A Project Report On

Simulation of different Modulation schemes and calculating Bit Error Rate by using different channels in Matlab

By

Vadlamudi Girish

(11013327)

Under the guidance of

Prof.Mrs. Pooja Mohnani

Contents

Abstract:

To transmit the information more efficiently from one place to another i.e. from source to destination, we have several modulation and multiplexing techniques. For some applications more speed is required and some errors in the transmitting information are allowed and for other applications speed is not the first priority but more precise information has to be transmitted. Considering the transmission of digital information, depending on the applications we choose different modulation and multiplexing schemes to transmit a message signal with the help of a carrier signal. In this project I have simulated several modulation and multiplexing techniques along with the noisy channels to estimate the Bit Error Rate when the message signal is transmitted from source to destination. The obtained results might be similar when applied to the real life applications. Here Signal to Noise Ratio and Bit Error Rate are the main measuring elements to calculate the efficiency of a communication system which is using certain modulation techniques. Based on these values we can adjust the gain of the receiver to get the maximum possible efficiency.

Motivation:

The motivation is to compare different Modulation schemes in terms of Bit Error Rate by transmitting the message signal through simulated noisy channels.

Aim:

Aim is to show the simulated results so that it will be easy to choose the modulation scheme and assume the channel characteristics while applying these schemes to the real life applications.

Introduction:

Now a day's communication (digital) became the part of life for everyone in this world. However the evaluation of the digital communication was pioneered by Claude Shannon in 1948 through his research paper titled "A Mathematical Theory of Communication". After his work, many other scientists worked in this field and implemented several ways to make the communication digital, fast and reliable. At present we are having the extremely high speed communication systems which can be easily accessed in our day to day life. Along with the implementation of new techniques in communication system, several new applications to use these communication systems also evolved. Theoretically we feel that communication can be easily done, But it is not so easy to establish a communication because of the hardware and atmospheric limitations. Due to these noise gets added to the message signal while transmitting. It is the fact that we cannot completely eliminate the noise but we can recover the acceptable message signal at the receiver by using several error correcting methods.

System Requirements:

To simulate several modulation and multiplexing techniques along with the noisy channels to estimate the Bit Error Rate I am using Matlab 2019a with the 'Communication systems' tool box. Even though I have implemented most of the techniques in terms of mathematical expressions, for some of the channels and modulations, I have used the direct Matlab inbuilt function objects which requires the latest version of Matlab with Communication Systems toolbox installed.

Technical Details:

i) Message Signal:

The signal which contains a message to be transmitted, is called as a message signal. It is a baseband signal, which has to undergo the process of modulation, to get transmitted. Hence, it is also called as the modulating signal.

ii) Carrier Signal:

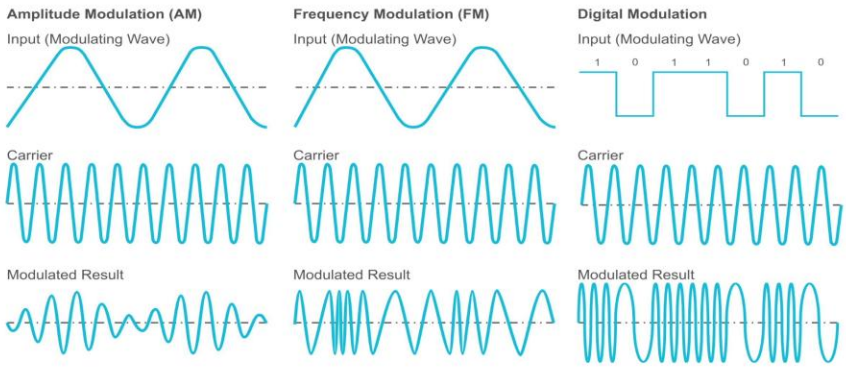
In telecommunications, a carrier wave, carrier signal, or just carrier, is a waveform (usually sinusoidal) that is modulated (modified) with an information bearing signal for the purpose of conveying information. This carrier wave usually has a much higher frequency than the input signal does. The purpose of the carrier is usually either to transmit the information through space as an electromagnetic wave (as in radio communication), or to allow several carriers at different frequencies to share a common physical transmission medium by frequency division multiplexing (as in a cable television system, for example). The term originated in radio communication, where the carrier wave creates the radio waves which carry the information (modulation) through the air from the transmitter to the receiver. The term is also used for an unmodulated emission in the absence of any modulating signal.

iii) Modulation:

In electronics and telecommunications, modulation is the process of varying one or more properties of a periodic waveform, called the carrier signal, with a modulating signal that typically contains information to be transmitted.

