16 - QAM Commmunication Model

Priya Mandot
2022102053
ECE

International Institute Of Information Technology, Hyderabad
Hyderabad, India

priya.mandot@students.iiit.ac.in

Aniruth Suresh
2022102055
ECE

International Institute Of Information Technology, Hyderabad
Hyderabad, India

aniruth.suresh@students.iiit.ac.in

***Abstract*—In this project, we develop and simulate a communication model utilizing 16-QAM (Quadrature Amplitude Modulation). 16-QAM is a modulation scheme widely used in digital communication systems for transmitting digital data over radio and optical communication channels. This project involves designing a communication system that encodes digital information into 16-QAM symbols, transmits them through a simulated channel, and decodes them back to the original data. Through this simulation, we aim to understand the intricacies of 16-QAM modulation, channel effects, and decoding algorithms, while showcasing our proficiency in applying theoretical concepts to practical communication scenarios.**

I. Introduction

Quadrature Amplitude Modulation (QAM) is a modulation scheme widely used in digital communication systems for transmitting digital data over radio and optical communication channels. In QAM, two carriers are used, each having the **same frequency** but are **90° out of phase** with each other, hence the term **quadrature**.

16QAM, in particular, is a modulation scheme that utilizes **16 different amplitude and phase** combinations to encode **four bits per symbol**. It is commonly employed in applications where a balance between data rate and spectral efficiency is required. Each symbol in 16QAM represents a unique combination of both amplitude and phase, allowing for efficient transmission of digital data.

The implementation of a 16QAM communication model involves several key blocks, including symbol mapping (encoder), pulse shaping (line coding), modulation, channel effect, demodulation, line decoding, and decoder. Figure 1 shows the description of the different blocks needed to be implemented to establish a full streamed communication model .

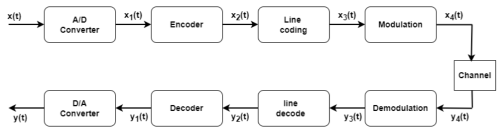


Fig. 1. Complete System Model

II. Procedure- At the Transmitter

In this section, we will delve into the detailed implementation of the 16-QAM modulation block. We will cover each step from the input data processing to the demodulation of the obtained signal after channel effects (AWGN) .

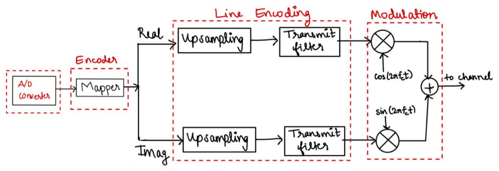


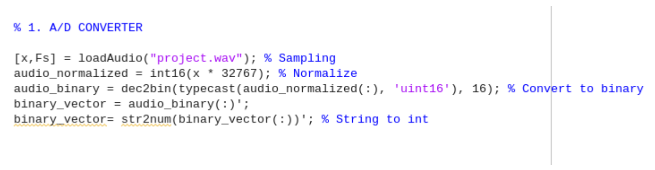
Fig. 2. Transmitter for QAM

A. Analog-to-Digital Converter (A/D Converter)

An Analog-to-Digital Converter (A/D Converter) is a crucial component in digital signal processing systems that transforms continuous-time analog signals into discrete-time digital signals. This conversion process involves sampling the analog signal at regular intervals and quantizing each sample to a digital value. The A/D Converter typically consists of three main stages:

1) **Sampling:** The analog signal is sampled at a specified rate known as the sampling frequency (f_s). The sampling frequency determines how often the analog signal is measured and converted to digital form.
2) **Quantization:** Each sampled value is quantized into a digital representation with a finite number of bits. The quantization process involves dividing the range of possible analog values into discrete levels and assigning each sampled value to the closest level. The number of bits used for quantization determines the resolution of the digital signal.
3) **Encoding:** The quantized digital values are encoded into binary form for further processing or storage. This binary representation allows the digital signal to be manipulated and transmitted using digital processing techniques.

The accuracy and fidelity of the digital signal obtained from the A/D Converter depend on factors such as the sampling rate, resolution, and quantization error. **A higher sampling rate** and resolution result in a **more accurate representation of the original analog signal** but may require **more computational resources and storage space**.



```
% 1. A/D CONVERTER

[x,Fs] = loadAudio("project.wav"); % Sampling
audio_normalized = int16(x * 32767); % Normalize
audio_binary = dec2bin(typecast(audio_normalized(:), 'uint16'), 16); % Convert to binary
binary_vector = audio_binary(:)';
binary_vector= str2num(binary_vector(:))'; % String to int
```

Fig. 3. A/D converter

In digital communication systems, the A/D Converter plays a crucial role in converting analog audio signals, sensor readings, or other analog data into digital form for processing, transmission, and storage.

The digital audio data extracted from a waveform file is readied for processing through an analog-to-digital converter (ADC) in several sequential steps. First, the audio data undergoes normalization to ensure it fits within the range of a 16-bit signed integer, effectively scaling the values by 32767. Subsequently, the normalized data is converted into binary representation. This conversion involves typecasting each sample to a 16-bit unsigned integer and then converting it into its binary format. Finally, the binary data is structured into a vector format for streamlined processing in subsequent stages of the conversion process.

Shown below is the output of the A/D converter for first 1000 samples of the given audio signal "project.wav" . We can observe that the analog signal audio is now completely mapped to digital i.e, 0 and 1's .

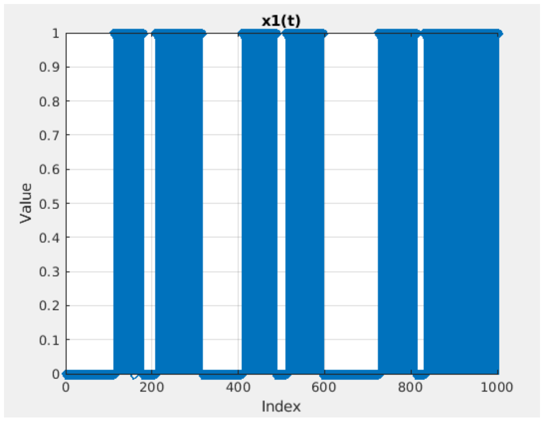


Fig. 4. Output of the A/D converter

B. *Encoding - Gray coding*

Gray Code Mapping is a type of encoding scheme used in digital communication systems to map binary data to symbols with minimal transition errors. In Gray Code Mapping, adjacent symbols differ by **only one bit, reducing the likelihood of errors caused by bit transitions.**

Gray Code Mapping is commonly used in modulation schemes such as Quadrature Amplitude Modulation (QAM), where it helps improve the robustness and reliability of communication systems in noisy environments. By reducing the likelihood of bit errors, Gray Code Mapping contributes to the overall performance and efficiency of digital communication systems.

Given below is an illustration of Gray coding in case of 16-QAM :

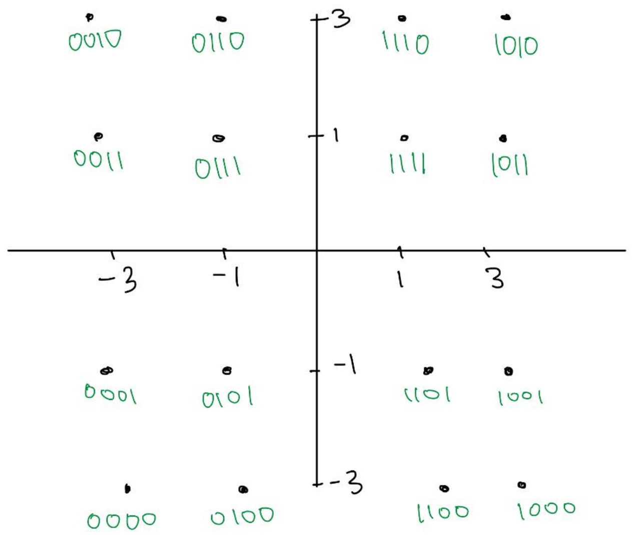


Fig. 5. Gray Coding for 16-QAM

*1) **Power and BER calculation**:* M-ary QAM typically consists of two $\sqrt{M}$-PAM signals, in-phase and quadrature, one for each of the two orthogonal carriers $\phi_1(t) = \frac{\sqrt{2}}{\sqrt{T_M}}\cos(\omega_c t)$ and $\phi_2(t) = \frac{\sqrt{2}}{\sqrt{T_M}}\cos(\omega_c t)$. Specifically for rectangular M-ary QAM, the transmitted signal is represented by

$$s_i(t) = a_i\frac{\sqrt{2}}{\sqrt{T_M}}\cos(\omega_c t) + b_i\frac{\sqrt{2}}{\sqrt{T_M}}\sin(\omega_c t), \quad 0 \le t \le T_M$$

where

$$a_i = \pm\frac{d}{2}, \pm\frac{3d}{2}, \dots \pm \left(\sqrt{M}-1\right)\frac{d}{2}$$

$$b_i = \pm\frac{d}{2}, \pm\frac{3d}{2}, \dots \pm \left(\sqrt{M}-1\right)\frac{d}{2}$$

It is easy to observe that the QAM signal space is two-dimensional with basis functions $\phi_1(t)$ and $\phi_2(t)$.
We assume all signals to be **equiprobable** in an AWGN channel. The first quadrant of the signal space is reproduced in Fig. 7. Because all the signals are equiprobable, the **decision region boundaries will be perpendicular bisectors joining various signals** which are indicated by the dotted lines .

