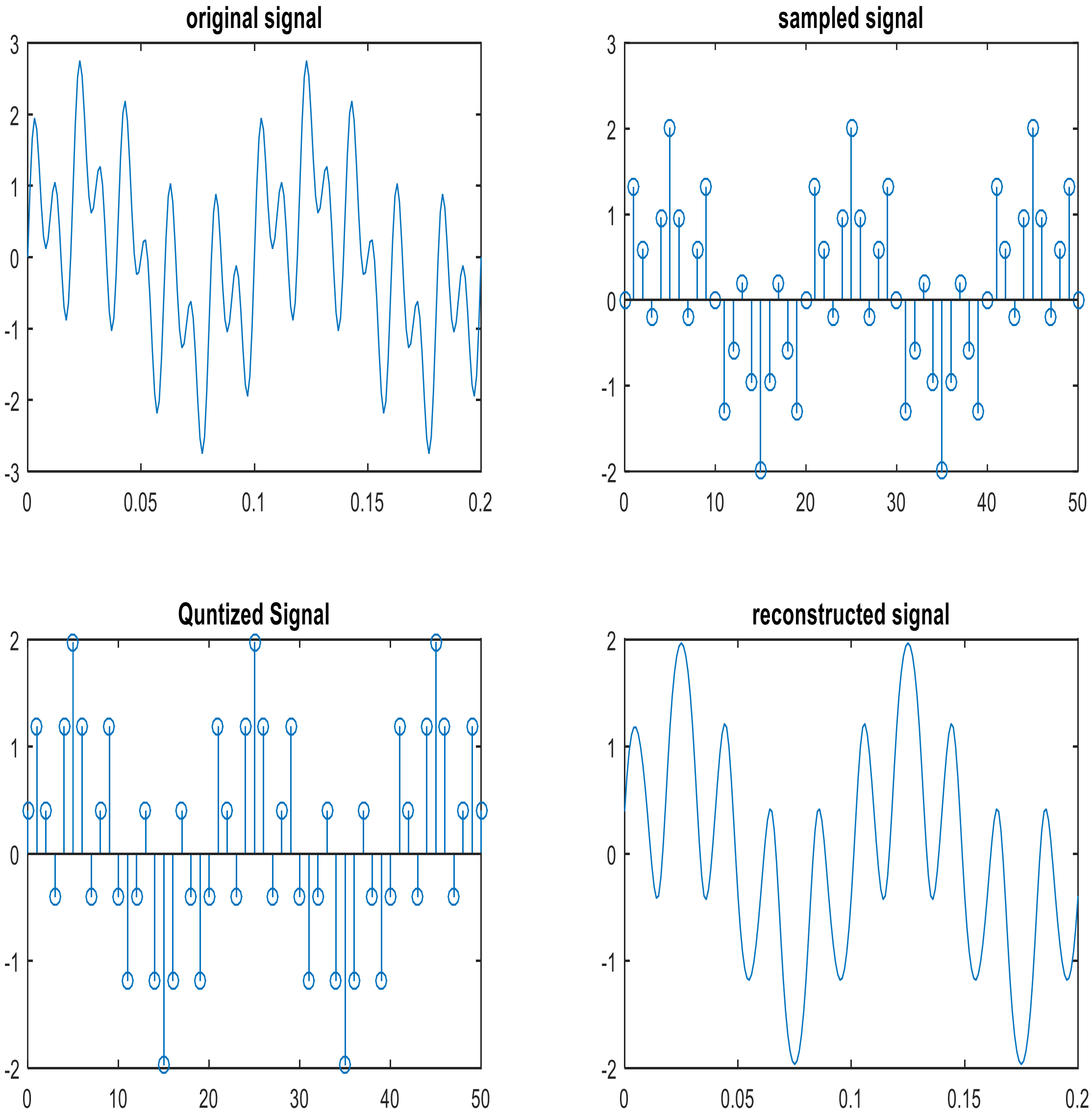
```
t=0:0.001:0.2;
fs=200;
y=sin(2*pi*10*t)+sin(2*pi*50*t)+sin(2*pi*100*t);
figure;plot(t,y);title('original signal');
%sampling
n=0:1:50;
y_samp=sin(2*pi*(10/fs)*n)+sin(2*pi*(50/fs)*n)+sin(2*pi
*(100/fs)*n);
figure;stem(n,y_samp);title('sampled signal');
MAX=max(y);
MIN=min(y);
%quantizer
b=3;
L=2^b;
del =( MAX - MIN ) /( L -1) ;
for i=1:L
    l(i)=MIN+del*(i-1);
end
yq=y_samp;
for i=1:length(y_samp)
    for j=1:L-1
        if(y_samp(i)>l(j) && y_samp(i)<l(j+1))
            u=y_samp(i)-l(j);
            v=l(j+1)-y_samp(i);
            if(u>v)
                yq(i)=l(j+1);
            else
                yq(i)=l(j);
            end
        end
    end
end
%reconstruction
y_rec=interp1(n/fs,yq,t,'spline');
figure;plot(t,y_rec);title('reconstructed signal');
%SQNR Determination
error=yq-y_samp;
qn=mean((yq-y_samp).^2);
```

```
sp=mean(y.^2);
sqnr=sp./qn;
sqnrdb_practical=10*log10(sqnr)
sqnrdb_formula=1.76+6.023*b
```