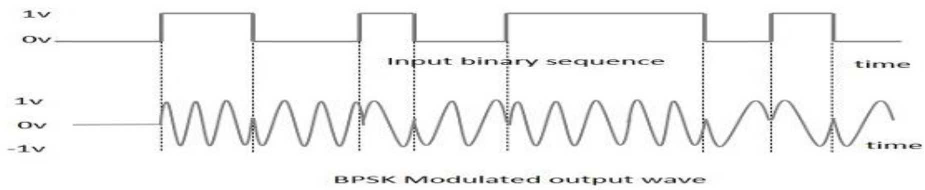
The aim of digital baseband modulation methods, also known as line coding, is to transfer a digital bit stream over a baseband channel, typically a non-filtered copper wire such as a serial bus or a wired local area network.

The aim of pulse modulation methods is to transfer a narrowband analog signal, for example, a phone call over a wideband baseband channel or, in some of the schemes, as a bit stream over another digital transmission system.



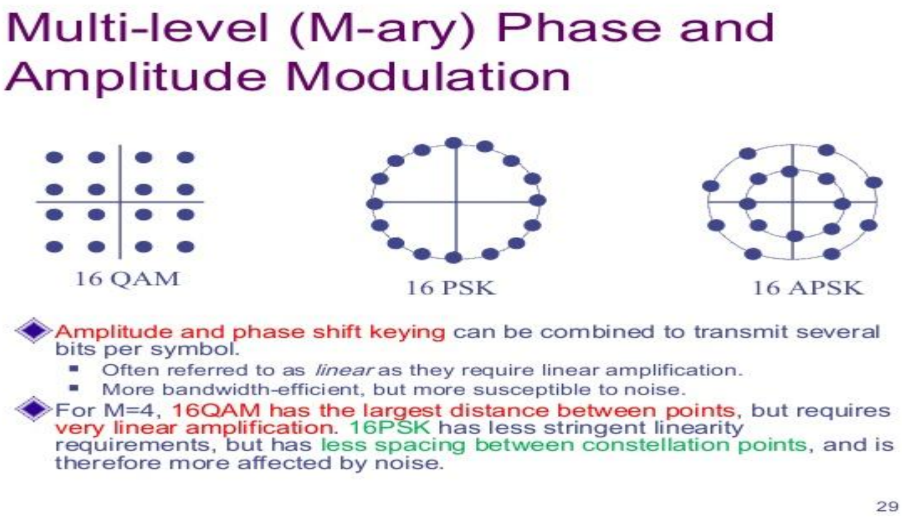
a) Phase Shift keying (PSK):

Phase-shift keying (PSK) is a digital modulation process which conveys data by changing (modulating) the phase of a constant frequency reference signal (the carrier wave). The modulation is accomplished by varying the sine and cosine inputs at a precise time. It is widely used for wireless LANs, RFID and Bluetooth communication.



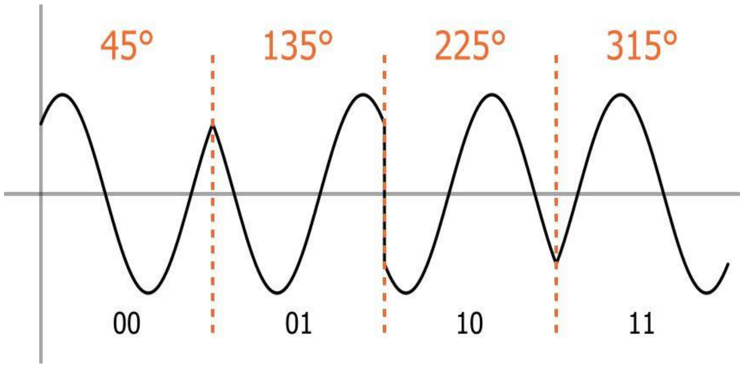
b) M- array Phase Shift Keying (M-PSK):

M-ary phase-shift keying (MPSK) is employed in some of the digital cellular standards and communication geostationary satellite systems. MPSK employs a set of *M* equal-energy signals to represent *M* equiprobable symbols. This constant energy restriction (i.e., the constant envelope constraint) warrants a circular constellation for the signal points. In MPSK, the phase of the carrier takes on one of *M* possible values.



