EXPERIMENT – 3

PULSE CODE MODULATION

February 4 - 2025

B Siddharth Sekhar - EC22B1064

Aim:

To implement Pulse Code Modulation (PCM) of a sinusoidal signal using built-in and custom functions for various quantization levels (N = 1, 2, 3, and 4 bits). Simulating the PCM process, plotting the message, quantized, encoded, quantization error, and decoded signals along with their spectra, and calculating key parameters such as quantization levels, step size, maximum error, and SQNR (in dB).

Theory:

PULSE CODE MODULATION (FM)

Pulse Code Modulation (PCM) is a digital representation technique where an analog signal x(t) (with amplitude in the range $[-x_{tmax}, x_{tmax}]$ is sampled at a uniform rate and then quantized into $L = 2^N$ discrete levels, where N is the number of bits per sample. The quantizer's step size is given by:

$$\Delta = \frac{2\mathrm{x_{tmax}}}{2^{\mathrm{N}}}$$

Each sample is rounded to the nearest quantization level, introducing a quantization error defined as:

QE=x(t)−xq(t)

The theoretical quantization noise power is approximated by:

$$P_{noise} = \Delta^2/12$$

The Signal-to-Quantization Noise Ratio (SQNR) in decibels is calculated as:

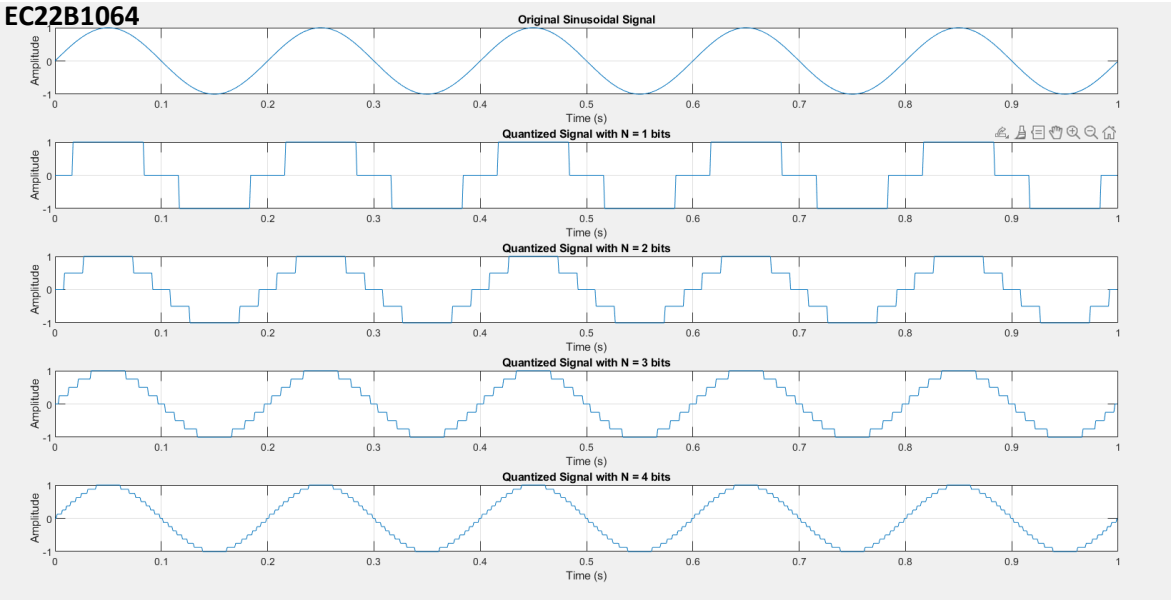
$$SQNR = 10 * \log_{10}(Signal\ Power)/(\Delta^2/12)$$

In practice, SQNR is also computed using the variance of the quantization error, allowing a comparison between theoretical and experimental values.

Q1) Pulse Code Modulation (PCM) for various levels of quantization.