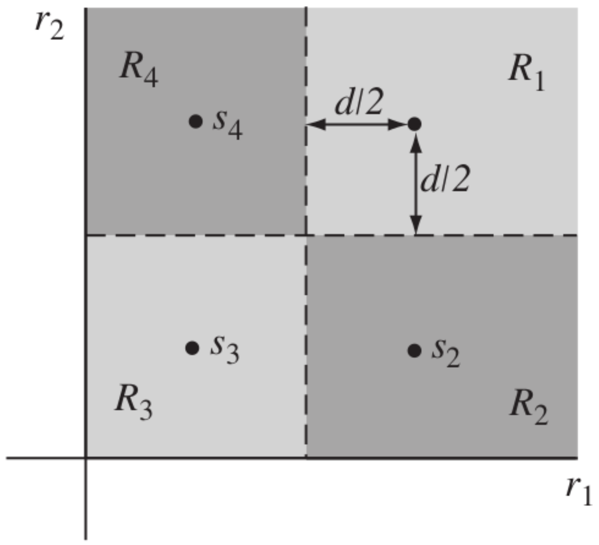


Fig. 6. Quadrants in 16-QAM

From Fig. 7, it's clear that the noise amplitude has to be greater than $-\frac{d}{2}$, so that the symbols don't switch regions and stay in their designated regions and resulting in a successful detection.

$$\begin{aligned}
P(C|m_1) &= P(\text{noise vector originating at } s_1 \text{ lies within } R_1) \\
&= P(n_1 > -\frac{d}{2}, n_2 > -\frac{d}{2}) \\
&= P(n_1 > -\frac{d}{2}) \cdot P(n_2 > -\frac{d}{2}) \\
&= [1 - Q\left(\frac{d}{2\sigma_n}\right)]^2 \\
&= \left[1 - Q\left(\frac{d}{\sqrt{2N}}\right)\right]^2
\end{aligned}$$

For convenience, let us define $d_p = 1 - Q\left(\frac{d}{\sqrt{2N}}\right)$.
Hence,

$$P(C|m_1) = p^2$$

Using similar arguments, we have

$$\begin{aligned}
P(C|m_2) = P(C|m_4) = (1-Q\left(\frac{d}{\sqrt{2N}}\right))(1-2Q\left(\frac{d}{\sqrt{2N}}\right)) \\
= p(2p-1)
\end{aligned}$$

$$P(C|m_3) = (2p-1)^2$$

Because of the symmetry of the signals in all four quadrants, we get similar probabilities for the four signals in each quadrant. Hence, the probability of correct decision is

$$= \left[4p^2 + 4p(2p-1) + 4p(2p-1) + 4(2p-1)^2\right]\frac{1}{16}$$

$$= \frac{(3p-1)^2}{4}$$

and

$$P_{eM} = 1 - P(C) = \frac{9}{4}\left(p + \frac{1}{3}\right)(1-p)$$

In practice,$P_{eM} \to 0$ if SNR is high and, hence, $P(C) \to 1$. This means $p \approx 1$ and $p + \frac{1}{3} \approx 1\frac{1}{3}$

$$P_{eM} = 3(1-p) = 3Q\left(\frac{d}{\sqrt{2N}}\right)$$

To express this in terms of the **received power** S_i, we determine E, the average energy of the signal set . Because E_k, the energy of s_k, is the square of the distance of s_k from the origin,

$$E_1 = \frac{9d^2}{2} + \frac{9d^2}{2} = \frac{9d^2}{2}$$

$$E_2 = \frac{3d^2}{2} + \frac{9d^2}{2} = \frac{5d^2}{2}$$

Similarly,

$$E_3 = \frac{d^2}{2}$$

$$E_4 = \frac{5d^2}{2}$$

Hence, the average symbol energy is

$$E = \frac{1}{4}\left(\frac{9d^2}{2} + \frac{5d^2}{2} + \frac{5d^2}{2} + \frac{d^2}{2}\right) = \frac{d^2}{2}$$

and $d^2 = 0.4E$. Moreover, for $M = 16$, each symbol carries the information of $\log_2 16 = 4$ bits.
Hence, **The average energy per bit E_b is**

$$E_b = \frac{E}{4}$$

and

$$E_b = \frac{E}{4N} = \frac{5d^2}{8N}$$

Hence, for large $\frac{E_b}{N}$,

$$P_{eM} = 3Q(\sqrt{\frac{d}{2N}})$$

$$= 3Q(\sqrt{\frac{4E_b}{5N}})$$

where P_{eM} is the symbol error rate, which is slightly larger than the BER .

*2) **Plots and Observation**:* To understand Gray coding, let's consider a random set of 48 integers and then encode them using Gray code.

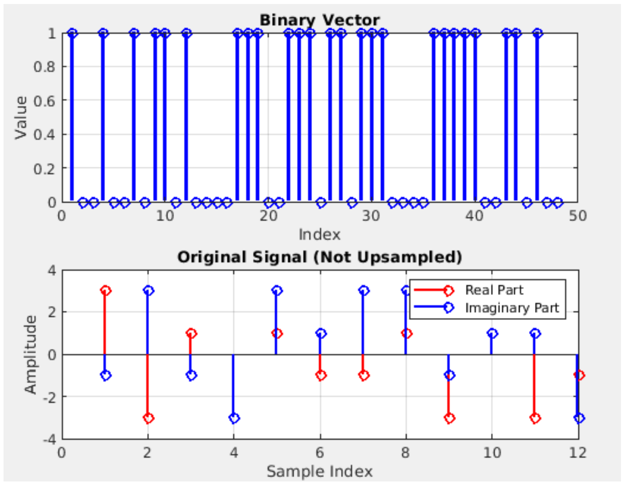


Fig. 7. Gray Coding

Clearly from Figure 6, we can see that a set of 4 bits are getting mapped to a symbol according to the Gray coding scheme mentioned in Figure 5.

C. Line Coding

Line coding is a crucial process in digital communication systems that converts a sequence of binary symbols into a continuous-time signal suitable for transmission over a communication channel.

In this subsection, we will generate the output of the block for the line coding scheme according to the equation:

$$x_3(t) = \sum_k a_k p(t - kT_b)$$

where $p(t)$ is a pulse of duration T_b. This equation represents the line coding process, where binary symbols a_k are multiplied by corresponding pulses $p(t - kT_b)$ and summed up to generate the output signal $x_3(t)$.

We will implement the line coding for two different pulse shapes:

1) **Raised cosine pulse:** This pulse shape has a smooth transition from zero to its maximum amplitude and back to zero within its duration T_b. It is characterized by its roll-off factor, which determines the rate of transition at its edges.
2) **Rectangular pulse:** This pulse shape has a constant amplitude over its duration T_b and abruptly transitions to zero outside this interval.

*1) **Raised Cosine Pulse**:* Let's consider the case of a Raised Cosine pulse with a **roll-off factor of 1**, adhering to the Nyquist first criterion. In this scenario, we are utilizing a full Raised Cosine pulse, and the **oversampling factor is 10**.

It's important to upsample the symbols before passing them through the filter, as they need to match with the filter characteristics. Upsampling increases the number of samples per symbol, ensuring that the symbols are adequately represented and aligned with the filter's response.

