Department of Electronic & Telecommunication Engineering
University of Moratuwa

EN2074 - Communication Systems Engineering

Simulation Assignment
Eye diagrams and Equalization

Team Members

1. 210387D Mihiranga N.G.D.
2. 210391J Morawakgoda M.K.I.G.

Contents

1. Introduction

Every communication system comprises three essential components: the transmitter, the medium, and the receiver. In the context of digital communication systems, the transmission process involves passing a modulated baseband signal through a Pulse Shaping Filter before sending it across the medium. It is crucial to account for the errors induced by the channel, particularly as it behaves akin to a low-pass filter. Communication engineers strive to enhance the robustness of communication systems against these errors by meticulously analyzing the channel's susceptibility and fine-tuning both transmitter and receiver circuits to mitigate such discrepancies. Utilizing engineering tools like "Eye Diagrams" facilitates this analysis, providing invaluable insights into system performance.

This report explores the performance analysis of a digital communication system across diverse scenarios. Leveraging baseband 2-PAM signaling, the study explores the efficacy of different pulse shaping filters in enhancing robustness against noise, sampling time discrepancies, and synchronization errors. Furthermore, it investigates the impact of an additive white Gaussian noise (AWGN) channel on system performance and devises a zero-forcing (ZF) equalizer tailored for multipath channels.

Pulse amplitude modulation (PAM)

Pulse Amplitude Modulation (PAM) is a type of modulation where the amplitude of a pulse signal is altered to convey digital information. In this assignment, we are generating 2-PAM, such that two symbols are modulated using two amplitude levels, typically represented by +1 and -1. These levels correspond to binary values of 1 and 0, respectively, allowing for binary data transmission. The modulation involves changing the pulse amplitude at discrete time intervals according to the sequence of symbols being transmitted. 2-PAM is akin to Binary Phase Shift Keying (BPSK) modulation, as both result in similar positions in the constellation diagram.

Pulse Shaping

Pulse shaping plays a crucial role in ensuring reliable communication by mitigating inter-symbol interference (ISI) and reducing susceptibility to channel distortions and noise. While rectangular pulses offer zero ISI, their high bandwidth requirements make them prone to greater susceptibility to channel distortions. Sinc pulses satisfy Nyquist's criteria but are not time limited. Raised cosine pulses provide a compromise between bandwidth efficiency and ISI suppression, offering adjustable excess bandwidth to meet system requirements.

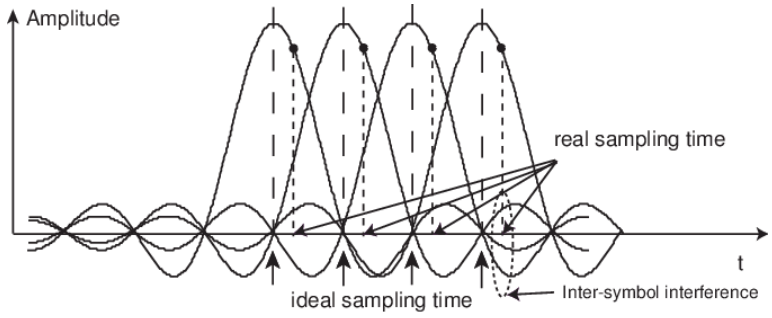


Figure 1: Inter Symbol Interference (ISI)

Additive White Gaussian Noise (AWGN)

AWGN, characterized by noise added to the original signal with a flat power spectral density across all frequencies, poses a significant challenge in digital communication. Understanding its properties, such as zero mean and variance dependent on power spectral density, is essential for effective noise management strategies.

Eye Diagram

Eye diagrams provide valuable insights into signal quality by superimposing multiple waveform segments on a single plot, resembling the shape of an "eye." They visualize signal integrity, with the horizontal axis representing time and the vertical axis representing the amplitude or voltage level of the received signal.

