Performance of Waveform Coding Using PCM

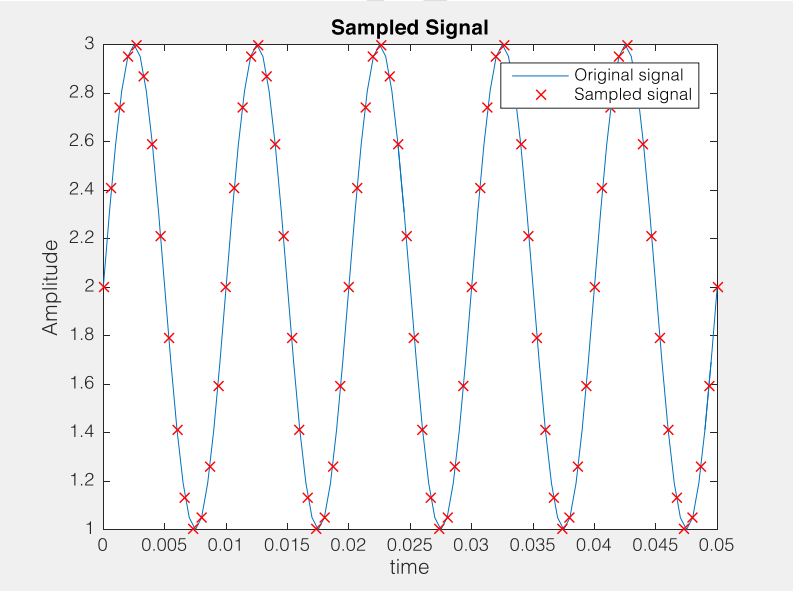
1. Generate a sinusoidal waveform with a DC offset so that it takes only positive amplitude value.
2. Sample and quantize the signal using a uniform quantizer with number of representation levels L. Vary L. Represent each value using decimal to binary encoder.
3. Compute the signal-to-noise ratio in dB.
4. Plot the SNR versus number of bits per symbol. Observe that the SNR increases linearly.