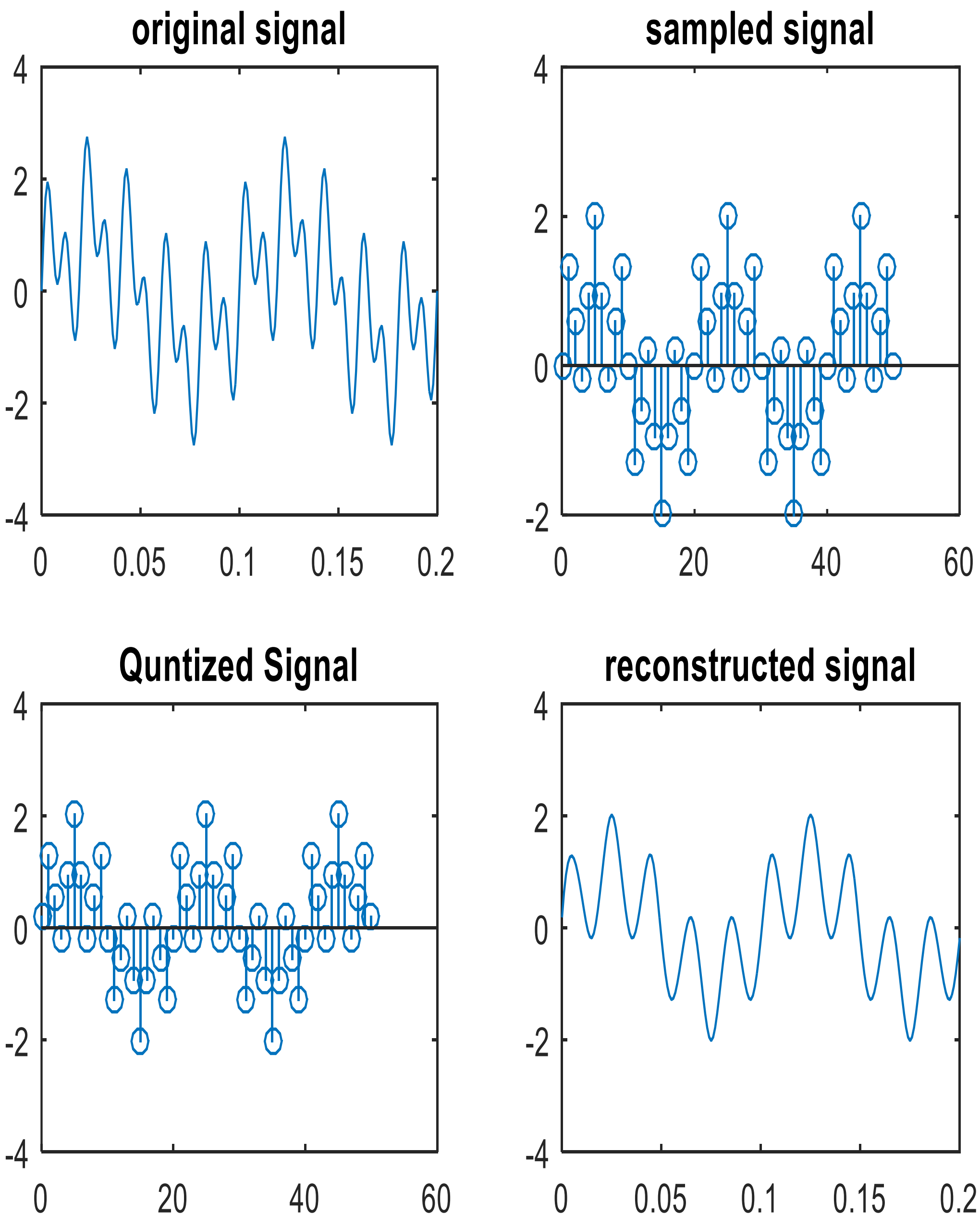
SQNR in theory = 19.8290