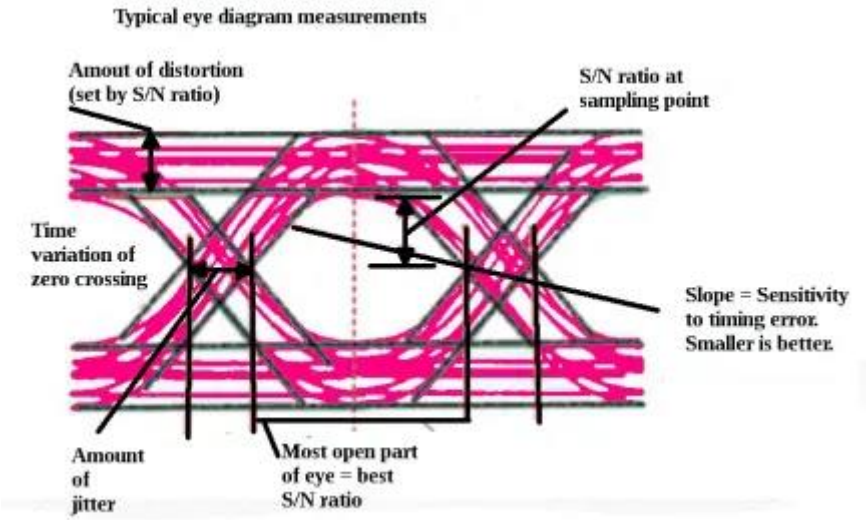


Figure 2: Eye Diagram

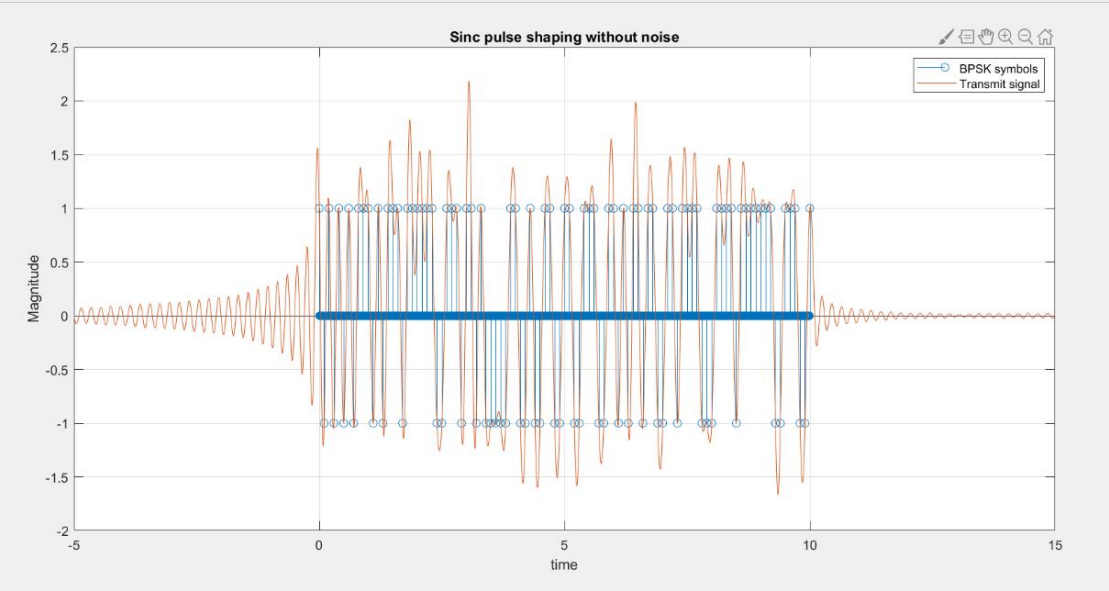
Channel Equalization

Channel equalization techniques are indispensable for compensating for frequency-dependent amplitude and phase characteristics of communication channels. These techniques counteract impairments like multipath fading, ISI, and noise, ensuring reliable data transmission. Equalizers, such as zero-forcing (ZF) and minimum mean square error (MMSE) equalizers, adaptively adjust the amplitude and phase of received signals to align them with the original transmitted signals, thereby mitigating the impact of channel-induced distortions.

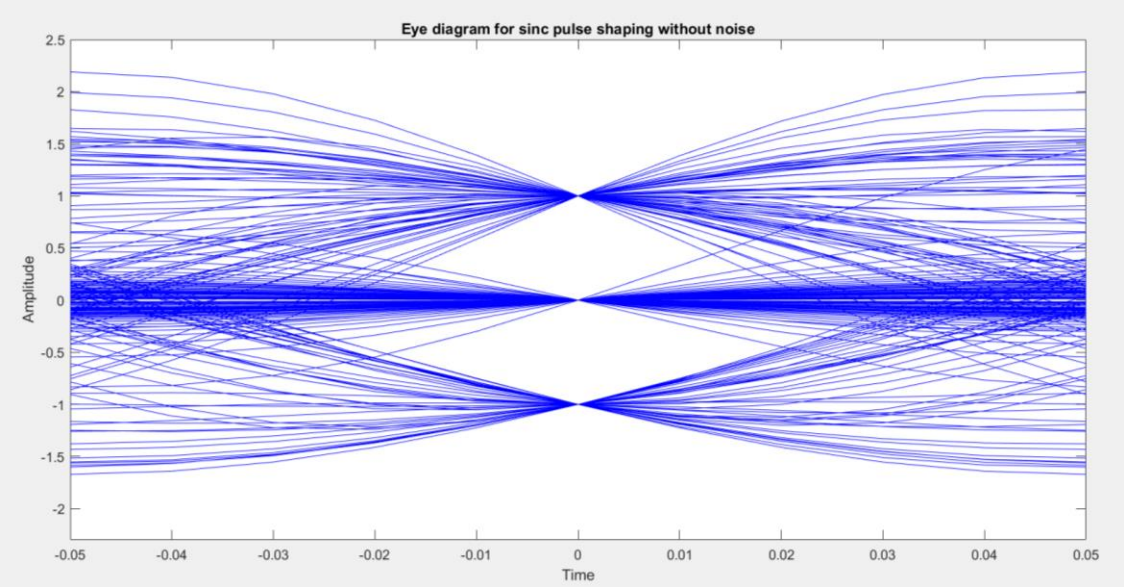
2. Discussion

2.1. Task 1

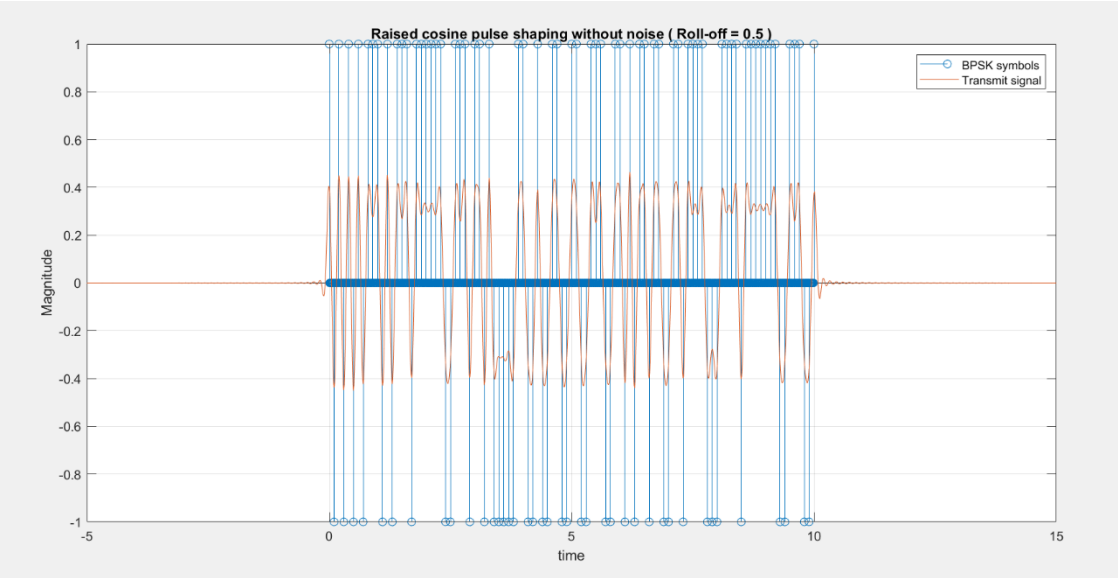
In this task we generated 1000 data bits with a bit rate of 10bits per second. Then converted them into BPSK symbols where bit 1 is represented as +1 and bit 0 is represented as -1. After that, we convolved the generated BPSK symbols with a Sinc pulse with sampling period 1s to generate the transmit signal.