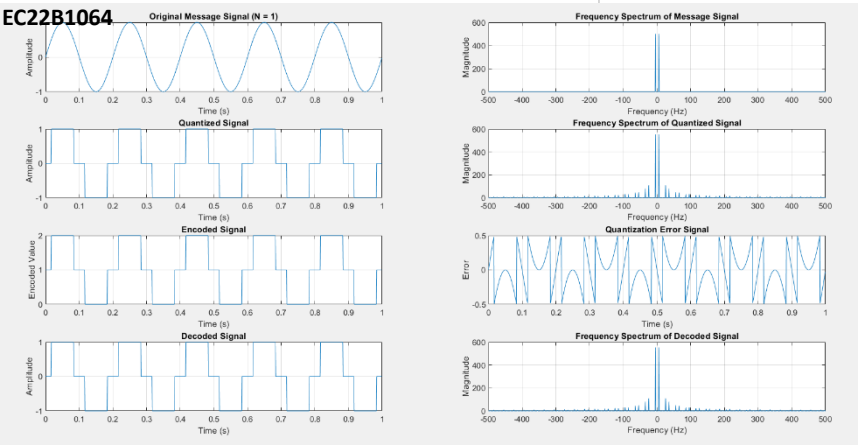
```
fs = 1000;
t = 0:1/fs:1-1/fs;
f = 5;
signal = sin(2 * pi * f * t);

N_values = [1, 2, 3, 4];

figure;
subplot(5, 1, 1);
plot(t, signal);
title('Original Sinusoidal Signal');
xlabel('Time (s)');
ylabel('Amplitude');
grid on;
for i = 1:length(N_values)
    N = N_values(i);
    levels = 2^N;
    step_size = 2 / (levels);

    quantized_signal = round((signal + 1) / step_size) * step_size - 1;
     y = uencode(quantized_signal,2^N);
     ybin = dec2bin(y,2^N);

     subplot(5, 1, i + 1);
     plot(t, quantized_signal);
     title(['Quantized Signal with N = ', num2str(N), ' bits']);
     xlabel('Time (s)');
     ylabel('Amplitude');
     grid on;
 end
```

