**Problem 1:**

We call the maximum and minimum values of the sampled signal xs(nTs) over the interval [0, Ts], which we will call xmax­ and xmin, respectively.

xmax = 8cos(2π) = 8cos(2π) ≈ 8

xmin = –8

Step size Δq =

Therefore, we have the quantization signal xq(n)

xq(n) = round

**Code:**

**close all;**

**clear all;**

**xmin = -8;**

**xmax = 8;**

**n =3;**

**t= 0:0.0005:2;**

**x = 8\*cos(4\*pi\*t);**

**Ts = 0.01;**

**t2 = 0:Ts:2;**

**xs = 8\*cos(4\*pi\*t2);**

**figure**

**subplot(211)**

**plot(t,x, 'LineWidth',1.2)**

**xlabel('time (s)')**

**ylabel('x\_s(t)')**

**hold on**

**stem(t2,xs,'.')**

**title('Original and Sampled signals')**

**set(gca, "linewidth", 1, "fontsize", 18);**

**grid on**

**legend('x(t)', 'x\_s(t)')**

**v = xmin:0.001:xmax;**

**lsb = (xmax-xmin)/2^n;**

**function [xq] = quantizer(x, n, vmin, vmax)**

**% this is the quantizer function, this produced the quantized bits as outputs**

**lsb = (vmax-vmin)/2^n;**

**levels = vmin+lsb\*(1:(2^n-1));**

**xq = zeros(size(x));**

**for i = 1:(2^n-1)**

**xq = xq+(x>=levels(i));**

**end**

**end**

**function [xq] = zerohold(x, t, t2)**

**% this generates the zero hold signal**

**xq = zeros(size(t));**

**for i = 1:length(t)**

**xq(i) = x(find(t2<=t(i), 1, 'last'));**

**end**

**end**

**figure**

**quant = quantizer(v, n, xmin, xmax);**

**subplot(211)**

**plot(v, quant, 'LineWidth', 1.2)**

**xlabel('V ')**

**ylabel('Quantized bits')**

**title('Quantization Function')**

**set(gca, "linewidth", 1, "fontsize", 18);**

**grid on**

**xq = quantizer(xs, n, xmin, xmax);**

**fprintf("The first 10 encoded bits are :\n" )**

**dec2bin(xq(1:10))**

**xq = zerohold(xq, t, t2);**

**figure(1)**

**xq = xmin+xq\*lsb;**

**subplot(212)**

**plot(t,x, 'LineWidth',1.2)**

**xlabel('time (s)')**

**ylabel('x\_q(t)')**

**hold on**

**plot(t,xq, 'LineWidth',1.2)**

**legend('x(t)', 'x\_q(t)')**

**title('Original signal and Quantized Signal')**

**set(gca, "linewidth", 1, "fontsize", 18);**

**grid on**

**figure(2)**

**subplot(212)**

**plot(t,x-xq, 'LineWidth',1.2)**

**xlabel('time (s)')**

**ylabel('n\_q(t) = x(t)-x\_q(t)')**

**title('Quantization Error')**

**set(gca, "linewidth", 1, "fontsize", 18);**

**grid on**

**Graphical user interface

Description automatically generated**

**Problem 2:**

The code and the graph of the quantization error 𝜀[𝑛] is executed in problem 1

Code sequence represent the 8 levels of quantization:

**% Define the quantization levels**

**Q = 8;**

**gay = linspace(-Q/2, Q/2, Q);**

**function code = encode\_level(level, gay)**

**index = find(gay == level);**

**code = dec2bin(index-1, ceil(log2(length(gay))));**

**if (length(code) < ceil(log2(length(gay))))**

**code = ['0'\*(ceil(log2(length(gay)))-length(code)), code];**

**endif**

**endfunction**

**% Define the quantization step**

**delta = gay(2) - gay(1);**

**% Display the code sequence**

**disp("Quantization Code Sequence:");**

**for i = 1:length(gay)**

**code = encode\_level(gay(i), gay);**

**fprintf("x̂(nTs) = %g -> %s\n", gay(i), code);**

**endfor**

**Text, letter

Description automatically generated**

**Problem 3:**

(a)

Code:

**function answer = quantized\_example(x, R, B)**

**delta = R/(2^B);**

**answer = delta\*floor(x/delta+0.5);**

**end**

(b)

**x=-8:1:7;**

**R=16;**

**B=3;**

**function answer = quantized\_example(x, R, B)**

**delta = R/(2^B);**

**answer = delta\*floor(x/delta+0.5);**

**end**

**y=quantized\_example(x, R, B);**

**Text, letter

Description automatically generated**

(c)

**figure**

**subplot(211);**

**plot(x);**

**title('Input Signal');**

**xlabel('Sample Number');**

**ylabel('Amplitude (V)');**

**set(gca, "linewidth", 1, "fontsize", 18);**

**grid on**

**subplot(212);**

**stem(y);**

**title('Quantized Signal');**

**xlabel('Sample Number');**

**ylabel('Amplitude (V)');**

**set(gca, "linewidth", 1, "fontsize", 18);**

**grid on**

Graphical user interface

Description automatically generated