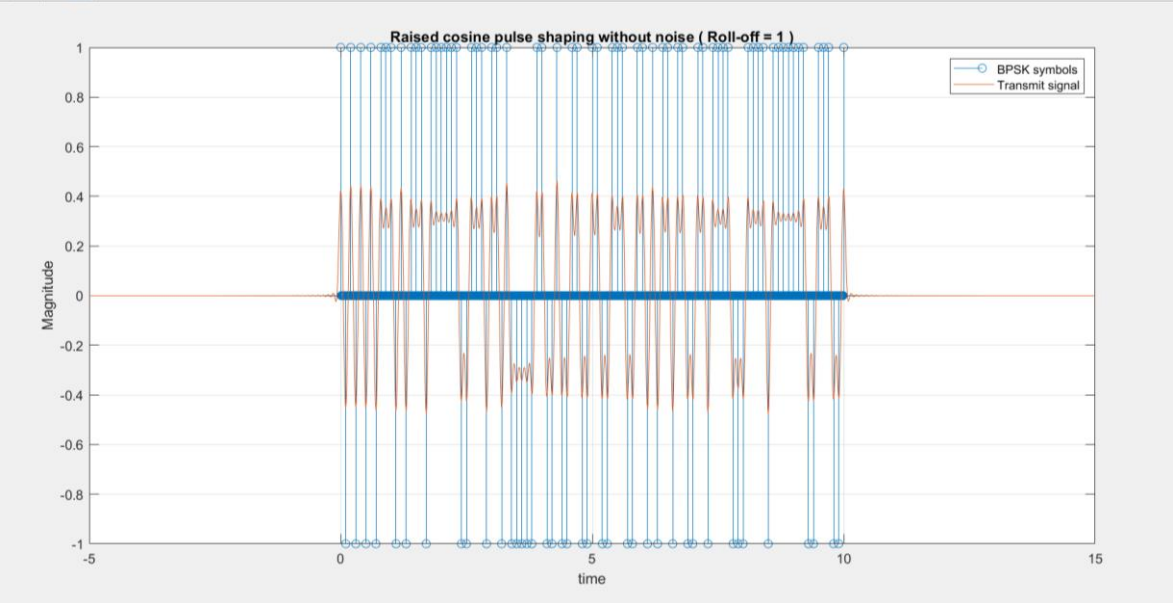
The eye diagram for the transmitted signal.



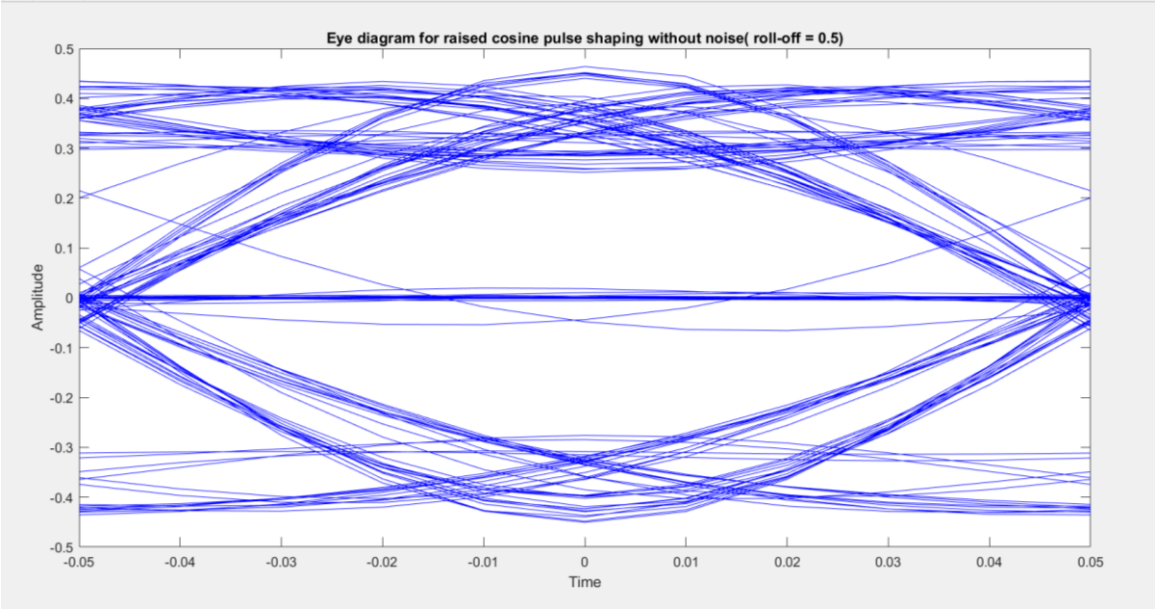
Repeated the previous step from raised cosine pulses having roll-off factor = 0.5.



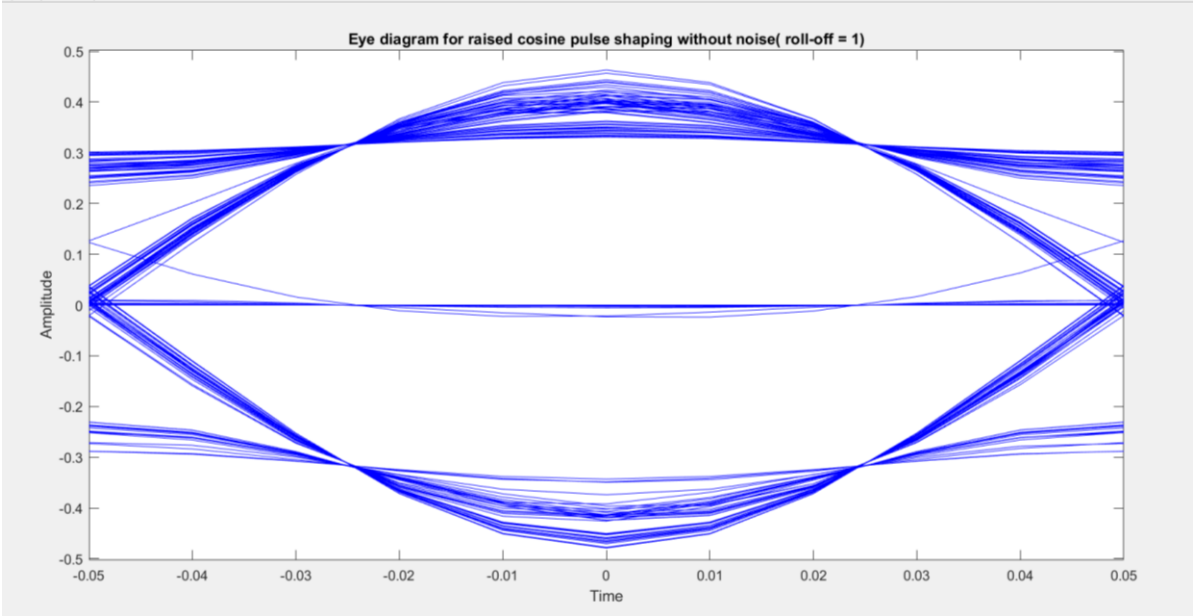
Repeated the previous step from raised cosine pulses having roll-off factor = 1.



Eye diagram for raised cosine pulses having roll-off factor = 0.5.



Eye diagram for raised cosine pulses having roll-off factor = 1.