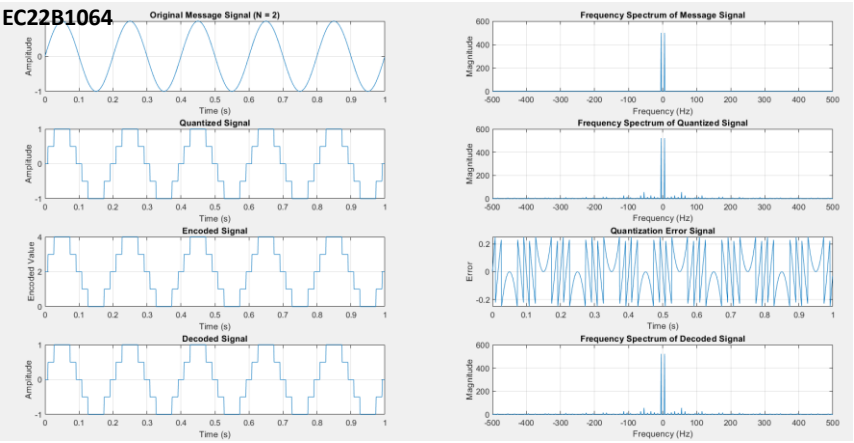
EC22B1064

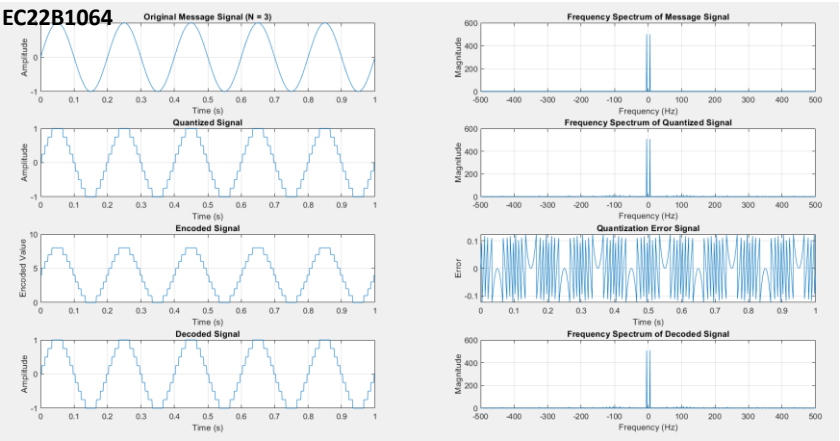


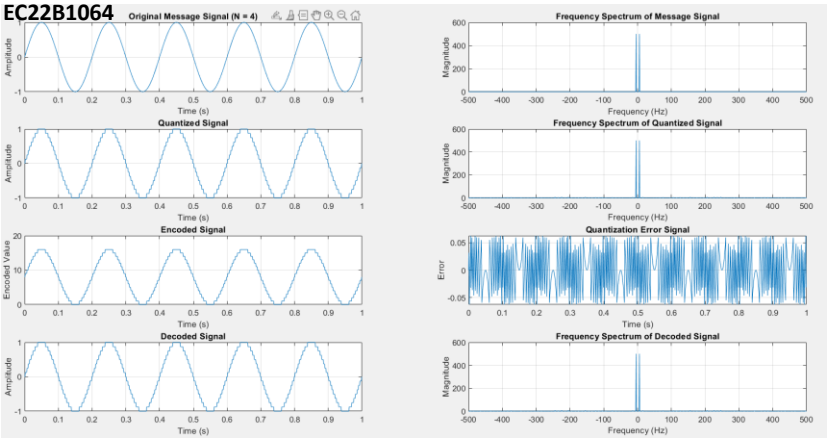
Q2) Plot the message signal, quantized signal, encoded signal, quantization error signal, and decoded signal.

```
fs = 1000;
t = 0:1/fs:1-1/fs;
f = 5;
signal = sin(2 * pi * f * t);

for N = 1:4
    levels = 2^N;
    step_size = 2 / levels;
    quantized_signal = round((signal + 1) / step_size) * step_size - 1;
    encoded_indices = round((quantized_signal + 1) / step_size);
    decoded_signal = (encoded_indices * step_size) - 1;
    quantization_error = signal - quantized_signal;

    message_f = abs(fftshift(fft(signal)));
    quantized_f = abs(fftshift(fft(quantized_signal)));
    decoded_f = abs(fftshift(fft(decoded_signal)));

    M = length(signal);
    freq_axis = linspace(-fs/2, fs/2, M);

    figure;

    subplot(4, 2, 1); plot(t, signal); title(['Original Message Signal (N = ', num2str(N), ')']); xlabel('Time (s)'); ylabel('Amplitude'); grid on;

    subplot(4, 2, 2); plot(freq_axis, message_f); title('Frequency Spectrum of Message Signal'); xlabel('Frequency (Hz)'); ylabel('Magnitude'); grid on

    subplot(4, 2, 3); plot(t, quantized_signal); title('Quantized Signal'); xlabel('Time (s)'); ylabel('Amplitude'); grid on;

    subplot(4, 2, 4); plot(freq_axis, quantized_f); title('Frequency Spectrum of Quantized Signal'); xlabel('Frequency (Hz)'); ylabel('Magnitude'); gri

    subplot(4, 2, 5); plot(t, encoded_indices); title('Encoded Signal'); xlabel('Time (s)'); ylabel('Encoded Value'); grid on;

    subplot(4, 2, 6); plot(t, quantization_error); title('Quantization Error Signal'); xlabel('Time (s)'); ylabel('Error'); grid on;

    subplot(4, 2, 7); plot(t, decoded_signal); title('Decoded Signal'); xlabel('Time (s)'); ylabel('Amplitude'); grid on;

    subplot(4, 2, 8); plot(freq_axis, decoded_f); title('Frequency Spectrum of Decoded Signal'); xlabel('Frequency (Hz)'); ylabel('Magnitude'); grid on
end
```









