EXPERIMENT – 4

DELTA MODULATION

February 11 - 2025

B Siddharth Sekhar - EC22B1064

Aim:

To implement Delta Modulation and Demodulation of a sinusoidal signal in MATLAB, analyze slope overload distortion and granular noise for different step sizes (Δ) and compare the reconstructed signal with the original message in both time and frequency domains.

Theory:

Delta Modulation (DM):

Delta Modulation is a digital pulse modulation method where the difference between successive samples of the message signal is encoded into a single bit. It reduces bandwidth by recording whether the signal is increasing or decreasing, rather than encoding the entire sample value.

Working Principle:

1. **Quantization Process:** DM uses a one-bit quantizer and a feedback loop to approximate the input signal. The error between the current and previous sample is calculated.
2. **Encoding:** If the error is positive, it is encoded as '1'; if negative, as '0'.
3. **Demodulation:** The encoded signal is integrated to reconstruct the original signal, and a low-pass filter smooths the approximation.

Error Signal:

e[n]=m[n]−$m_q[n-1]$

where m[n] is the original message signal

and mq[n−1] is the previous approximation.

Quantized Error Signal:

$e_q[n]$ =Δ·sgn(e[n])

where Δ is the step size, and sgn(e[n]) determines

the direction of change.

Sampling Rate:

$fs = 1/T_s$

where f_s is the sampling frequency.

Oversampling at $f_s = 4f_m$ improves correlation between adjacent samples.

Noise in Delta Modulation:

Delta Modulation faces two types of noise:

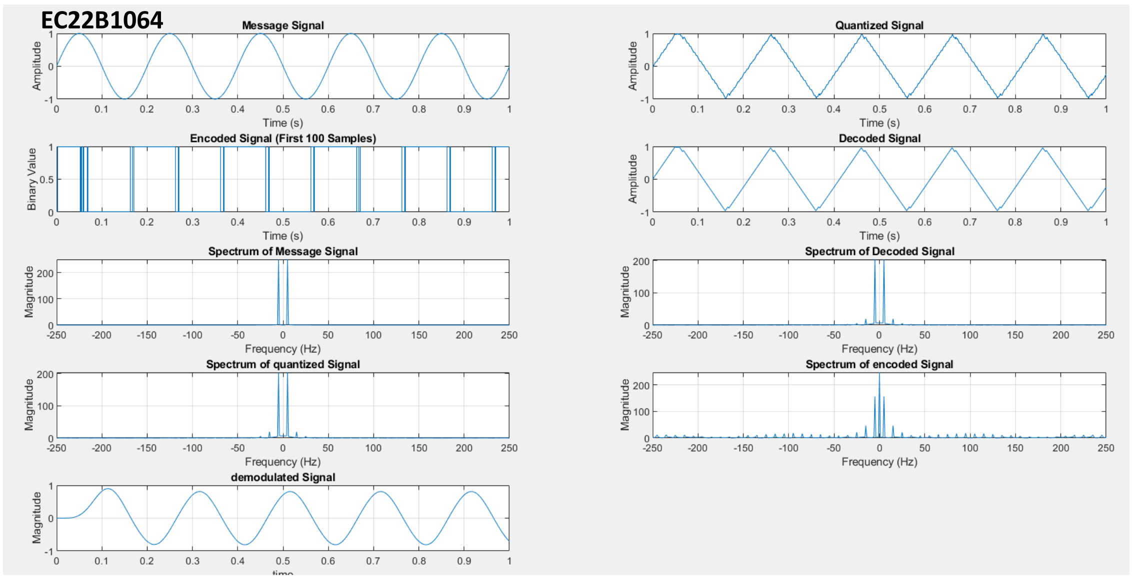
1. **Slope Overload Distortion:** Occurs when the step size Δ\DeltaΔ is too small to track rapid signal changes.
2. **Granular Noise:** Occurs when the step size is too large, causing oscillations in flat signal segments.