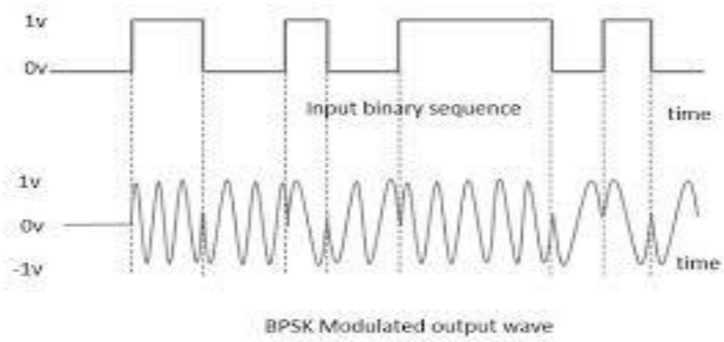
c) Quadrature Phase Shift Keying (QPSK):

QPSK-Quadrature Phase-Shift Keying is a system of modulating digital signals onto a radio-frequency carrier signal using four phase 0, 90, 180, 270 degree states to code two digital bits. Offset Quadriphase Shift Keying is a phase shift of 45 degrees in each one of its two binary channels.



d) Binary Phase Shift Keying (BPSK):

Binary Phase Shift Keying (BPSK) is a two phase modulation scheme, where the 0's and 1's in a binary message are represented by two different phase states in the carrier signal: for binary 1 and for binary 0. The carrier's phase contains all the information that is being transmitted.



iv) Channel:

A communication channel refers either to a physical transmission medium such as a wire, or to a logical connection over a multiplexed medium such as a radio channel in telecommunications and computer networking. A channel is used to convey an information signal, for example a digital bit stream, from one or several senders (or transmitters) to one or several receivers. A channel has a certain capacity for transmitting information, often measured by its bandwidth in Hz or its data rate in bits per second.

a) Additive White Gaussian Noise (AWGN) Channel:

Additive white Gaussian noise (AWGN) is a basic noise model used in Information theory to mimic the effect of many random processes that occur in nature. The modifiers denote specific characteristics:

- **Additive** because it is added to any noise that might be intrinsic to the information system.
- **White** refers to the idea that it has uniform to power across the frequency band for the information system. It is an analogy to the color white which has uniform emissions at all frequencies in the visible spectrum.
- **Gaussian** because it has a normal distribution in the time domain with an average time domain value of zero.

Wideband noise comes from many natural noise, such as the thermal vibrations of atoms in conductors (referred to as thermal noise or Johnson–Nyquist noise), shot noise, black-body radiation from the earth and other warm objects, and from celestial sources such as the Sun. The central limit theorem of probability theory indicates that the summation of many random processes will tend to have distribution called Gaussian or Normal.

b) Rayleigh Fading Channel:

Rayleigh fading is a statistical model for the effect of a propagation environment on a radio signal, such as that used by wireless devices.

Rayleigh fading models assume that the magnitude of a signal that has passed through such a transmission medium (also called a communication channel) will vary randomly, or fade, according to a Rayleigh distribution — the radial component of the sum of two uncorrelated Gaussian random variables.

Rayleigh fading is viewed as a reasonable model for tropospheric and ionospheric signal propagation as well as the effect of heavily built-up urban environments on radio signals. Rayleigh fading is most applicable when there is no dominant propagation along a line of sight between the transmitter and receiver. If there is a dominant line of sight, Rician fading may be more applicable. Rayleigh fading is a special case of two-wave with diffuse power (TWDP) fading.

c) Rician Fading Channel:

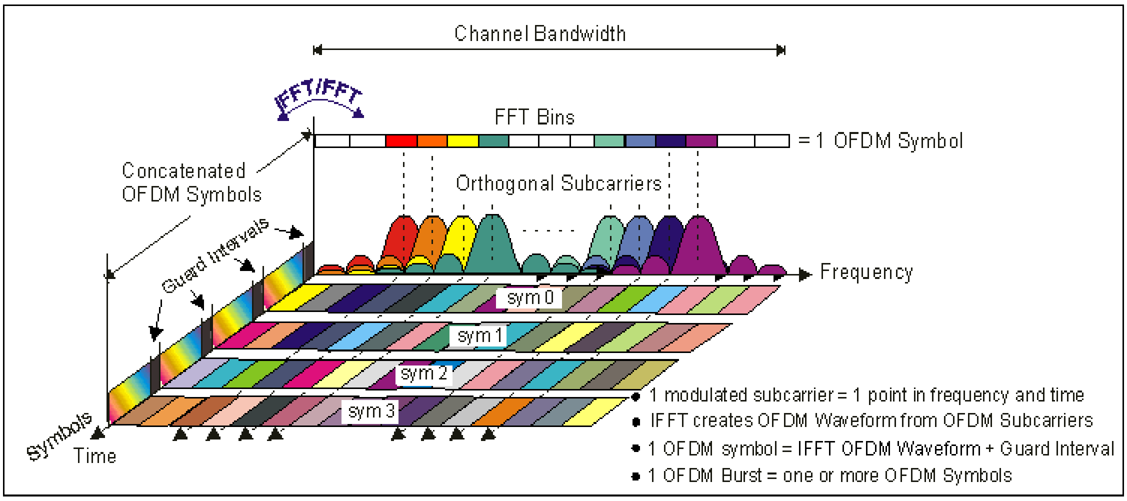
Rician fading is a stochastic model for radio propagation anomaly caused by partial cancellation of a radio signal by itself — the signal arrives at the receiver by several different paths (hence exhibiting multipath interference), and at least one of the paths is changing (lengthening or shortening). Rician fading occurs when one of the paths, typically a line of sight signal or some strong reflection signals, is much stronger than the others. In Rician fading, the amplitude gain is characterized by a Rician distribution.

Rayleigh fading is the specialized model for stochastic fading when there is no line of sight signal, and is sometimes considered as a special case of the more generalized concept of Rician fading. In Rayleigh fading, the amplitude gain is characterized by a Rayleigh distribution. Rician fading itself is a special case of two-wave with diffuse power (TWDP) fading.

v) Orthogonal Frequency Division Multiplexing (OFDM):

In telecommunications, orthogonal frequency-division multiplexing (OFDM) is a type of digital modulation, a method of encoding digital data on multiple carrier frequencies. OFDM has developed into a popular scheme for wideband digital communication, used in applications such as digital television and audio broadcasting, DSL internet access, wireless networks, power line networks, and 4G mobile communications.

OFDM is a frequency-division multiplexing (FDM) scheme used as a digital multi-carrier modulation method. It was introduced by Chang of Bell Labs in 1966. In OFDM, multiple closely spaced orthogonal subcarrier signals with overlapping spectra are transmitted to carry data in parallel.[4] Demodulation is based on Fast Fourier Transform algorithms. OFDM was improved by Weinstein and Ebert in 1971 with the introduction of a guard interval, providing better orthogonality in transmission channels affected by multipath propagation.[5] Each subcarrier (signal) is modulated with a conventional modulation scheme (such as Quadrature amplitude modulation or phase shift keying) at a low symbol rate. This maintains total data rates similar to conventional single-carrier modulation schemes in the same bandwidth.



Frequency-Time Representative of an OFDM signal

vi) Reed Solomon Coding:

Reed–Solomon codes are a group of error-correcting codes that were introduced by Irving S. Reed and Gustave Solomon in 1960.[1] They have many applications, the most prominent of which include consumer technologies such as CDs, DVDs, Blue-ray discs, QR codes, data transmission technologies such as DSL and WiMAX, broadcast systems such as satellite communications, DVB and ATSC, and storage systems such as RAID 6.