SQNR in Experiment = 14.9939

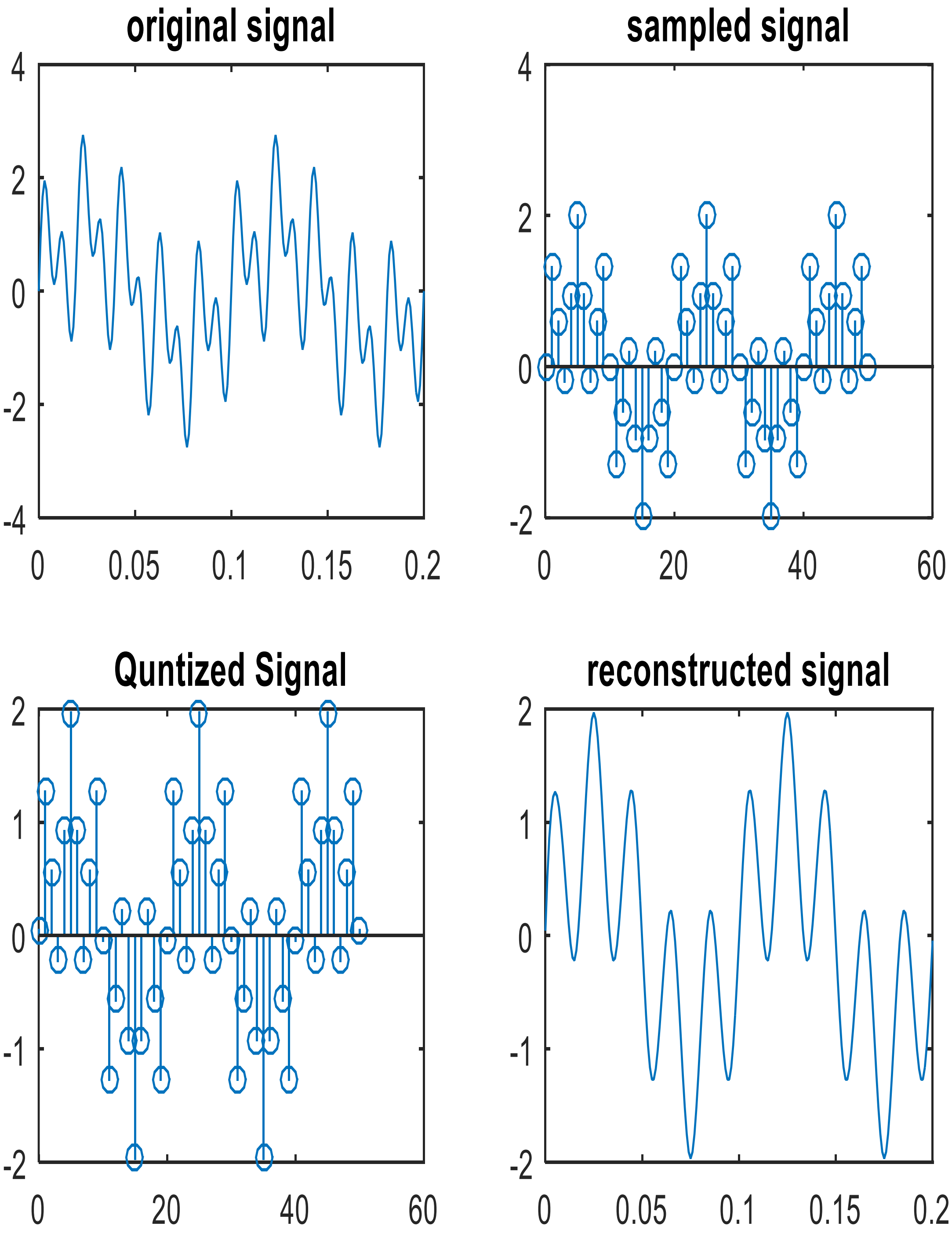
Output:



Uniform 4 bit Quantizer :



Uniform 6-bit Quantizer:



For 4-bit:
Quantization noise power: 0.0046
SQNR Experimental : 25.0935
SQNR formula = 25.8520

For 6-bit :
Quantization noise power: 0.0011
SQNR Experimental : 31.1920
SQNR formula = 37.8980

Comment: By increasing the number of bits, quantization noise power is decreasing due to increased number of quantization level and reconstructed signal will be more accurate.

Part C:

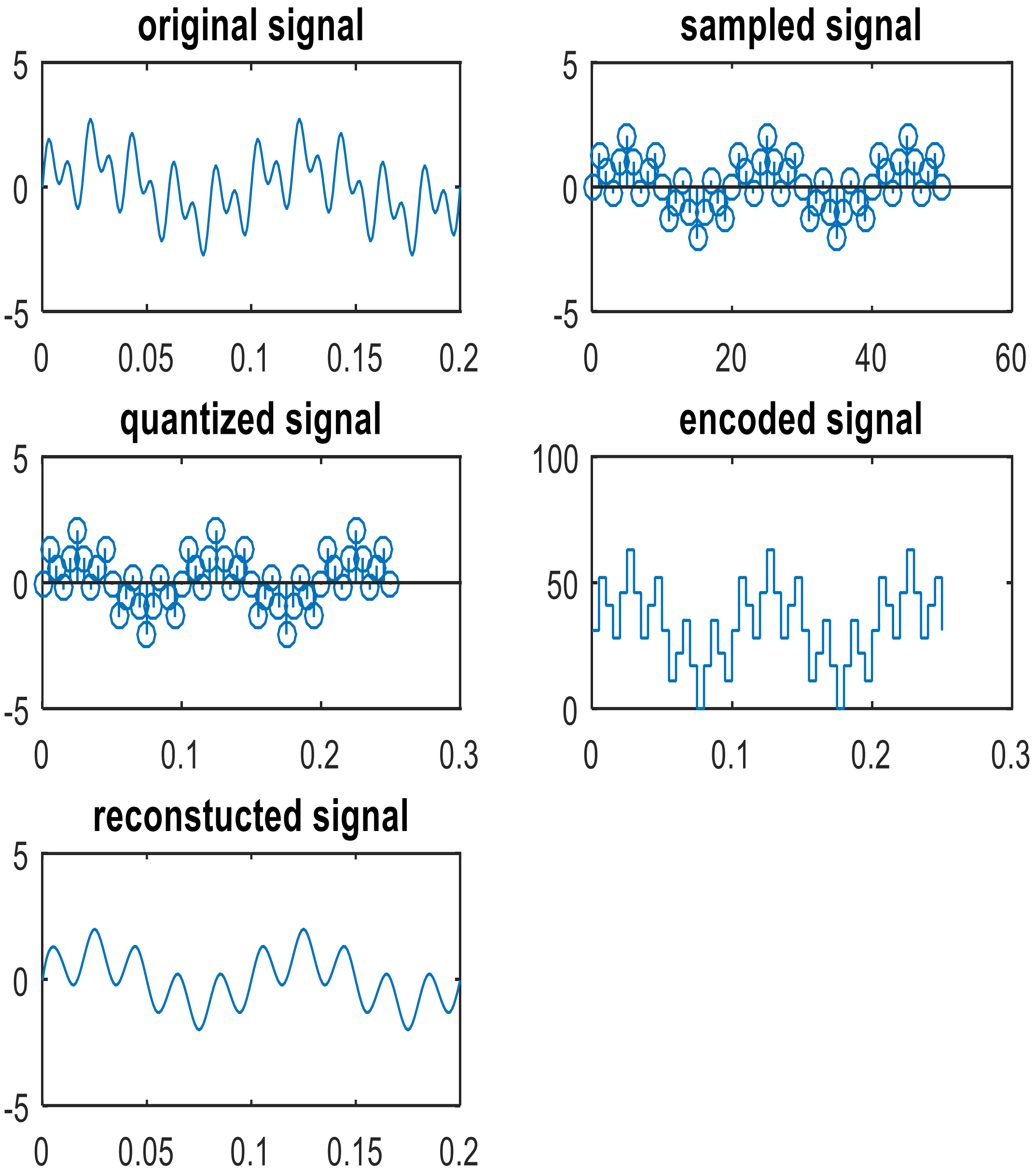
```
close all
clc;
t=0:0.001:0.2;
fs=200;

%original signal
y=sin(2*pi*10*t)+sin(2*pi*50*t)+sin(2*pi*100*t);
subplot(3,2,1);plot(t,y);title('original signal');
%sampling
n=0:1:50;
y_samp=sin(2*pi*(10/fs)*n)+sin(2*pi*(50/fs)*n)+sin(2*pi
*(100/fs)*n);
subplot(3,2,2);stem(n,y_samp);title('sampled signal');

%Uniform Quantizer
b=6;
L=2^b;
del=(max(y_samp)-min(y_samp))/(L-1);
for i=1:L
    l(i)=min(y_samp)+del*(i-1);
end
yq=y_samp;
for i=1:length(y_samp)
```