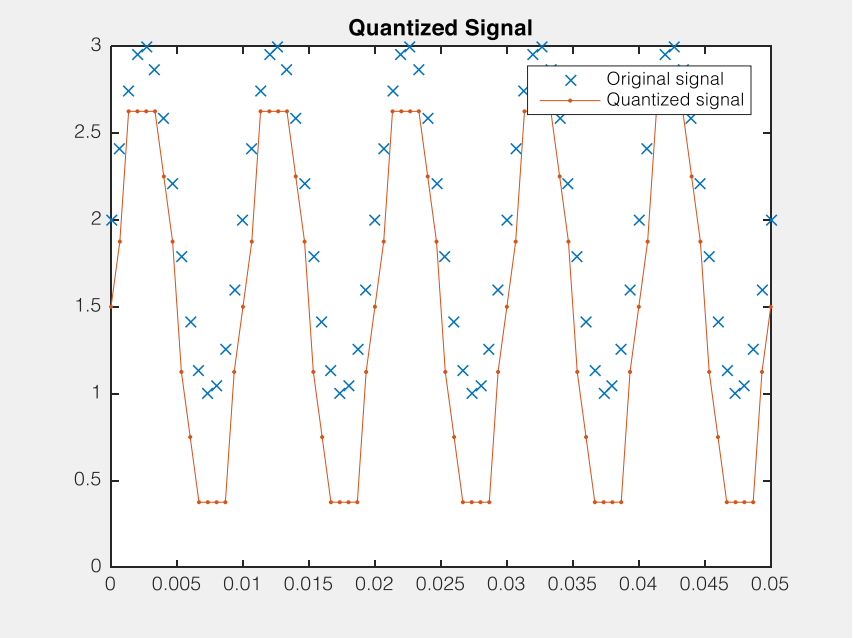
Program name: IMPL_quantiz_level_pcm.m

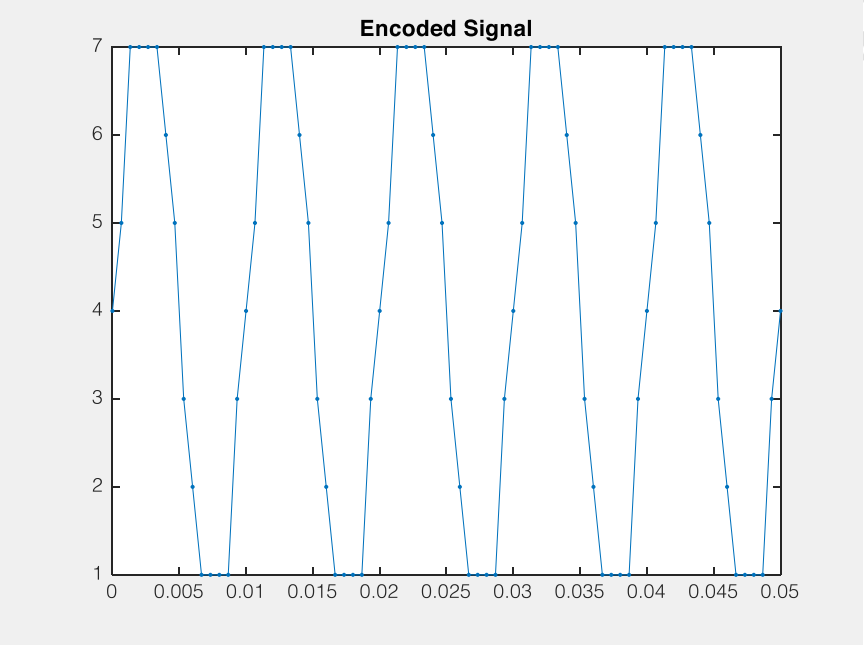
```
% Code created by Manu Prasad (IMPLearn)%
% Generate a sinusoidal waveform with a DC offset so that it
takes only
% positive amplitude value
clear;% clearing the variables
close all;% closing any opened figures
% Plotting the offset sinusoidal signal
time = 0:.0005:.05;
freq_msg=100; %wave form frequency
dc_ofst=2; % signal offset
signal=sin(2*pi*freq_msg*time)+dc_ofst; %Generating the signal
% plotting the signal
figure;plot(time,signal)
xlabel('time')
ylabel('Amplitude')
title('Signal')

% Sampling the signal
freq_sample=15*freq_msg; % sampling frequency
samp_time=0:1/freq_sample:0.05; % sampling time
samp_signal=dc_ofst+sin(2*pi*freq_msg*samp_time);% generating
the sampled signal
hold on
plot(samp_time,samp_signal,'rx') % plotting the sampled signal
title('Sampled Signal')
legend('Original signal','Sampled signal');

% Uniform Quantizer
L=8; %No of Quantization levels
smin=round(min(signal));
smax=round(max(signal));
Quant_levl=linspace(smin,smax,L); % Length 8, to represent 9
intervals
codebook = linspace(0,smax,L+1); % Length 9, one entry for
each interval
[index,quants] = quantiz(samp_signal,Quant_levl,codebook); %
Quantize.
```

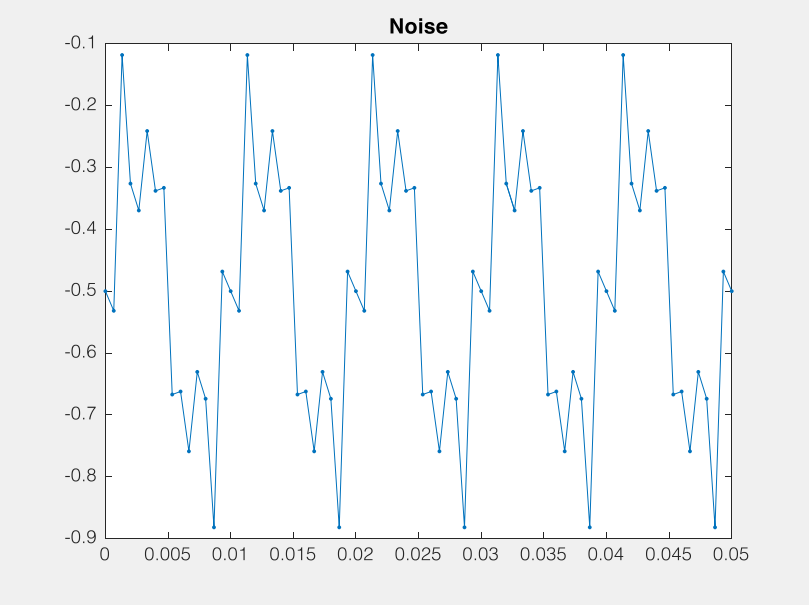
```
figure;plot(samp_time,samp_signal,'x',samp_time,quants,'.-')%
plotting sampled signal and quantization level
title('Quantized Signal')
legend('Original signal','Quantized signal');
figure;plot(samp_time,index,'.-')% plotting quantization
levels of input signal
title('Encoded Signal')

% % Quantization Levels plotting
% u = 0:0.01:1;
% y = uencode(u,4);
% figure;plot(u,y)
% title('Quantization Levels')

% Binary coding
for i=1:length(index)
    bincode_sig{i}=dec2bin(round(index(i)),7);
end
disp('binary encoded signal')
disp(bincode_sig')
% SNR ratio calculation
noise=quants-samp_signal; % calculating noise
figure;plot(samp_time,noise,'.-') % plotting figure
title('Noise')
r=snr(index,noise);% SNR
snr1=['SNR :',num2str(r)];
disp(snr1)
```