Shown below is the subplot of both the mapped symbols and the upsampled version of them with upsampling factor = 10

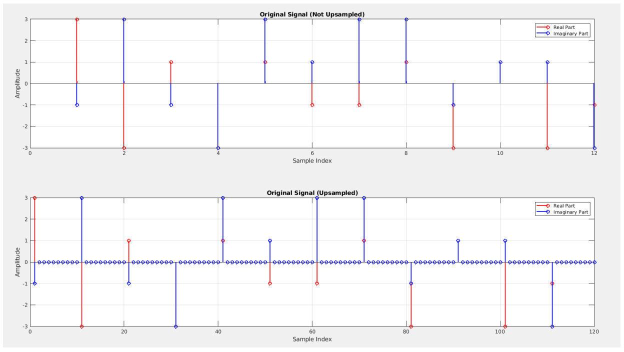


Fig. 8. Upsampling before transmit filter

Consider a raised cosine pulse with a roll-off factor of 1 and an oversampling factor of 10, depicted in the graph below:

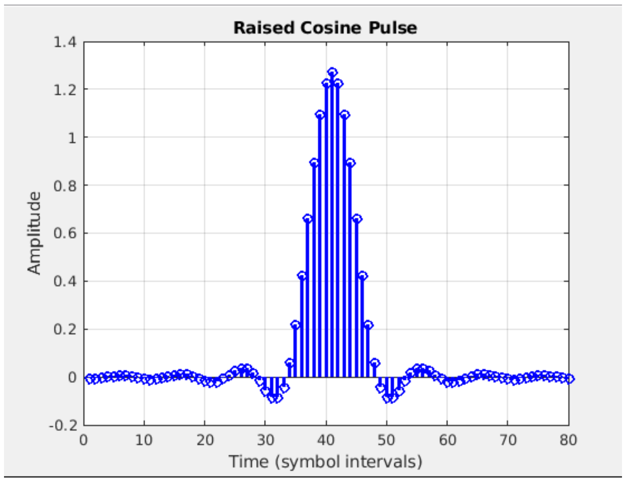


Fig. 9. Raised Cosine Pulse

After the symbols are upsampled , we can pass them through the transmit filter for line encoding and the outputs are shown below .

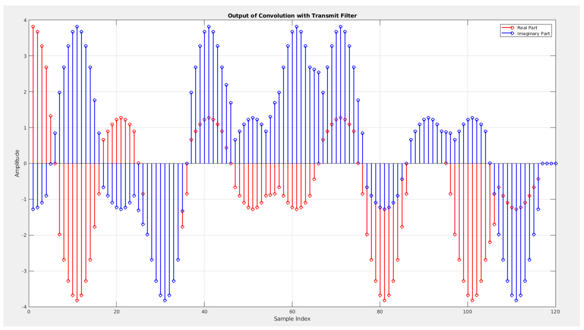


Fig. 10. Output of transmit filter

The **Power Spectral Density (PSD)** of a signal is a measure of how the power of the signal is distributed across different frequencies. It gives information about the power content of the signal at different frequency components. The PSD is often represented as a plot with frequency on the x-axis and power/frequency on the y-axis, typically measured in dB/Hz. It provides insights into the signal's frequency characteristics, such as bandwidth, dominant frequency components, and noise levels.
The PSD of the line encoded signal is as follows :

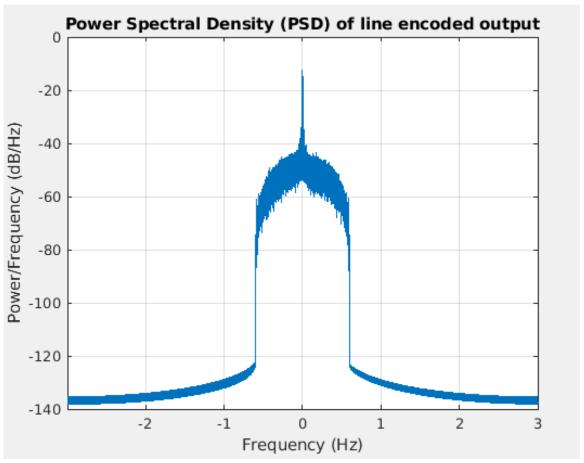


Fig. 11. PSD of line encoded signal

*2) **Rectangular Pulse**:* Consider a rectangular pulse with a width of 16 units. Unlike other pulse shapes such as raised cosine or Gaussian, the rectangular pulse has a simple shape characterized by a constant amplitude within its width and zero amplitude outside.

Rectangular pulses are straightforward to generate and analyze due to their simplicity. However, they exhibit poor spectral properties, leading to increased bandwidth occupancy and susceptibility to **inter-symbol interference (ISI)**.

Consider a rectangular pulse of width 16 , depicted in the graph below:

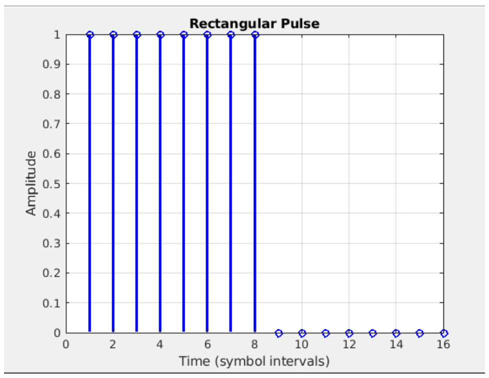


Fig. 12. Rectangular Pulse

After the symbols are upsampled , we can pass them through the transmit filter for line encoding and the outputs are shown below .

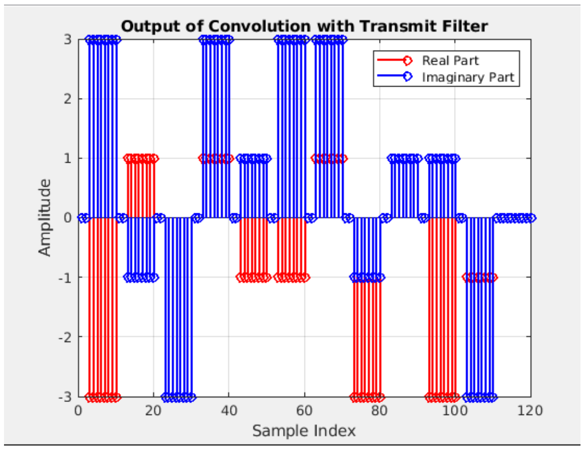
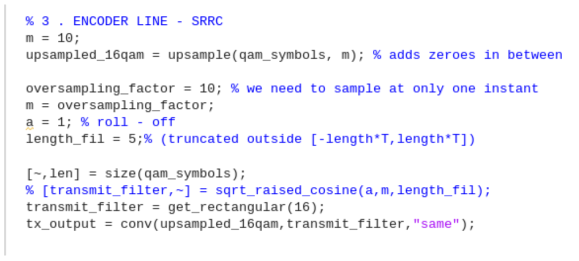


Fig. 13. Output of Transmit Filter - Rectangular Pulse

Code snippet for the line encoding block is given below :



```
% 3 . ENCODER LINE - SRRC
m = 10;
upsampled_16qam = upsample(qam_symbols, m); % adds zeroes in between

oversampling_factor = 10; % we need to sample at only one instant
m = oversampling_factor;
a = 1; % roll - off
length_fil = 5;% (truncated outside [-length*T,length*T])

[~,len] = size(qam_symbols);
% [transmit_filter,~] = sqrt_raised_cosine(a,m,length_fil);
transmit_filter = get_rectangular(16);
tx_output = conv(upsampled_16qam,transmit_filter,"same");
```

Fig. 14. Code snippet for Line encoding

The PSD of the line encoded signal is as follows :

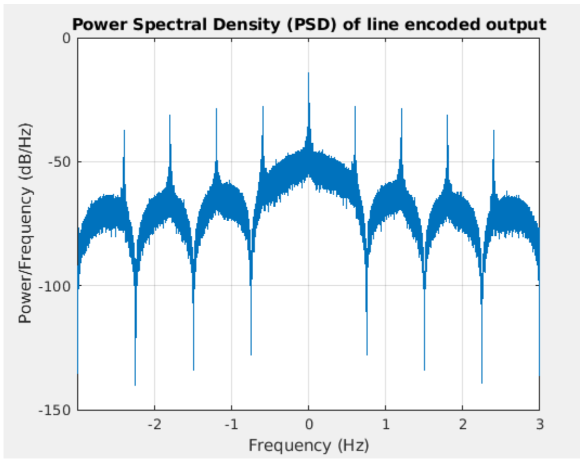


Fig. 15. PSD of line encoded signal

Comparison between the PSD's in both the cases :

CASE 1 : Rectangular pulse

1) The PSD of a Rectangular pulse exhibits **significant spectral leakage** due to its abrupt transitions in the time domain.
2) The Rectangular pulse has a wide bandwidth and high sidelobes in its frequency spectrum.
3) The energy of the Rectangular pulse is spread across a wide range of frequencies, leading to inefficient use of the available bandwidth.

CASE 2 : Raised Cosine pulse

1) The PSD of an SRRC pulse is characterized by a **smoother frequency response** with controlled side-lobes.
2) SRRC pulses are designed to minimize spectral leakage and inter-symbol interference (ISI) by shaping the signal's frequency spectrum.
3) SRRC pulses concentrate most of their energy within the main lobe of the frequency spectrum, leading to improved spectral efficiency and reduced out-of-band interference.