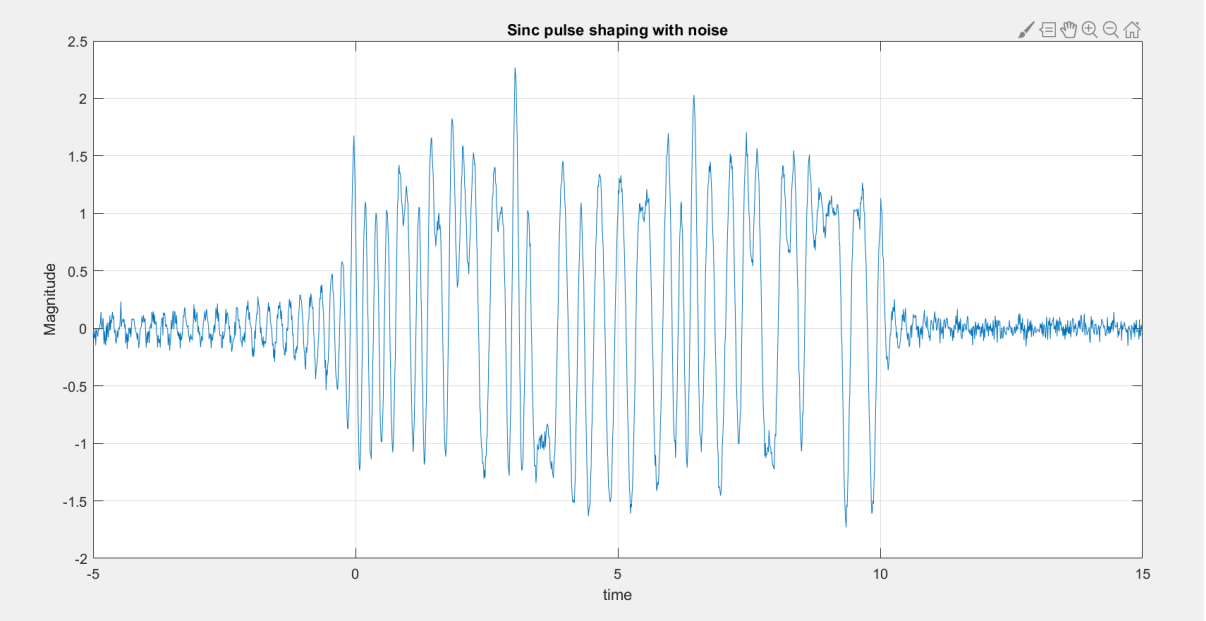


Comparison

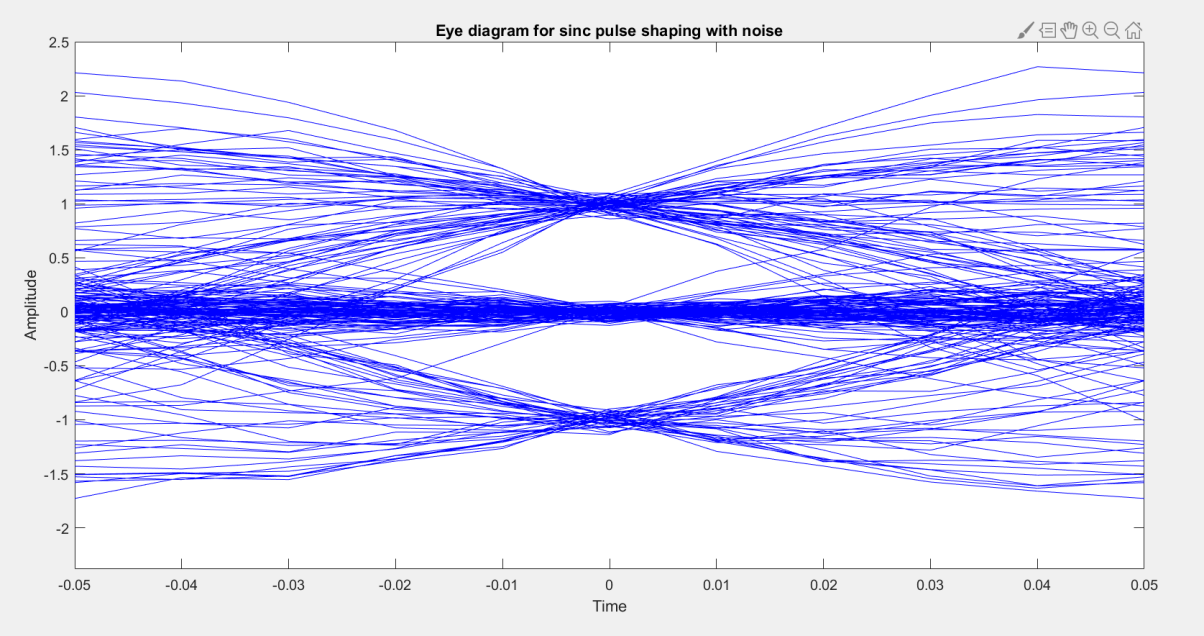
Robustness of the system w.r.t.	Noise	Sampling time	Synchronization Errors
Sinc Pulse	Eye opening is small and no error free sampling region. Hence robustness is **Low.**	T = 0 point in the eye diagram. No sampling time offset	**Low** The eye diagram has sharp edges. Very high timing jitter
Raised cosine signal roll-off = 0.5	Eye opening is wide and minimum Inter Symbol Interference sampling region. Hence robustness is **Medium.**	T=0 point in the eye diagram. (At maximum eye opening)	**Medium** Thinner edges in eye diagram than the Sinc pulse.
Raised cosine signal roll-off = 1	Eye opening is wider and error free sampling region. Hence robustness is **High.**	T=-0.025 or +0.025 point in the eye. diagram. (At maximum eye opening)	**High** Sharp edges in the eye diagram Very low timing jitter because of negligible ambiguity at level crossings.