Q3) Determine the number of quantization levels, step size, maximum quantization error, and the SQNR

```
N_values = [1 2 3 4];
fs = 1000;
t = 0:1/fs:1-1/fs;
f = 5;
signal = sin(2 * pi * f * t);
for i = 1:length(N_values)
    N = N_values(i);
    L = 2^N;
    step_size = 2*max(abs(signal))/L;
    quantized_signal = round((signal + 1) / step_size) * step_size - 1;
    ns = signal-quantized_signal;
    fprintf("At N = %d\n",N);
    disp("Value of L");
    disp(L);
    fprintf('The number of quantization levels are: %d\n', L);
    fprintf("The Step size is %f\n",step_size);
    max_error = max(signal-quantized_signal);
    fprintf("The maximum quantization error is: %f\n",max_error);
    sqnr = var(signal)/var(ns);
    sqnr_db = 10*log10(sqnr);
    fprintf("The SQNR value is: %f\n",sqnr_db);

end
```

```
At N = 1
Value of L
     2

The number of quantization levels are: 2
The Step size is 1.000000
The maximum quantization error is: 0.490959
The SQNR value is: 8.929358
At N = 2
Value of L
     4

The number of quantization levels are: 4
The Step size is 0.500000
The maximum quantization error is: 0.249889
The SQNR value is: 14.572311
At N = 3
Value of L
     8

The number of quantization levels are: 8
The Step size is 0.250000
The maximum quantization error is: 0.124667
The SQNR value is: 20.350337
At N = 4
Value of L
    16

The number of quantization levels are: 16
The Step size is 0.125000
The maximum quantization error is: 0.062381
The SQNR value is: 26.091952
>>
```

Q4) Tabulate the above results and compare the values obtained for SQNR with the expected theoretical values.

```
N_values = [1 2 3 4];
fs = 1000;
t = 0:1/fs:1-1/fs;
f = 5;
signal = sin(2 * pi * f * t);
theo = [0 0 0 0 ];
prac = [0 0 0 0];
for i = 1:length(N_values)
    N = N_values(i);
    L = 2^N;
    step_size = (2*max(signal))/L;
    quantized_signal = round((signal + 1) / step_size) * step_size - 1;
    n1 = (step_size)^2/12;
    sqnr1 = 10*log10(var(signal)/n1);
    theo(i)=sqnr1;
    err = signal - quantized_signal;
    sqnr2 = 10*log10(var(signal)/var(err));
    prac(i)=sqnr2;
end
T = table(prac,theo);
disp(T);
```

```
>> EC22B1064_lab3_q4
```

prac				theo			
8.9294	14.572	20.35	26.092	7.7859	13.806	19.827	25.848

Inference:

- **PCM Process Characteristics:** In Pulse Code Modulation, the analog signal is sampled, quantized, encoded, and finally decoded, with the quantization level directly dependent on the number of bits NNN.
- **Quantization Levels and Resolution:** Increasing NNN increases the number of levels 2^N and reduces the step size Δ, thereby lowering the quantization error.
- **Error and SQNR:** The quantization error diminishes with higher bit resolutions, leading to an improved SQNR. The observed practical SQNR values closely align with the theoretical predictions.
- **Spectral Integrity:** The spectra of the original, quantized, and decoded signals

demonstrate that the essential frequency components are preserved despite the presence of quantization noise.

Conclusion:

The PCM simulation successfully digitized the sinusoidal signal, with higher bit resolutions yielding reduced quantization error and improved SQNR. Both time-domain and frequency-domain analyses confirmed that the core characteristics of the original signal were maintained after processing. The strong correlation between practical and theoretical SQNR values validates the PCM approach and underscores its importance in digital signal processing applications

References:

[1] Simon Haykins, Communication systems, 2nd ed. (New York John Wiley and Sons, 2005).