D. Modulation

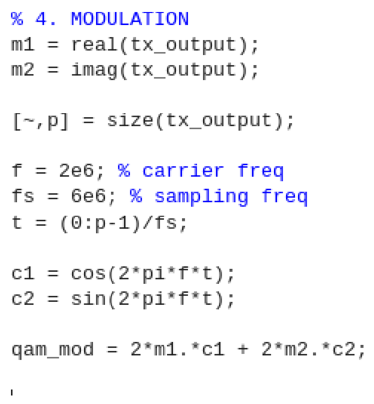
To implement 16-QAM modulation, the real and imaginary parts of each symbol are modulated separately using cosine and sine carriers, respectively. Let I represent the in-phase component of the symbol, and Q represent the quadrature component.

1) **Real Part Modulation**: Multiply the in-phase component (I) of each symbol by $\cos(2\pi f_c t)$, where f_c is the carrier frequency and t represents time.
2) **Imaginary Part Modulation**: Multiply the quadrature component (Q) of each symbol by $\sin(2\pi f_c t)$.

The use of sine and cosine carriers ensures that the in-phase and quadrature components are transmitted in phase quadrature with each other, allowing for the recovery of both amplitude and phase information at the receiver.

Typically, we use a carrier frequency of 2×10^6 Hz (2 MHz) for our communication system. However, for visualization purposes, we have chosen a lower carrier frequency of 100 Hz. This lower frequency allows us to easily plot and visualize the signals without overwhelming the graphical representation.

Code snippet for modulation is as follows :


```
% 4. MODULATION
m1 = real(tx_output);
m2 = imag(tx_output);

[~,p] = size(tx_output);

f = 2e6; % carrier freq
fs = 6e6; % sampling freq
t = (0:p-1)/fs;

c1 = cos(2*pi*f*t);
c2 = sin(2*pi*f*t);

qam_mod = 2*m1.*c1 + 2*m2.*c2;
```

Fig. 16. Code snippet for modulation

*1) **Raised Cosine Pulse**:* The modulated signal, assuming 48 random bits and $f_c = 100$ Hz, is represented in the time domain. This signal is the result of modulating the digital data onto a carrier wave. The modulated signal in this case is as follows :

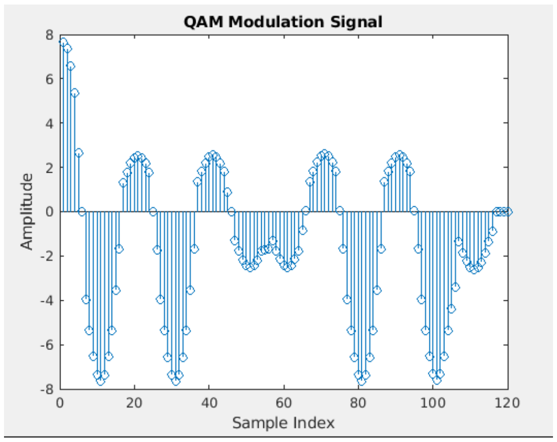


Fig. 17. Modulated Signal - Raised Cosine

The power spectral density (PSD) of the modulated signal is essentially a shifted version of the PSD of the line-encoded signal due to the multiplication by the cosine and sine waves, which results in a frequency shift in the spectrum.

Mathematically, if $X(f)$ represents the PSD of the line-encoded signal, and $Y(f)$ represents the PSD of the modulated signal, where f denotes frequency, then the relationship between them can be expressed as:

$$Y(f) = X(f - f_c) + X(f + f_c)$$

where f_c is the carrier frequency.

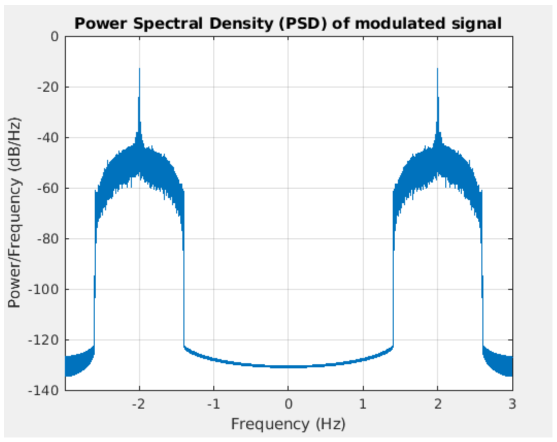


Fig. 18. PSD of Modulated Signal - Raised Cosine

This equation illustrates that the PSD of the modulated signal is obtained by shifting the frequency spectrum of the line-encoded signal by $\pm f_c$ in the frequency domain. This shifting effect is due to the multiplication of the line-encoded signal by cosine and sine waves with frequency f_c, resulting in the translation of spectral components.

*2) **Rectangular Pulse**:* In case of rectangular pulse , we obtain the following modulated signal :

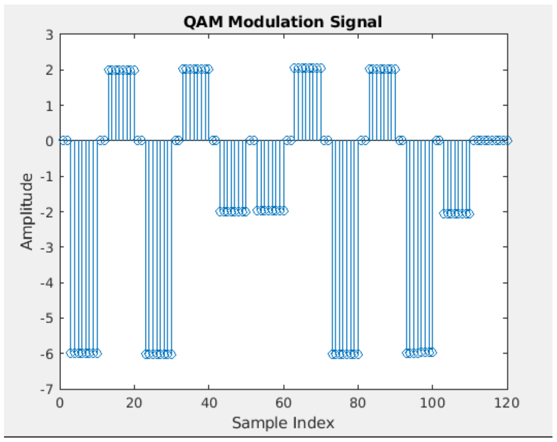


Fig. 19. Modulated Signal - Rectangular Pulse

The power spectral density (PSD) of the modulated signal is essentially a shifted version of the PSD of the line-encoded

signal due to the multiplication by the cosine and sine waves, which results in a frequency shift in the spectrum.

Mathematically, if $X(f)$ represents the PSD of the line-encoded signal, and $Y(f)$ represents the PSD of the modulated signal, where f denotes frequency, then the relationship between them can be expressed as:

$$Y(f) = X(f - f_c) + X(f + f_c)$$

where f_c is the carrier frequency.

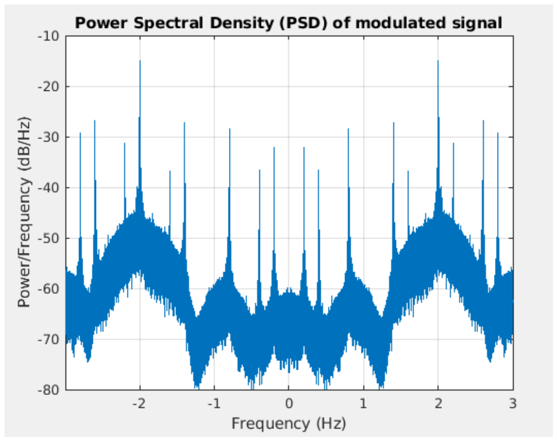


Fig. 20. PSD of Modulated Signal - Rectangular Pulse

III. Channel

After modulating the signal, it is passed through two types of channels, each introducing noise to the transmitted signal:

1) **Memoryless AWGN Channel:**
   In the memoryless AWGN (Additive White Gaussian Noise) channel, the received signal $r(t)$ is given by:

$$r(t) = s(t) + n(t)$$

   where $s(t)$ denotes the transmitted signal and $n(t)$ represents the additive white Gaussian noise. This noise is characterized by its Gaussian probability distribution and uniform power spectral density across all frequencies.

2) **AWGN Channel with Memory:**
   In the AWGN channel with memory, the received signal $r(t)$ is expressed as:

$$r(t) = h(t) * s(t) + n(t)$$

   where $h(t)$ represents the channel impulse response, $*$ denotes convolution, $s(t)$ is the transmitted signal, and $n(t)$ is the additive white Gaussian noise.
   The channel impulse response $h(t)$ accounts for channel memory and is defined as:

$$h(t) = a\delta(t) + (1 - a)\delta(t - bT_b)$$

where $\delta(t)$ is the Dirac delta function, a is the direct path attenuation factor, b is the relative delay of the multipath component in terms of symbol duration T_b.

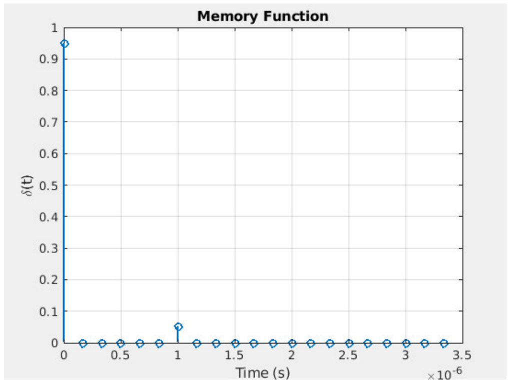


Fig. 21. Channel impulse response for a = 0.95, b = 1, Tb = 1/1e6

Both types of channels introduce noise to the transmitted signal, degrading its quality and impacting the reliability of communication systems.