2.2. Task 2

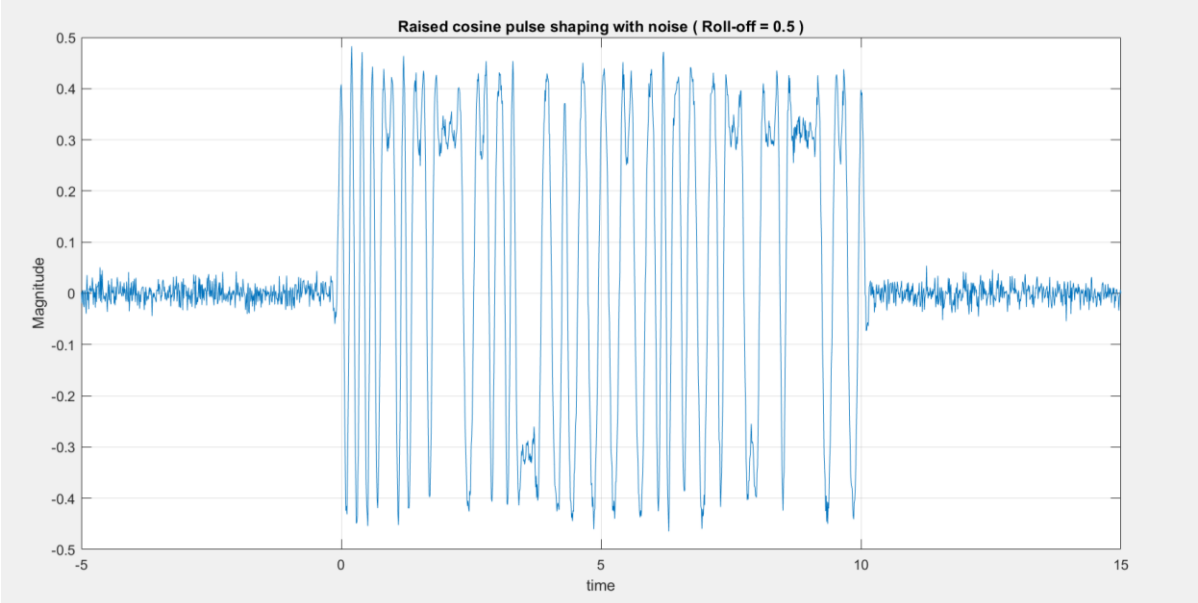
In Task 2, we redid Task 1 but with Additive White Gaussian noise (AWGN). We used the ‘’randn’’ function in MATLAB to create the noise. We adjusted the noise variance so that the signal-to-noise ratio was 10 dB, where the signal-to-noise ratio compares the signal energy per bit to the noise power spectral density. We calculated the signal energy per bit by dividing the total signal power by the symbol rate or bit rate. (In a 2-PAM system, the symbol rate equals the bit rate.)



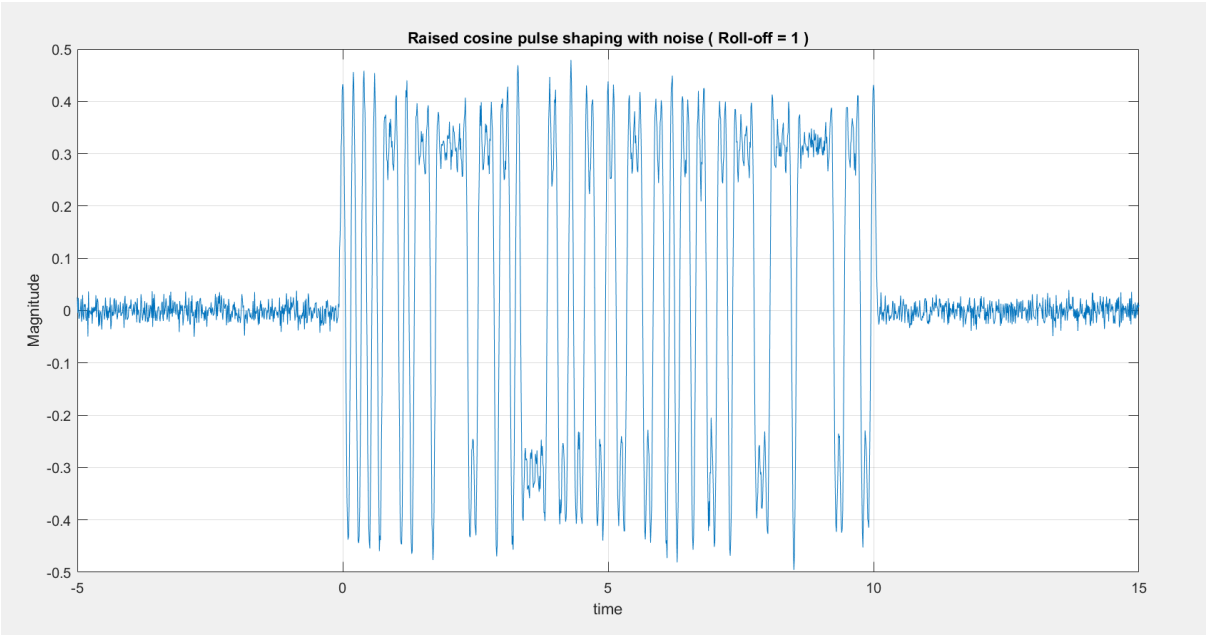
Eye diagram after adding noise,



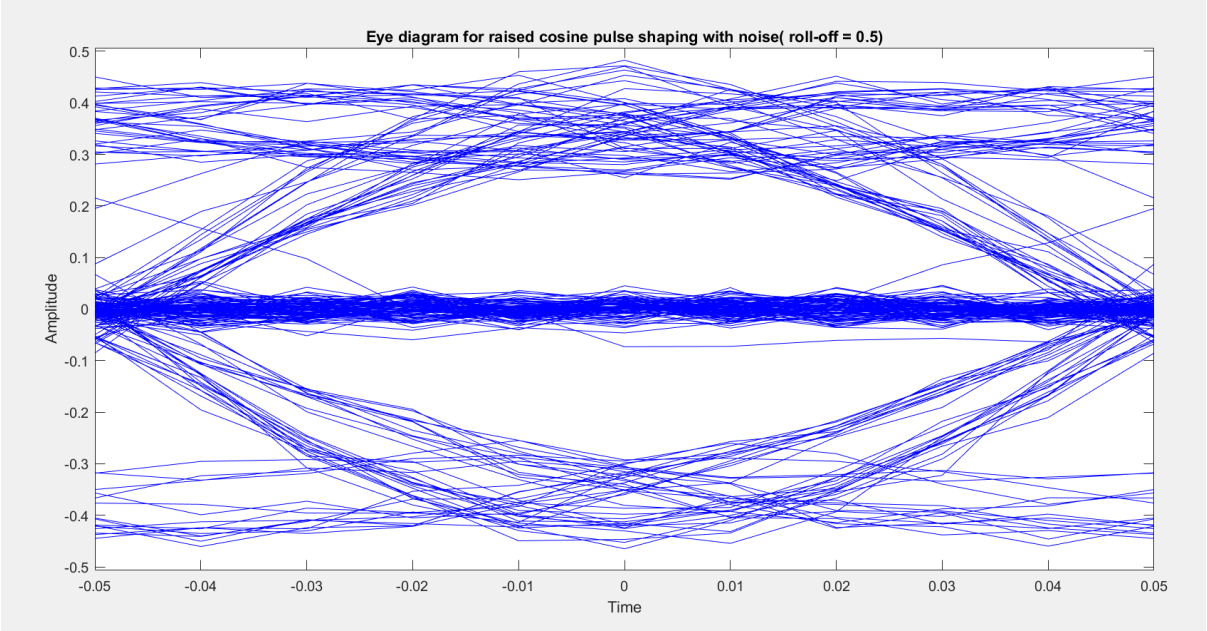
Repeat raised cosine pulses having roll-off factor = 0.5 after adding AWGN.