To avoid slope overload distortion:

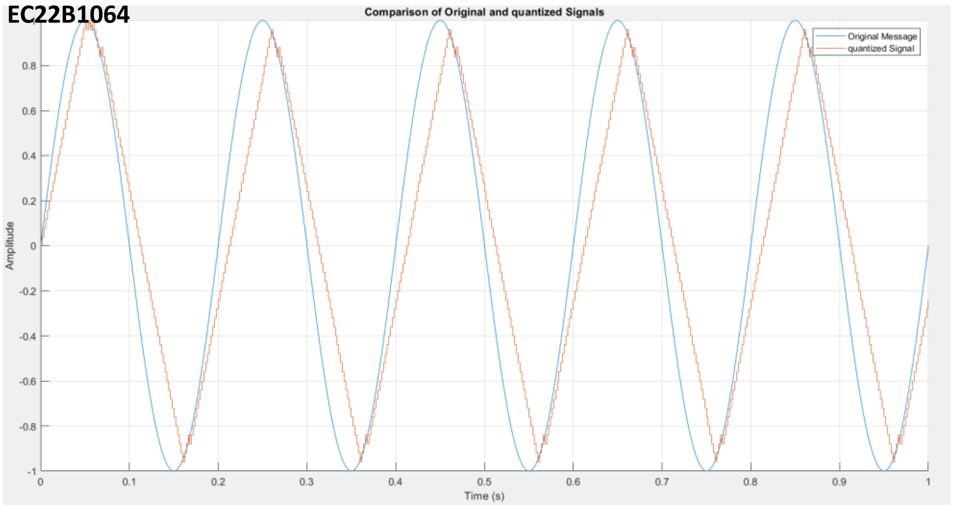
$$\frac{\Delta}{T_s} \geq \max\left(\frac{dm(t)}{dt}\right)$$

Q1) Implementation of Delta modulation and demodulation and plotting of signals with their spectra

```
fs = 1000;
t = 0:1/fs:1;
f = 5;
message_signal = sin(2*pi*f*t);
max_slope = 2*pi*f;
delta = (max_slope/fs);

delta_slope_overload = delta*0.8;
delta_granular = delta*10;

y_slope = zeros(1, length(message_signal));
y_granular = zeros(1, length(message_signal));
encoded_slope = zeros(length(message_signal));
encoded_granular = zeros(length(message_signal));

for i = 2:length(message_signal)

    if message_signal(i) >= y_slope(i-1)
        y_slope(i) = y_slope(i-1) + delta_slope_overload;
        encoded_slope(i) = 1;
    else
        y_slope(i) = y_slope(i-1) - delta_slope_overload;

    end

    if message_signal(i) >= y_granular(i-1)
        y_granular(i) = y_granular(i-1) + delta_granular;
        encoded_granular(i) = 1;
    else
        y_granular(i) = y_granular(i-1) - delta_granular;
        encoded_granular(i) = 0;
    end
end


figure;
subplot(3,1,1);
plot(t, message_signal);
title('Message Signal');
xlabel('Time (s)');
ylabel('Amplitude');
grid on;


subplot(3,1,2);
plot(t, y_slope);
title('Quantized Signal (Slope Overload Distortion)');
xlabel('Time (s)');
ylabel('Amplitude');
grid on;

subplot(3,1,3);
plot(t, y_granular);
title('Quantized Signal (Granular Noise)');
xlabel('Time (s)');
ylabel('Amplitude');
grid on;
```