*1) **Memoryless AWGN Channel:*** In the memoryless AWGN (Additive White Gaussian Noise) channel, noise is directly added to the transmitted signal. This channel model assumes that each transmitted symbol is affected independently by Gaussian noise, without any correlation or memory between symbols.

Approach 1: The direct addition of noise to the transmitted signal can be implemented using the `awgn` function in signal processing software such as MATLAB. This function generates Gaussian noise samples with a specified signal-to-noise ratio (SNR) and adds them to the transmitted signal. The resulting received signal is then passed through the channel without any memory or correlation effects.
Approach 2If the nth transmitted symbol is $b[n]$, the average received energy per symbol is given by:

$$E_s = |b[n]|^2 \cdot \|g_{TX}^* * g_C\|^2$$

Dividing this by the number of bits per symbol yields E_b. The noise variance per dimension is $\sigma^2 = \frac{N_0}{2}$, allowing you to compute E_b/N_0 for your simulation model.
The signal-to-noise ratio (E_b/N_0) is usually expressed in decibels (dB):

$$E_b/N_0(\text{dB}) = 10 \log_{10}\left(\frac{E_b}{N_0}\right)$$

Thus, noise variance can be varied according to different values of E_b/N_0 **CASE 1 : Rectangular pulse**
In this scenario, noise is directly added to the signal that was line encoded using a Rectangular pulse. After the noise is added, the resulting signal appears as follows:

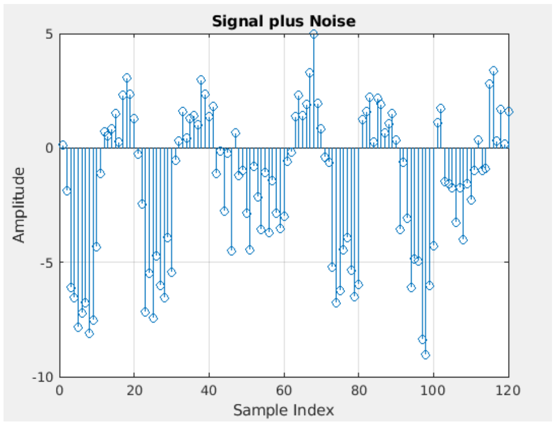


Fig. 22. Noisy- Rectangular Pulse

The noise-affected signal represents the received signal in the communication system, which undergoes further processing for decoding and error correction to recover the original transmitted information.

The **Power Spectral Density (PSD)** of the signal undergoes changes when noise is introduced. This alteration can be observed in the following plot, where variations in the PSD reflect the influence of noise on the signal's frequency characteristics. Such changes provide insights into how noise affects the spectral composition and integrity of the signal, crucial for analyzing its behavior in communication systems.

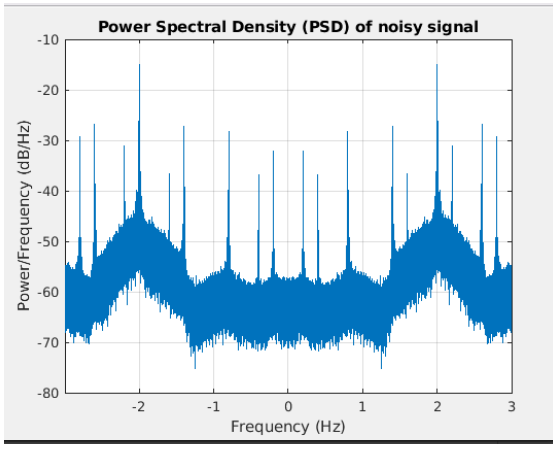


Fig. 23. Noisy PSD- Rectangular Pulse

CASE 2 : Raised Cosine Pulse

In this scenario, noise is directly added to the signal that was line encoded using a Raised Cosine pulse. After the noise is added, the resulting signal appears as follows:

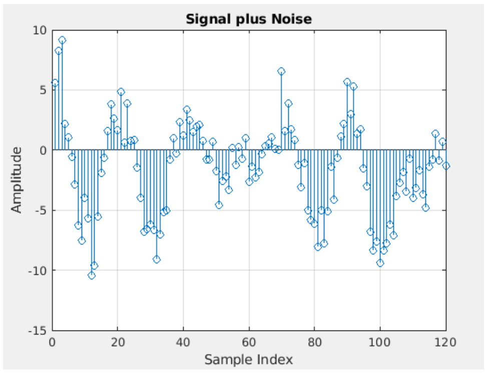


Fig. 24. Noisy- Raised Cosine Modulated Pulse

The **Power Spectral Density (PSD)** of the signal undergoes changes when noise is introduced. This alteration can be observed in the following plot :

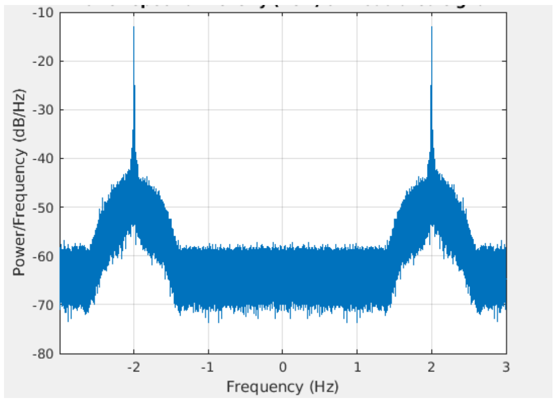


Fig. 25. Noisy PSD- Raised Cosine Modulated Pulse

*2) **AWGN Channel with Memory**:* Although ideally $r(t)$ will be equal to $s(t)$, in practice this will rarely be the case. There are two reasons for this. First, since the channel is never ideal, it will introduce some distortion. For example, channel dispersion which is the result of the nonlinear phase characteristics of the channel. This causes a distortion of the pulse shape, thereby causing neighboring pulses to interfere with each other, resulting in an effect known as intersymbol interference. The second reason is that the received waveform will invariably contain noise. This noise may be introduced by the channel or it may be the result of non-ideal elements in the transmitter and receiver. Assuming a linear dispersive channel, a model for the received sequence $r(t)$ is

$$r(t) = \sum_{k=0}^{\infty} s(t)h(t-k) + n(t)$$

where $h(t)$ is the unit sample response of the channel and $n(t)$ is additive noise.

In order to reduce the error resulting from this, the receiver employs a channel equaliser to reduce the effects of the channel distortion. Since the channel impulse response is not usually known, the equaliser is an adaptive filter which approximated to the inverse of h(t). Here as the channel impulse response was known, an exact inverse was obtained and convoluted with the channel output at the Receiver.

CASE 1 : Rectangular pulse

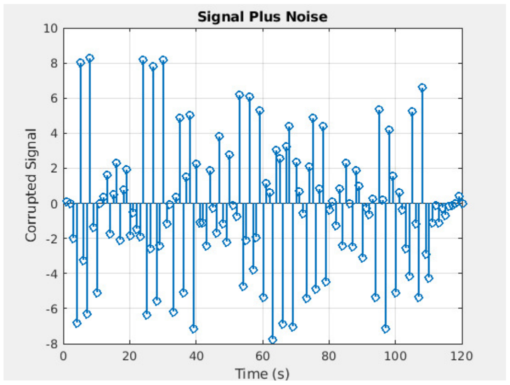


Fig. 26. Noisy Memory channel for rectangular pulse

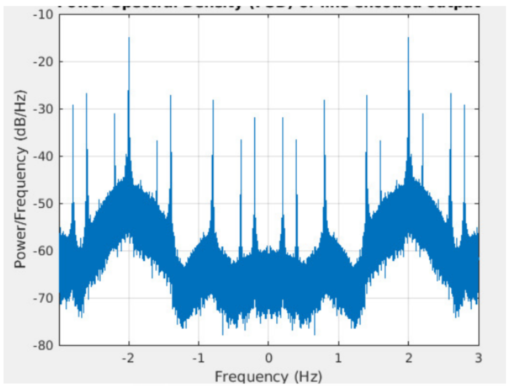


Fig. 27. Noisy Memory channel PSD for Rectangular pulse

CASE 2 : SRRC pulse

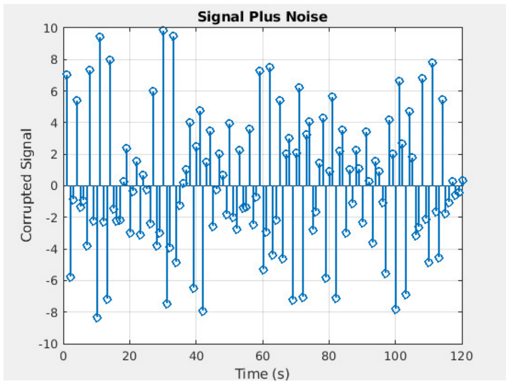


Fig. 28. Noisy Memory channel for SRRC pulse

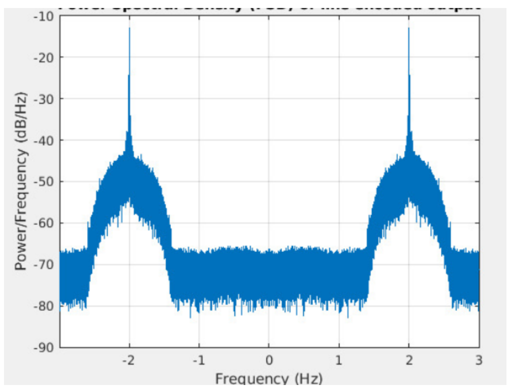


Fig. 29. Noisy Memory channel PSD for SRRC pulse

IV. Procedure - at the Receiver

After passing through the channel the received signal is processed at the Receiver to obtain the originally transmitter signal. This process involves demodulation, line decoding, decoder and D/A converter.