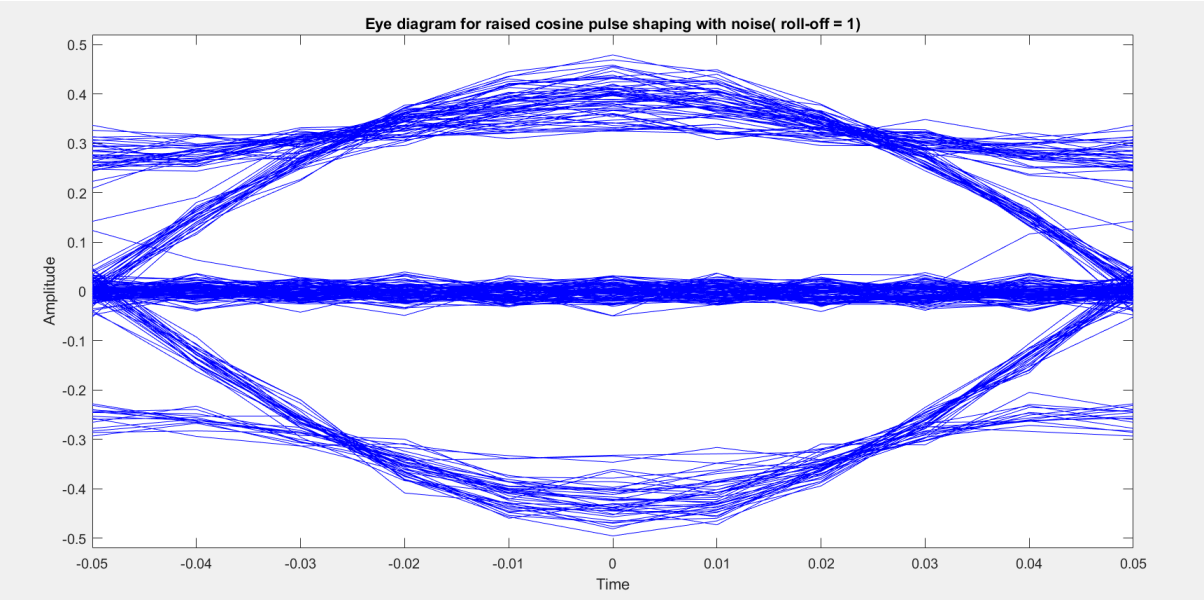
Repeat raised cosine pulses having roll-off factor = 1 after adding AWGN.



Eye diagram for raised cosine pulses having roll-off factor = 0.5 after adding AWGN.



Eye diagram for raised cosine pulses having roll-off factor = 1 after adding AWGN.



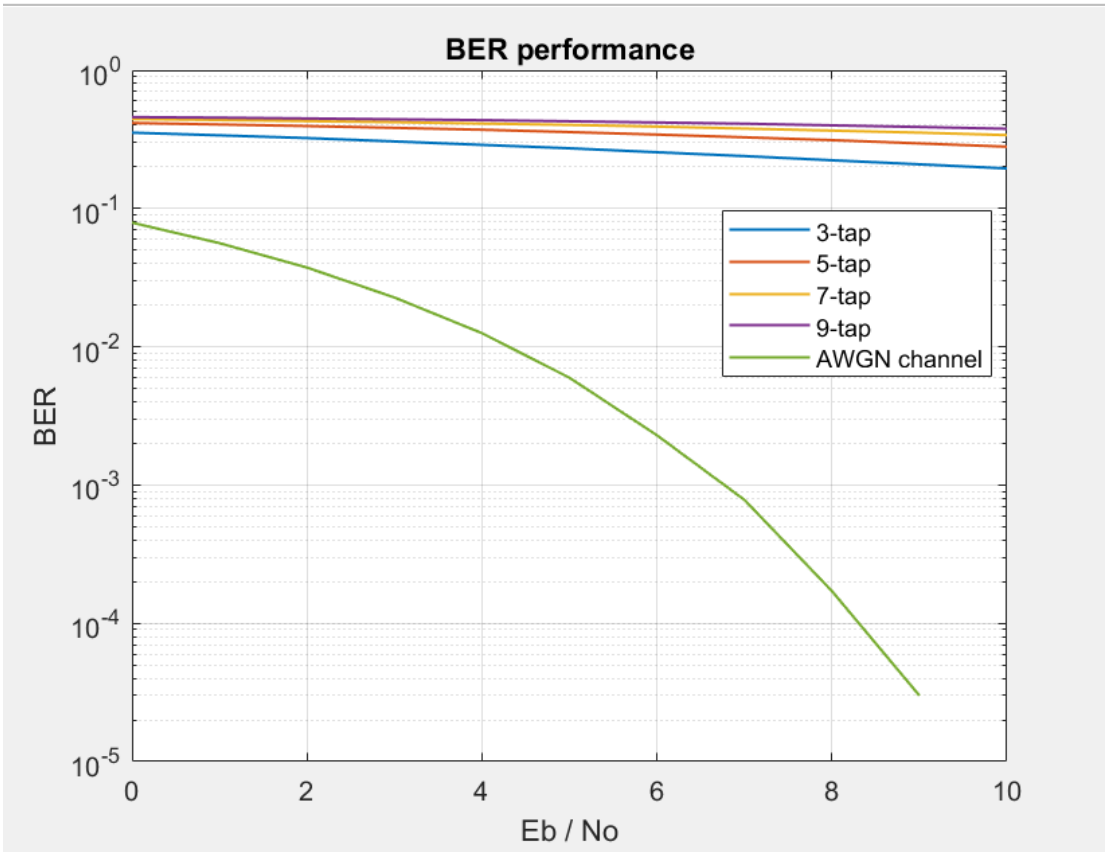
After adding noise, the eye openings for all three signals got reduced and the edges became thicker. However, the raised cosine pulse-shaped signal with a roll-off of 1 noise immunity is very high due to wide eye opening and synchronization and jitters are very small when compared to other two signals.