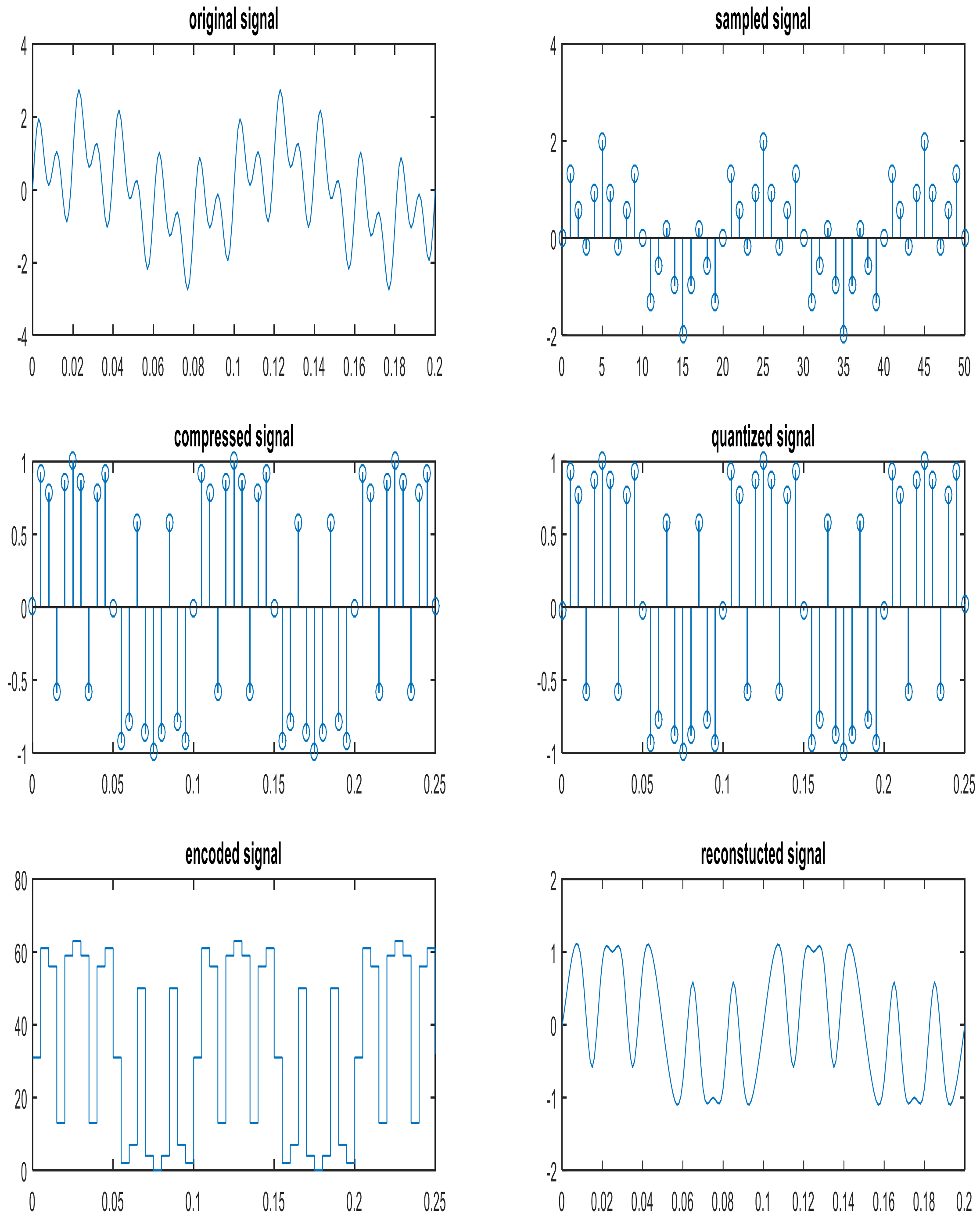
```
    for j=1:L-1
        if(y_samp(i)>l(j) && y_samp(i)<l(j+1))
            p=y_samp(i)-l(j);
            q=l(j+1)-y_samp(i);
            if(p>q)
                yq(i)=l(j+1);
            else
                yq(i)=l(j);
            end
        end
    end
end
subplot(3,2,3);stem(n/fs,yq);
title('quantized signal');
%encoding
for i=1:length(yq)
    y_encoded(i)=(yq(i)-min(yq))/del;
end
encoded_values=dec2bin(y_encoded);
subplot(3,2,4);stairs(n/fs,y_encoded);title('encoded
signal');
%reconstructed signal
y_rec=interp1(n/fs,yq,t,'spline');
subplot(3,2,5);plot(t,y_rec);
title('reconstucted signal');
```