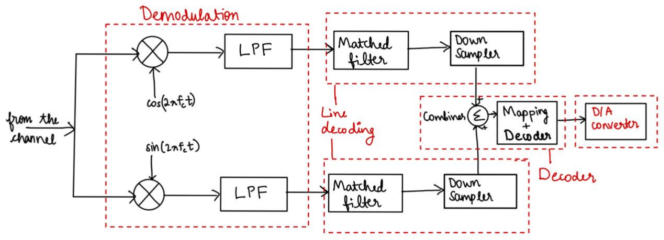


Fig. 30. Receiver for QAM

A. Demodulation

At the receiver, we assume the local carrier oscillator is exactly synchronized, and phase and frequency of local carrier of receiver part is same as at transmitter. The received signal is then multiplied with $cos(2\pi f_c t)$ and $sin(2\pi f_c t)$ to obtain the inphase (I) and quadrature (Q) of the signal. Here, $s(t)$ is composed of the symbols in-phase (a_I) mixed with cosine and the symbols in quadrature (a_Q) mixed with a sine.

$$\begin{aligned} \mathrm{I} &= r(t) \cdot \cos(2\pi f_c t) \\ &= s(t) \cdot \cos(2\pi f_c t) + n(t) \\ &= (a_I(t) \cdot \cos(2\pi f_c t) + a_Q(t) \cdot \sin(2\pi f_c t)) \cdot \cos(2\pi f_c t) \\ &= \frac{a_I(t)}{2} \cdot [1 + \cos(4\pi f_c t)] + \frac{a_Q(t)}{2} \cdot \sin(4\pi f_c t) + n(t) \end{aligned} \tag{1}$$

$$\begin{aligned} \mathrm{Q} &= r(t) \cdot \sin(2\pi f_c t) \\ &= s(t) \cdot \sin(2\pi f_c t) + n(t) \\ &= (a_I(t) \cdot \cos(2\pi f_c t) + a_Q(t) \cdot \sin(2\pi f_c t)) \cdot \sin(2\pi f_c t) \\ &= \frac{a_I(t)}{2} \cdot [\cos(4\pi f_c t)] + \frac{a_Q(t)}{2} \cdot [1 + \sin(4\pi f_c t)] + n(t) \end{aligned} \tag{2}$$

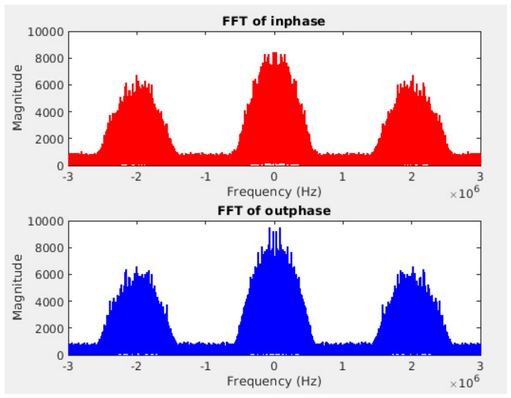


Fig. 31. FFT of the inphase and quadrature demodualted signals before LPF

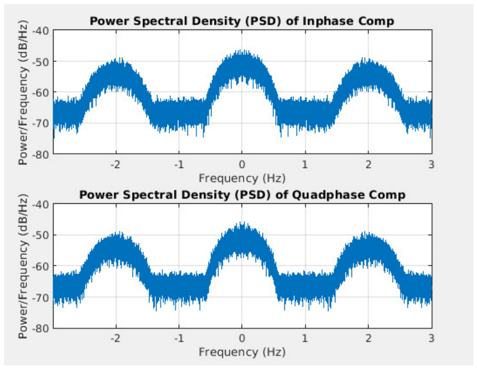


Fig. 32. PSD of the inphase and quadrature demodulated signals

The mixing process will result in a baseband signal with high-frequency components $(2f_c)$. The FFT contains two sidelobes indicating the presence of high frequency components at $(f - 2f_c)$ and $(f + 2f_c)$. The PSD also contains the sidelobes indicating the presence of high frequency components.

As only the baseband signal is required, these two signals are passed through a Butterworth Low Pass filter to obtain $a_I(t)$ and $a_Q(t)$.

A **Butterworth Low-Pass Filter** (LPF) is a type of analog or digital filter that attenuates or suppresses frequencies above a certain cutoff frequency while allowing lower frequencies to pass through with minimal attenuation. The key characteristic of a Butterworth LPF is its maximally flat frequency response in the passband, meaning that it has a uniform gain (or magnitude response) up to the cutoff frequency. This implies that the Butterworth LPF has no ripples or oscillations in the passband, resulting in a smooth transition from the passband to the stopband.

Mathematically, the transfer function $H(s)$ of an nth-order Butterworth LPF in the Laplace domain is given by:

$$H(s) = \frac{1}{1 + (\frac{s}{\omega_c})^{2n}} \tag{3}$$

where:

- s is the complex frequency variable,
- ω_c is the cutoff frequency,
- n is the filter order.

The filter is tuned with a normalised cutoff frequency of $Wn = \frac{2*f_c}{fs}$, where $f_c = 10^6 Hz$ and $fs = 6 * 10^6 Hz$. As a result it allows frequency components till $10^6 Hz$ to pass through without any attenuation and significantly attenuates the higher components.

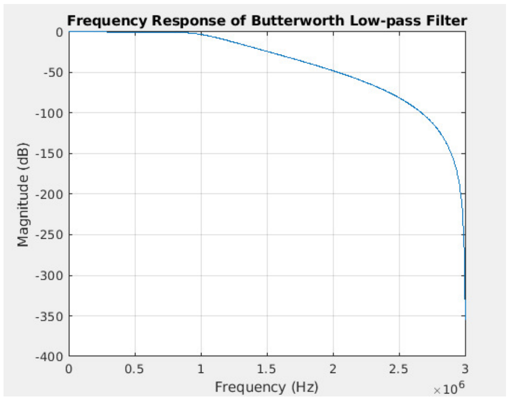


Fig. 33. Frequency response of the LPF

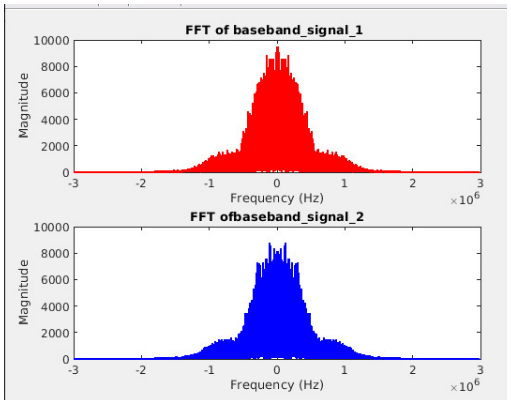


Fig. 34. FFT of the inphase and quadrature baseband signals after LPF

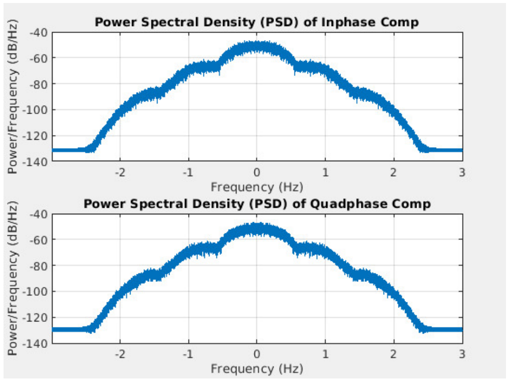


Fig. 35. PSD of the inphase and quadrature baseband signals after LPF

The LPF provides the baseband signals while eliminating the higher frequency components.