2.3. Task 3

In this task, our goal is to reduce the impact of a multipath channel on a transmitted signal by using a ZF equalizer. First, we created a random binary sequence to send as the input signal. Then, we used Pulse Amplitude Modulation (2-PAM) to modulate the binary sequence, with "0" as -1 and "1" as +1. We assume that we are transmitting impulses. To simulate the multipath channel, we convolved the transmitted symbols with a 3-tap impulse response, represented by h = [0.3 0.7 0.4]. This causes inter-symbol interference (ISI) because of delayed and attenuated replicas of the signal. The multipath channel generates a signal by shifting and scaling the original signal.

Then, we added Additive White Gaussian noise to the received signal to simulate real-world communication conditions. The noise level was adjusted to achieve an Eb/No (average bit energy to noise power spectral density ratio) of 0 dB. At the receiver, we created ZF equalization filters with varying tap lengths (3, 5, 7, and 9 taps) to counteract the effects of ISI. Next, we demodulated the equalized signal and converted the symbols back into bits to compare with the original transmitted sequence. We calculated the Bit Error Rate for each tap setting by counting the number of bit errors between the received and original transmitted sequences.

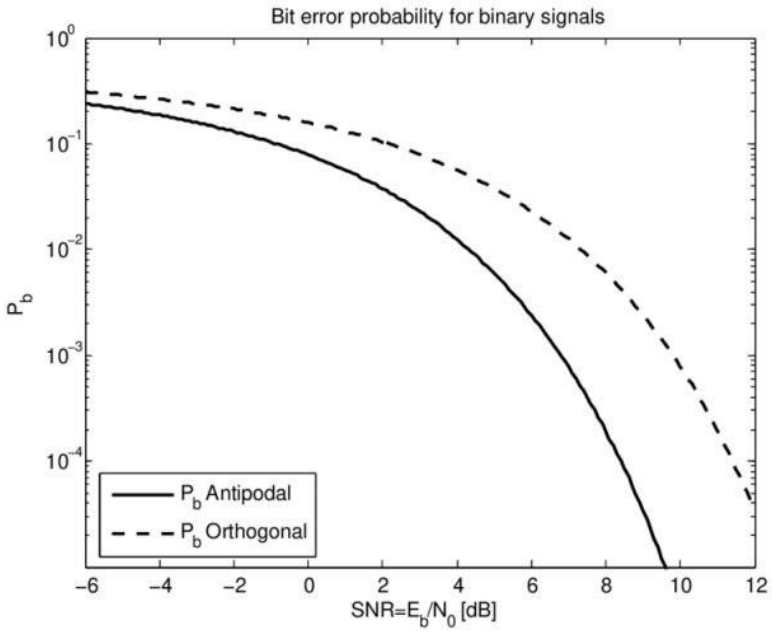
Next, we repeated Steps 1-7 for different Eb/No values ranging from 0 to -10 dB to analyse the performance of the system under various noise levels. Then, we created a plot showing the BER results for different tap settings and Eb/No values together, offering a comparative analysis of the performance of the system across various scenarios. Additionally, we included BER results for an AWGN channel in the same plot to compare the performance of the ZF equalized multipath channel with an ideal channel containing only AWGN noise.



In an AWGN channel, the bit error rate is reduced compared to an ISI multipath channel with a Zero Forcing equalizer at the receiver. This is because the Zero Forcing equalizer behaves like an amplifier for noise in certain frequency ranges. This observation can be derived from the frequency responses of both the channel and the equalizers. In the passband, the equalizer amplifies noise, leading to a reduction in the system's BER. Additionally, imperfect channel estimation can result in a higher bit error rate. Another factor is that Zero Forcing equalizers do not force the pulse to be zero at all sampling instances, only at selected finite instances (Tap number), contributing to residual interference and differences in performance.

Binary orthogonal signaling

In binary orthogonal signaling, distances between symbols are smaller than BPSK. Therefore, noises will cause to bit errors rather than BPSK modulation. So BER will be increased.



3. References

[1] Proakis, John G., and Masoud Salehi. Digital Communications. McGraw-Hill Education, 2008.

[2] Stefan Host. A short introduction to digital communications. 06 2023

4. Appendices

4.1. MATLAB code - Task 1 & 2

```
%% Task 1 & 2

%% Impulse train representing BPSK symbols


bit_rate           = 10;
bit_duration       = 1/bit_rate;
sampling_frequency = bit_duration/10;

endtime = 10;
time    = 0:sampling_frequency:endtime;

BPSK_symbols = zeros(size(time));

% BPSK modulation
for i=1:bit_duration/sampling_frequency:numel(BPSK_symbols)
    BPSK_symbols(i) = 2*randi([0 1])-1;
end

Eb_No_dB = 10 ; % 10dB
Eb_No = 10^ (Eb_No_dB/10) ; % converting SNR per bit into linear range

figure;
stem(time,BPSK_symbols);
hold on;


```

```

%% Sinc pulse shaping filter

timefor_sinc = -endtime:sampling_frequency:endtime;

sinc_filter          = sinc(timefor_sinc/bit_duration);
transmit_signal_sinc = conv(sinc_filter,BPSK_symbols,"same") ;

Eb_transmit_signal_sinc = bandpower(transmit_signal_sinc)/bit_rate ; % Eb = P (signal power )/R ( symbol or bit rate )
No_transmit_signal_sinc = Eb_transmit_signal_sinc / Eb_No ; % power spectraldensity of the transmit signal
sigma_transmit_signal_sinc = sqrt(No_transmit_signal_sinc/2); % noise variance

plot(timefor_sinc+endtime/2,transmit_signal_sinc);
grid on;
title("Sinc pulse shaping without noise");
xlabel("time");
ylabel("Magnitude");
legend("BPSK symbols","Transmit signal");

% Generating noise

sinc_with_noise                = sigma_transmit_signal_sinc * randn(1,numel(transmit_signal_sinc));
transmit_sinc_signal_with_noise = transmit_signal_sinc+sinc_with_noise ;


% plotting the signal after adding noise
figure
plot(timefor_sinc+endtime/2 , transmit_sinc_signal_with_noise);
title("Sinc pulse shaping with noise");
xlabel("time");
ylabel("Magnitude");
grid on;

```

```
%% Eye diagrams for sinc pulse shaping

% plotting the eye diagram without noise
eyediagram(transmit_signal_sinc,bit_duration/sampling_frequency,bit_duration);
title("Eye diagram for sinc pulse shaping without noise");

% plotting the eye diagram with noise
eyediagram(transmit_sinc_signal_with_noise,bit_duration/sampling_frequency,bit_duration);
title("Eye diagram for sinc pulse shaping with noise");