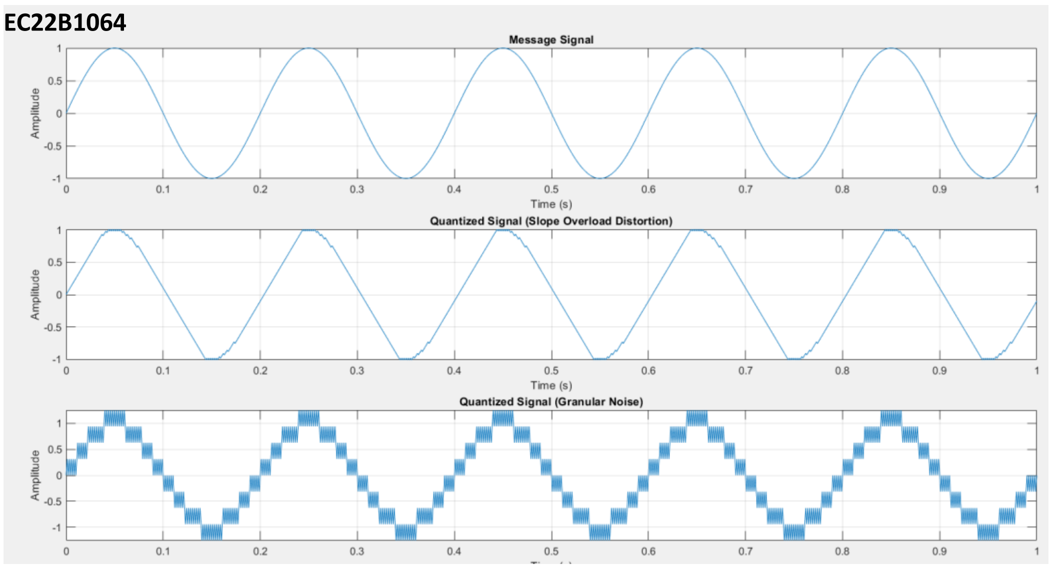
EC22B1064



Q2)slope overload distortion and granular noise

```
fs = 1000;
t = 0:1/fs:1;
f = 5;
message_signal = sin(2*pi*f*t);
max_slope = 2*pi*f;
delta = (max_slope/fs);

delta_slope_overload = delta*0.8;
delta_granular = delta*10;

y_slope = zeros(1, length(message_signal));
y_granular = zeros(1, length(message_signal));
encoded_slope = zeros(length(message_signal));
encoded_granular = zeros(length(message_signal));

for i = 2:length(message_signal)

    if message_signal(i) >= y_slope(i-1)
        y_slope(i) = y_slope(i-1) + delta_slope_overload;
        encoded_slope(i) = 1;
    else
        y_slope(i) = y_slope(i-1) - delta_slope_overload;
        encoded_slope(i) = 0;
    end

    if message_signal(i) >= y_granular(i-1)
        y_granular(i) = y_granular(i-1) + delta_granular;
        encoded_granular(i) = 1;
    else
        y_granular(i) = y_granular(i-1) - delta_granular;
        encoded_granular(i) = 0;
    end
end


figure;
subplot(3,1,1);
plot(t, message_signal);
title('Message Signal');
xlabel('Time (s)');
ylabel('Amplitude');
grid on;

subplot(3,1,2);
plot(t, y_slope);
title('Quantized Signal (Slope Overload Distortion)');
xlabel('Time (s)');
ylabel('Amplitude');
grid on;

subplot(3,1,3);
plot(t, y_granular);
title('Quantized Signal (Granular Noise)');
xlabel('Time (s)');
ylabel('Amplitude');
grid on;
```

EC22B1064



Inference:

- **Delta Modulation Working:** DM effectively approximates a sinusoidal message signal using a stepwise approach, encoding only the difference between successive samples.
- **Effect of Step Size (Δ):** A small Δ leads to slope overload distortion, while a large Δ results in granular noise. Choosing an optimal Δ minimizes errors.
- **Reconstruction Accuracy:** The demodulated signal closely resembles the original signal when the correct step size and sampling rate are used.
- **Spectral Analysis:** The frequency domain representation of the message, encoded, quantized, and decoded signals shows the impact of modulation and demodulation.

Conclusion:

The Delta Modulation experiment successfully demonstrated the encoding and decoding of a sinusoidal signal. By selecting an appropriate step size, we minimized slope overload distortion and granular noise. The reconstructed signal retained the primary characteristics of the original signal, verifying the effectiveness of Delta Modulation in digital communication.

References:

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