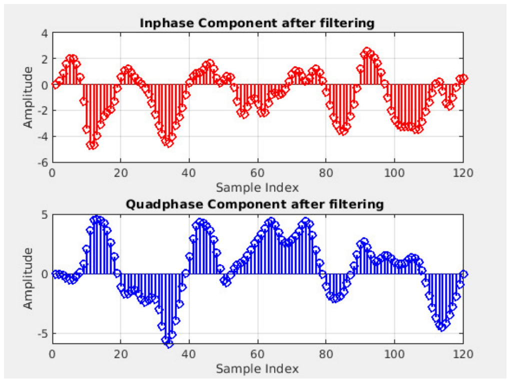


Fig. 36. Inphase and Quadrature signal after filtering in time domain

B. Line Decoding

In this step, the baseband signals obtained from the LPF are convoluted with the receive filter. The symbols $\{b[n]\}$

are passed through the transmit filter to obtain the waveform $\sum_n b[n]g_{TX}(t-nT_b)$. This then goes through the channel filter $g_C(t)$, and then the receive filter $g_{RX}(t)$. Thus, at the output of the receive filter, we have the linearly modulated signal $\sum_n b[n]p(t-nT_b)$, where $p(t) = (g_{TX} * g_C * g_{RX})(t)$ is the cascade of the transmit, channel, and receive filters.

The pulse $p(t)$ should be Nyquist at rate $\frac{10}{T_b}$, so that, in the absence of noise, the symbol rate samples at the output of the receive filter equal the transmitted symbols. In practice, the channel impulse response is unknown; hence, an ideal channel is assumed and a design such that the cascade of the transmit and receive filter, given by $(g_TX * g_{RX})(t)g_TX(f)g_{RX}(f)$, is Nyquist.

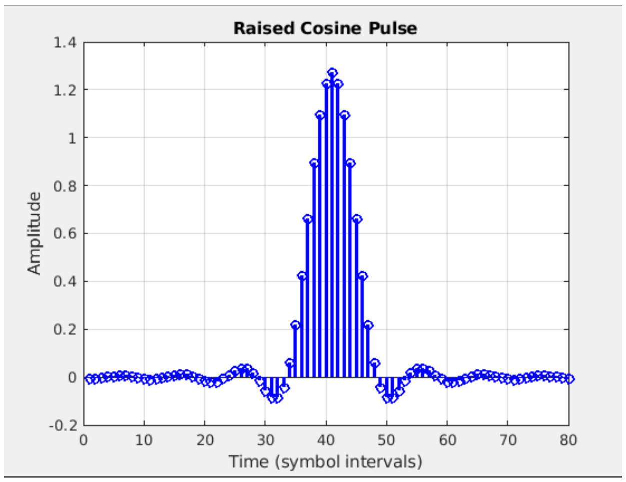


Fig. 37. Raised Cosine Pulse - Receive filter

As the digital signal was upsampled at the transmitter by a factor of 10, after convolution with the matched filter, the signal is again downsampled by a factor of 10 to get the original number of bits.

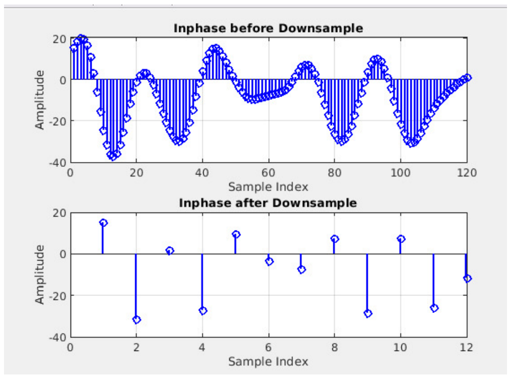


Fig. 38. Inphase signal through the Receive filter

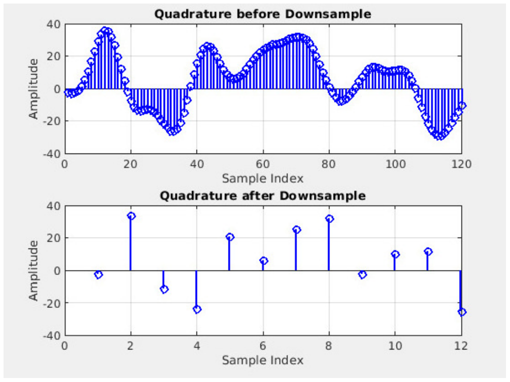


Fig. 39. Quadrature signal through the Receive filter

C. Decoder - MAP detection

Now, we have two signals r_I and r_Q. We define a signal $k = r_I + r_Q * i$ to obtain the received symbols for de-mapping. To design the decision device and obtain a threshold value to create sixteen non overlapping decision regions $(R_1, R_2....R_{16})$, each corresponding to one of the sixteen symbols according to the mapping scheme - the signal constellation for k is plotted.

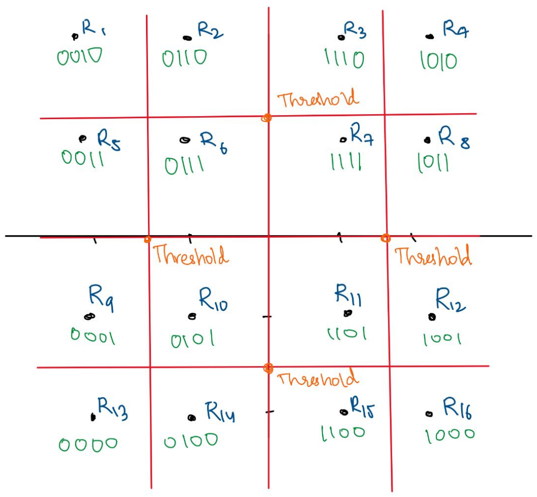


Fig. 40. Choosing a threshold value from the signal constellation to get minimum error

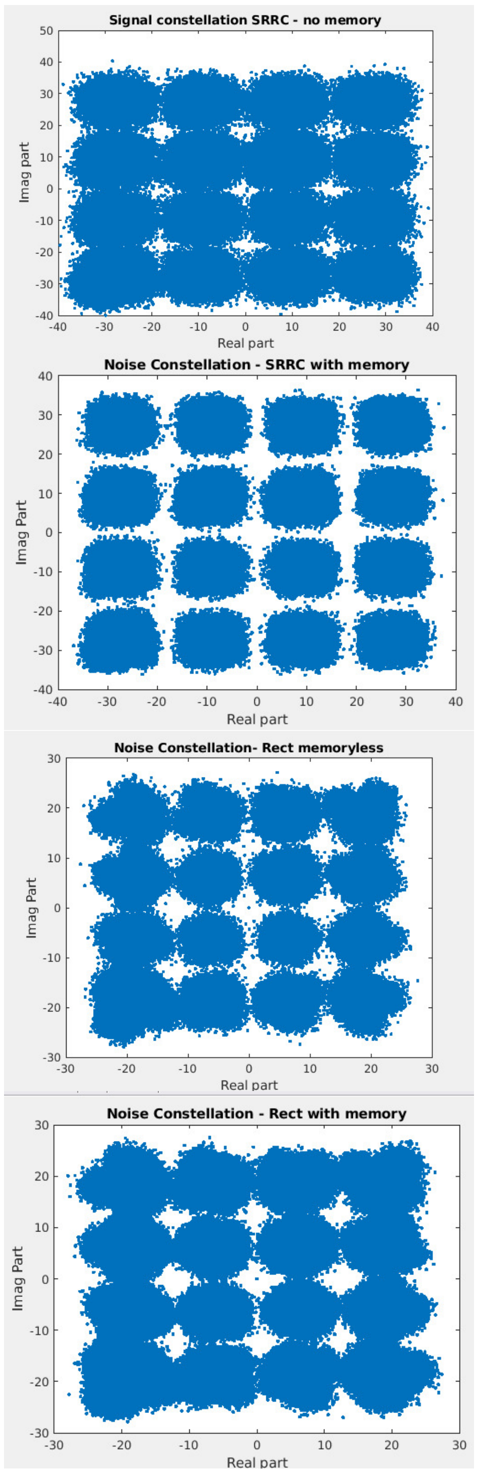


Fig. 41. Noise Constellations for SRRC and Rectangular pulse through AWGN Memoryless and Memory channels

Clearly, from these signal constellations, we obtain the threshold value as 20 for SRRC and 12 for the rectangular pulse. Mathematically, this value is $d/2$ where d is the distance between the mid points of two regions. Once these symbols are obtained, it is then decoded to information bits according to the mapping scheme also seen in Fig. 25.