Output









Command window output

>> IMPL_quantiz_level_pcm

binary encoded signal

Columns 1 through 14

'0000100' '0000101' '0000111' '0000111' '0000111' '0000111' '0000110' '0000101' '0000011' '0000010' '0000001' '0000001' '0000001' '0000001'

Columns 15 through 28

'0000011' '0000100' '0000101' '0000111' '0000111' '0000111' '0000111' '0000110' '0000101' '0000011' '0000010' '0000001' '0000001' '0000001'

Columns 29 through 42

'0000001' '0000011' '0000100' '0000101' '0000111' '0000111' '0000111' '0000111' '0000110' '0000101' '0000011' '0000010' '0000001' '0000001'

Columns 43 through 56

'0000001' '0000001' '0000011' '0000100' '0000101' '0000111' '0000111' '0000111' '0000111' '0000110' '0000101' '0000011' '0000010' '0000001'

Columns 57 through 70

'0000001' '0000001' '0000001' '0000011' '0000100' '0000101' '0000111' '0000111' '0000111' '0000111' '0000110' '0000101' '0000011' '0000010'

Columns 71 through 76

'0000001' '0000001' '0000001' '0000001' '0000011' '0000100'

SNR :18.6833

To plot the Quantization level vs SNR changing the same program (**IMPL_quantiz_level_pcm.m**) Into a function (**IMPL_Quant.m**) and running it with **IMPL_Qlevel_vs_SNR.m**

Function name: IMPL_Quant

```
% Code generated by Manu Prasad %
% function for ploting Quant_level vs SNR
function [ r ] = IMPL_Quant( l )
    % Plotting the offset sinusoidal signal
    time = 0:.0005:.05;
    freq_msg=100; %wave form frequency
    dc_ofst=2; % signal offset
    signal=sin(2*pi*freq_msg*time)+dc_ofst; %Generating the
signal

    % Sampling the signal
    freq_sample=15*freq_msg; % sampling frequency
    samp_time=0:1/freq_sample:0.05; % sampling time
    samp_signal=dc_ofst+sin(2*pi*freq_msg*samp_time);%
generating the sampled signal

    % Uniform Quantizer
    L=l; %No of Quantization levels
    smin=round(min(signal));
    smax=round(max(signal));
    Quant_levl=linspace(smin,smax,L); % Length 8, to represent
9 intervals
    codebook = linspace(0.7,smax,L+1); % Length 9, one entry
for each interval
    [index,quants] = quantiz(samp_signal,Quant_levl,codebook);
% Quantize.

    % Binary coding
    for i=1:length(quants)
        bincode_sig{i}=dec2bin(round(quants(i)),3);
    end

    % SNR ratio calculation
    noise=quants-samp_signal; % calculating noise
    r=snr(index,noise);% SNR

end
```


Program Name: IMPL_Qlevel_vs_SNR.m

```
% Code created by Manu Prasad (IMPLearn)%
% Program for plotting quantization level vs SNR

l=[8,16,32,64,128];% defining different levels
for i=1:length(l)
    r(i) = IMPL_Quant(l(i));% calling the function
end
%Plotting
plot(l,r)
xlabel('L')
ylabel('SNR')
title('L vs SNR')
```

Output