```

```
%% Raised cosine filters

% Roll off factor = 0.5
r1 = 0.5 ;
raised_cosine_1 = rcosdesign(r1,(numel(timefor_sinc)-1)/(bit_duration/sampling_frequency),bit_duration/sampling_frequency);

transmit_signal_rcosine1=conv(raised_cosine_1,BPSK_symbols,"same");

figure
stem(time,BPSK_symbols);
hold on;
plot(timefor_sinc+endtime/2,transmit_signal_rcosine1);
legend("BPSK symbols" , "Transmit signal");
title("Raised cosine pulse shaping without noise ( Roll-off = 0.5 )");
xlabel("time");
ylabel("Magnitude");
grid on;

% Generating noise
Eb_transmit_signal_rcosine_1    = bandpower(transmit_signal_rcosine1)/bit_rate ;
No_transmit_signal_rcosine_1    = Eb_transmit_signal_rcosine_1 / Eb_No ;
sigma_transmit_signal_rcosine_1 = sqrt(No_transmit_signal_rcosine_1/2);



% plotting the signal after adding noise ;
figure
plot(timefor_sinc+endtime/2 , transmit_rcosine1_signal_afternoise);
title("Raised cosine pulse shaping with noise ( Roll-off = 0.5 )");
xlabel("time");
ylabel("Magnitude");
grid on;

eyediagram(transmit_signal_rcosine1,bit_duration/sampling_frequency,bit_duration);
title("Eye diagram for raised cosine pulse shaping without noise( roll-off = 0.5)");

eyediagram(transmit_rcosine1_signal_afternoise,bit_duration/sampling_frequency,bit_duration);
title("Eye diagram for raised cosine pulse shaping with noise( roll-off = 0.5)");



% Roll off factor = 1
r2 = 1;
raised_cosine_2 = rcosdesign(r2,(numel(timefor_sinc)-1)/(bit_duration/sampling_frequency),bit_duration/sampling_frequency);

transmit_signal_rcosine_2=conv(raised_cosine_2,BPSK_symbols,"same");

figure
stem(time,BPSK_symbols);
hold on;
plot(timefor_sinc+endtime/2,transmit_signal_rcosine_2);
legend("BPSK symbols" , "Transmit signal");
title("Raised cosine pulse shaping without noise ( Roll-off = 1 )");
xlabel("time");
ylabel("Magnitude");
grid on;


% Generating noise
Eb_transmit_signal_rcosine_2 = bandpower(transmit_signal_rcosine_2)/bit_rate ;
No_transmit_signal_rcosine_2 = Eb_transmit_signal_rcosine_2 / Eb_No ;
sigma_transmit_signal_rcosine_2 = sqrt(No_transmit_signal_rcosine_2/2);

noise_for_rcosine_2 = sigma_transmit_signal_rcosine_2 * randn(1,numel(transmit_signal_rcosine_2));
transmit_rcosine_2_signal_afternoise = transmit_signal_rcosine_2+noise_for_rcosine_2 ;

% plotting the signal after adding noise ;

figure
plot(timefor_sinc+endtime/2 , transmit_rcosine_2_signal_afternoise);
title("Raised cosine pulse shaping with noise ( Roll-off = 1 )");
xlabel("time");
ylabel("Magnitude");
grid on;

eyediagram(transmit_signal_rcosine_2,bit_duration/sampling_frequency,bit_duration);
title("Eye diagram for raised cosine pulse shaping without noise( roll-off = 1)");

eyediagram(transmit_rcosine_2_signal_afternoise,bit_duration/sampling_frequency,bit_duration);
title("Eye diagram for raised cosine pulse shaping with noise( roll-off = 1)");
```

4.2. MATLAB code - Task 3

```
%% Task 3

bit_rate =10 ;
bit_duration = 1/bit_rate ;

time = 0:bit_duration:100000;
data_bits = randi([0 1],1,numel(time));

PAM2_data = real(pskmod(data_bits,2)); % 2-PAM modulation bit 1 -> +1 and bit 0 ->


% 3-tap multipath channel with impulse response h
h = [0.3 0.7 0.4] ;
rt = conv(PAM2_data,h,"same");

%Bit energy
Eb = sum(PAM2_data.^2)/length(PAM2_data) ;

% creating Zero forcing equalizer (M - tap equalizer)... h(t)* E(t)=delta(t) ,where h(t)is the  channel impulse response and E(t) is the equalizer
Eb_No_dB = 0:10 ; % dB

for M=3:2:9
    N = (M-1)/2 ;
    Po = zeros(1,M);
    Po(N+1) = 1;
    Pr = toeplitz([h(2) h(1) zeros(1,M-2)],[h(2) h(3) zeros(1,M-2)]) ; % Toeplitz matrix with filter coeffients
    C = Pr\Po' ;

    BER_ISI = zeros(1,11);

    for n=1:numel(Eb_No_dB)

        Eb_No = 10^(Eb_No_dB(n)/10);
        No    = Eb/Eb_No ;
        sigma = sqrt(No/2);

        %signal after adding noise
        rtn_ISI = rt + sigma*randn(1,numel(rt));

        rtn_e                 = conv(rtn_ISI,C,"same"); % Signal after the equalizer
        received_data_bits_ISI = real(pskdemod(rtn_e,2));

        bit_errors_ISI=numel(find(received_data_bits_ISI-data_bits)); % calculating number of bit errors

        BER_ISI(n)=bit_errors_ISI/numel(data_bits); % calculating bit error rate


    end

    semilogy(Eb_No_dB,BER_ISI,'linewidth',1);
    grid on;
    hold on;


end

% plotting BER for a AWGN channel in the same figure
BER_AWGN= zeros(1,11);

for n=1:numel(Eb_No_dB)

    Eb_No = 10^(Eb_No_dB(n)/10);
    No = Eb/Eb_No ;
    sigma = sqrt(No/2);

    %signal after adding noise
    rtn_awgn =PAM2_data + sigma*randn(1,numel(PAM2_data));
    received_data_bits_awgn = real(pskdemod(rtn_awgn,2));

    bit_errors_AWGN=numel(find(received_data_bits_awgn-data_bits)); % calculating number of bit errors

    BER_AWGN(n)=bit_errors_AWGN/numel(data_bits); % calculating bit error rate


end

semilogy(Eb_No_dB,BER_AWGN,'linewidth',1);
legend("3-tap","5-tap","7-tap","9-tap","AWGN channel");
title("BER performance");
xlabel("Eb / No");
ylabel("BER");
```