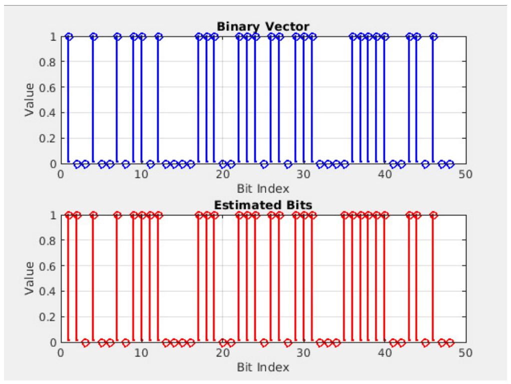


Fig. 42. Original vs Estimated bits

D. *Digital-to-Analog Converter (D/A Converter)*

Digital audio data, represented in binary form, is converted into analog signals using a 16-bit digital-to-analog converter (DAC). The process involves reshaping the binary data into a matrix, converting it to decimal, typecasting to 16-bit integers, and normalizing to fit within the range of [-1, 1]. This conversion enables faithful reconstruction of the original audio waveform for playback through audio output devices.

V. BER VS SNR ANALYSIS

A. ***Memoryless AWGN channel***

1) Raised Cosine Pulse: Bit Error Rate (BER) vs Signal-to-Noise Ratio (SNR) analysis is a fundamental aspect of communication systems performance evaluation. In this analysis, we investigate how the Bit Error Rate of a communication system varies with different levels of Signal-to-Noise Ratio.

In this analysis, we evaluate the Bit Error Rate (BER) over a range of Signal-to-Noise Ratio (SNR) values from 0 to 20 dB

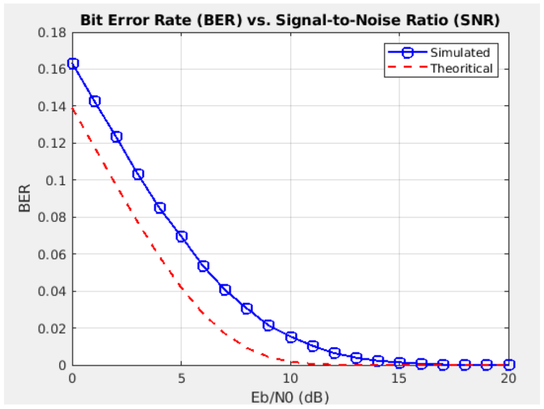


Fig. 43. SNR vs BER plot for both theoretical and Simulated

From the above plot, it is evident that our simulated bit error rate (BER) values consistently exceed the theoretical values. This deviation can be attributed to **Inter-symbol interference (ISI)** and overlapping effects.

Inter-symbol interference (ISI) occurs when the symbols transmitted in a digital communication system overlap with each other, leading to ambiguity in symbol detection at the receiver. This overlap can result from various factors such as multipath propagation, channel distortion, and insufficient pulse shaping. As a consequence, the received symbols may interfere with each other, causing errors in symbol detection and increasing the BER.

2) Rectangular Pulse: In this case, we plot the bit error rate (BER) for both simulated and ideal values. The ideal BER is determined using the formula derived in Section II.E.

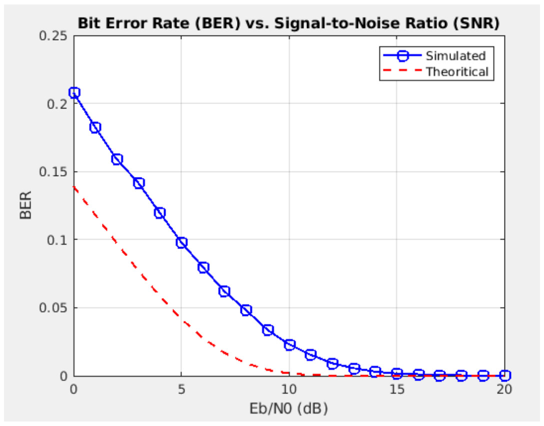


Fig. 44. SNR vs BER plot for both theoretical and Simulated

From the above plot, it is evident that our simulated bit error rate (BER) values consistently exceed the theoretical values as expected .

B. Memory AWGN channel

1) Raised Cosine Pulse: The channel impulse response $h(t)$ accounts for channel memory and is defined as:

$$h(t) = a\delta(t) + (1-a)\delta(t-bT_b)$$

where $\delta(t)$ is the Dirac delta function, a is the direct path attenuation factor, b is the relative delay of the multipath component in terms of symbol duration T_b. In our specific case,

we consider $a = 0.65$ and $b = 1$, implying that the current transmitted symbol has a higher weightage (0.65) compared to the previous transmitted symbol (1). This indicates that the present value has a weightage of 0.65 (as it is multiplied by a) and the past values are slightly weighed (as they are multiplied by $1-a$).

This model allows us to account for the influence of both the current and previous transmitted symbols on the received signal.

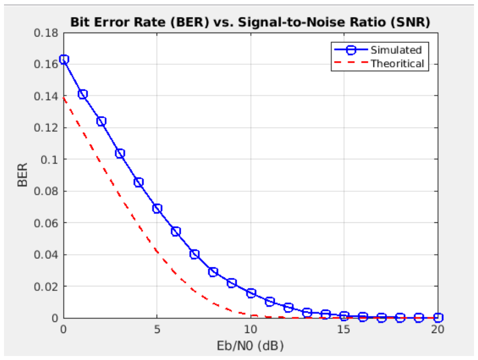


Fig. 45. SNR vs BER plot for both theoretical and Simulated

2) Rectangular Pulse: In this case , we consider the same value of a and b as in previous case and we observe the following :

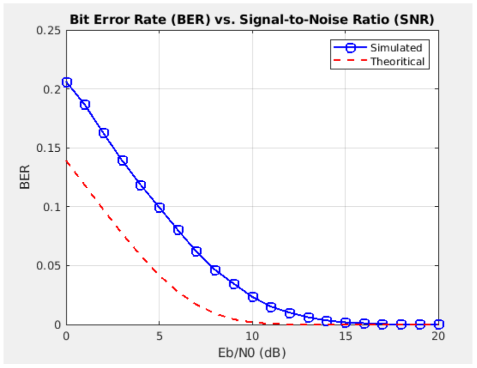


Fig. 46. SNR vs BER plot for both theoretical and Simulated

VI. Comparing Memory and Memoryless Channel

A. Raised Cosine Pulse

Shown below is the plot comparing the BER in both memory and memoryless channel.

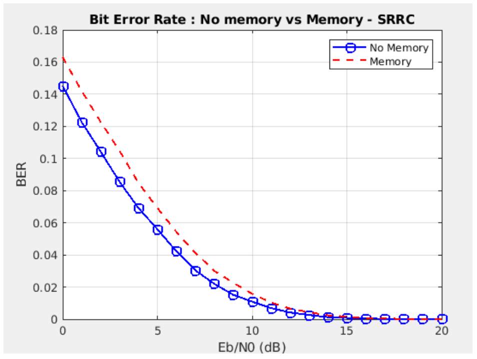


Fig. 47. Comparing BER in both memory and memoryless channel for Raised Cosine

In a memoryless channel:

- Each symbol transmission is independent of previous symbols.

- The noise affecting **each symbol is uncorrelated** with the noise affecting other symbols.
- The BER is primarily determined by the Signal-to-Noise Ratio (SNR) and the modulation scheme used.
- As the SNR increases, the BER tends to decrease, assuming a fixed modulation scheme and channel characteristics.

In a channel with memory:

- The current symbol transmission can be influenced by previous symbols due to channel effects like intersymbol interference (ISI) or noise correlations.
- The presence of memory can lead to a **higher BER compared to a memoryless channel**, especially if the receiver does not compensate adequately for the channel's memory.
- Techniques such as **equalization or coding** are often employed to mitigate the effects of channel memory and reduce BER.
- BER performance depends not only on the SNR but also on the channel's memory characteristics and the effectiveness of the mitigation techniques employed.

B. *Rectangular Pulse*

In case of rectangular pulse , we observe the following :

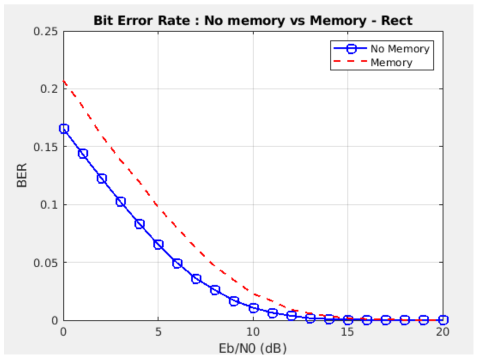


Fig. 48. Comparing BER in both memory and memoryless channel for Rectangular Pulse

Comparing Memory and Memoryless Channel :

- **Independence**: Memoryless channels exhibit symbol independence, while channels with memory introduce dependencies between symbols.
- **Impact of Noise**: In memoryless channels, noise affecting each symbol is uncorrelated, whereas in channels with memory, noise correlations may increase BER.
- **Mitigation Techniques**: Channels with memory often require additional equalization or coding techniques to mitigate memory effects and reduce BER, unlike memoryless channels.

References

[1] Lathi, B. P., Ding, Z. (2019). Modern digital and Analog Communication Systems (5th ed.). Oxford University Press.