Output:


original signal
5
0
-5
0
0.05
0.1
0.15
0.2
sampled signal
5
0
-5
0
20
40
60
quantized signal
5
0
-5
0
0.1
0.2
0.3
encoded signal
100
50
0
0
0.1
0.2
0.3
reconstucted signal
5
0
-5
0
0.05
0.1
0.15
0.2

Home task:

```
close all
clc;
t=0:0.001:0.2;
fs=200;
u=255;
%original signal
y=sin(2*pi*10*t)+sin(2*pi*50*t)+sin(2*pi*100*t);
subplot(3,2,1);plot(t,y);title('original signal');
%sampling
n=0:1:50;
y_samp=sin(2*pi*(10/fs)*n)+sin(2*pi*(50/fs)*n)+sin(2*pi
*(100/fs)*n);
subplot(3,2,2);stem(n,y_samp);title('sampled signal');
%normalization
ys_norm=y_samp/max(y_samp);
%figure;stem(n/fs,y_norm);
y_comp=log(1+u*abs(ys_norm))/log(1+u).*sign(ys_norm);
subplot(3,2,3);stem(n/fs,y_comp);title('compressed
signal');
%Uniform Quantizer
b=8;
L=2^b;
del=(max(y_comp)-min(y_comp))/(L-1);
for i=1:L
    l(i)=min(y_comp)+del*(i-1);
end
yq=y_comp;
for i=1:length(y_comp)
    for j=1:L-1
        if(y_comp(i)>l(j) && y_comp(i)<l(j+1))
            p=y_comp(i)-l(j);
            q=l(j+1)-y_comp(i);
            if(p>q)
                yq(i)=l(j+1);
            else
                yq(i)=l(j);
            end
        end
    end
end
subplot(3,2,4);stem(n/fs,yq);
title('quantized signal');
%encoding
for i=1:length(yq)
    y_encoded(i)=(yq(i)-min(yq))/del;
end
encoded_values=dec2bin(y_encoded);
subplot(3,2,5);stairs(n/fs,y_encoded);title('encoded
signal');
%reconstructed signal
y_rec=interp1(n/fs,yq,t,'spline');
subplot(3,2,6);plot(t,y_rec);
title('reconstucted signal');
```


original signal
4
2
0
-2
-4
0
0.02
0.04
0.06
0.08
0.1
0.12
0.14
0.16
0.18
0.2
sampled signal
4
2
0
-2
0
5
10
15
20
25
30
35
40
45
50
compressed signal
1
0.5
0
-0.5
-1
0
0.05
0.1
0.15
0.2
0.25
quantized signal
1
0.5
0
-0.5
-1
0
0.05
0.1
0.15
0.2
0.25
encoded signal
80
60
40
20
0
0
0.05
0.1
0.15
0.2
0.25
reconstucted signal
2
1
0
-1
-2
0
0.02
0.04
0.06
0.08
0.1
0.12
0.14
0.16
0.18
0.2