Reed–Solomon codes operate on a block of data treated as a set of finite field elements called symbols. Reed–Solomon codes are able to detect and correct multiple symbol errors. By adding t check symbols to the data, a Reed–Solomon code can detect (but not correct) any combination of up to and including t erroneous symbols, OR correct up to and including $\lfloor t/2 \rfloor$ symbols at arbitrary locations. As an erasure code, it can correct up to and including t erasures at locations that are known and provided to the algorithm, or it can detect and correct combinations of errors and erasures. Reed–Solomon codes are also suitable as multiple-burst bit-error correcting codes, since a sequence of b + 1 consecutive bit errors can affect at most two symbols of size b. The choice of t is up to the designer of the code, and may be selected within wide limits.

vii) Gray coding:

A Gray Code represents numbers using a binary encoding scheme that groups a sequence of bits so that only one bit in the group changes from the number before and after. It is named for Bell Labs researcher Frank Gray, who described it in his 1947 patent submittal on Pulse Code Communication.

viii) Signal to Noise Ratio:

Signal-to-noise ratio (abbreviated SNR or S/N) is a measure used in science and engineering that compares the level of a desired signal to the level of background noise. SNR is defined as the ratio of signal power to the noise power, often expressed in decibels. A ratio higher than 1:1 (greater than 0 dB) indicates more signal than noise.

While SNR is commonly quoted for electrical signals, it can be applied to any form of signal, for example isotope levels in an ice core, biochemical signaling between cells, or financial trading signals. Signal-to-noise ratio is sometimes used metaphorically to refer to the ratio of useful information to false or irrelevant data in a conversation or exchange. For example, in online discussion forums and other online communities, off-topic posts and spam are regarded as "noise" that interferes with the "signal" of appropriate discussion.

$$\mathrm{SNR} = \frac{P_{\mathrm{signal}}}{P_{\mathrm{noise}}},$$

$$\mathrm{SNR} = \frac{P_{\mathrm{signal}}}{P_{\mathrm{noise}}} = \left(\frac{A_{\mathrm{signal}}}{A_{\mathrm{noise}}}\right)^2,$$

$$\mathrm{SNR_{dB}} = 10\log_{10}\left[\left(\frac{A_{\mathrm{signal}}}{A_{\mathrm{noise}}}\right)^2\right] = 20\log_{10}\left(\frac{A_{\mathrm{signal}}}{A_{\mathrm{noise}}}\right) = \left(A_{\mathrm{signal,dB}} - A_{\mathrm{noise,dB}}\right).$$

ix) Bit Error Rate:

In digital transmission, the number of bit errors is the number of received bits of a data stream over a communication channel that have been altered due to noise, interference, distortion or bit synchronization errors.

The bit error rate (BER) is the number of bit errors per unit time. The bit error ratio (also BER) is the number of bit errors divided by the total number of transferred bits during a studied time interval. Bit error ratio is a unit less performance measure, often expressed as a percentage.

The bit error probability pe is the expectation value of the bit error ratio. The bit error ratio can be considered as an approximate estimate of the bit error probability. This estimate is accurate for a long time interval and a high number of bit errors.

$$\mathrm{BER} = \frac{1}{2}\,\mathrm{erfc}\left(\sqrt{E_b/N_0}\right).$$

Matlab codes:

a. QPSK with OFDM on AWGN Channel:

```
M = 4;              % Modulation alphabet
k = log2(M);        % Bits/symbol
numSC = 128;        % Number of OFDM subcarriers
cpLen = 32;         % OFDM cyclic prefix length
maxBitErrors = 100;  % Maximum number of bit errors
maxNumBits = 1e7;    % Maximum number of bits transmitted
%%
%

qpskMod = comm.QPSKModulator('BitInput',true);
qpskDemod = comm.QPSKDemodulator('BitOutput',true);
%%
%

ofdmMod = comm.OFDMModulator('FFTLength',numSC,'CyclicPrefixLength',cpLen);
ofdmDemod = comm.OFDMDemodulator('FFTLength',numSC,'CyclicPrefixLength',cpLen);
%%
%

channel = comm.AWGNChannel('NoiseMethod','Variance', ...
   'VarianceSource','Input port');
%%
%

errorRate = comm.ErrorRate('ResetInputPort',true);
%%
%

ofdmDims = info(ofdmMod)
%%
%
```

```
numDC = ofdmDims.DataInputSize(1)
%%
%

frameSize = [k*numDC 1];
%%
EbNoVec = (0:10)';
snrVec = EbNoVec + 10*log10(k) + 10*log10(numDC/numSC);
%%
%

berVec = zeros(length(EbNoVec),3);
errorStats = zeros(1,3);
%%
%

for m = 1:length(EbNoVec)
  snr = snrVec(m);

  while errorStats(2) <= maxBitErrors && errorStats(3) <= maxNumBits
    dataIn = randi([0,1],frameSize);              % Generate binary data
    qpskTx = qpskMod(dataIn);                     % Apply QPSK modulation
    txSig = ofdmMod(qpskTx);                      % Apply OFDM modulation
    powerDB = 10*log10(var(txSig));               % Calculate Tx signal power
    noiseVar = 10.^(0.1*(powerDB-snr));           % Calculate the noise variance
    rxSig = channel(txSig,noiseVar);              % Pass the signal through a noisy channel
    qpskRx = ofdmDemod(rxSig);                    % Apply OFDM demodulation
    dataOut = qpskDemod(qpskRx);                  % Apply QPSK demodulation
    errorStats = errorRate(dataIn,dataOut,0);     % Collect error statistics
  end

  berVec(m,:) = errorStats;                       % Save BER data
  errorStats = errorRate(dataIn,dataOut,1);       % Reset the error rate calculator
end
%%
%

berTheory = berawgn(EbNoVec,'psk',M,'nondiff');
```

```
%%
%

figure
semilogy(EbNoVec,berVec(:,1),'*')
hold on
semilogy(EbNoVec,berTheory)
legend('Simulation','Theory','Location','Best')
xlabel('Eb/No (dB)')
ylabel('Bit Error Rate')
grid on
hold off
```

b. QPSK with Gray Coding on Rayleigh Channel:

```
% QPSK simulation with Gray coding and simple Rayleigh (no LOS) multipath
% and AWGN included.
% Clear all the previously used variables and close all figures
clear all;
close all;
format long;
% Frame Length
bit_count = 10000;
% Range of SNR over which to simulate
Eb_No = -3: 1: 30;
% Convert Eb/No values to channel SNR
% Consult BERNARD SKLAR'S book 'Digital Communications'
SNR = Eb_No + 10*log10(2);
% Start the main calculation loop
for aa = 1: 1: length(SNR)

  % Initiate variables
  T_Errors = 0;
  T_bits = 0;

  % Keep going until you get 100 errors
  while T_Errors < 100
```

```
    % Generate some information bits
    uncoded_bits  = round(rand(1,bit_count));

    % Split the stream into two streams, for Quadrature Carriers
    B1 = uncoded_bits(1:2:end);
    B2 = uncoded_bits(2:2:end);

    % QPSK modulator set to pi/4 radians constellation
    % If you want to change the constellation angles
    % just change the angles. (Gray Coding)
    qpsk_sig = ((B1==0).*(B2==0)*(exp(i*pi/4))+(B1==0).*(B2==1)...
      *(exp(3*i*pi/4))+(B1==1).*(B2==1)*(exp(5*i*pi/4))...
      +(B1==1).*(B2==0)*(exp(7*i*pi/4)));

    % Variance = 0.5 - Tracks theoritical PDF closely
    ray = sqrt(0.5*((randn(1,length(qpsk_sig))).^2+(randn(1,length(qpsk_sig))).^2));

    % Include The Multipath
    rx = qpsk_sig.*ray;

    % Noise variance
    N0 = 1/10^(SNR(aa)/10);

    % Send over Gaussian Link to the receiver
    rx = rx + sqrt(N0/2)*(randn(1,length(qpsk_sig))+i*randn(1,length(qpsk_sig)));

%---------------------------------------------------------------

    % Equaliser
    rx = rx./ray;
    % QPSK demodulator at the Receiver
    B4 = (real(rx)<0);
    B3 = (imag(rx)<0);

    uncoded_bits_rx = zeros(1,2*length(rx));
    uncoded_bits_rx(1:2:end) = B3;
    uncoded_bits_rx(2:2:end) = B4;
```

```
        % Calculate Bit Errors
        diff = uncoded_bits - uncoded_bits_rx;
        T_Errors = T_Errors + sum(abs(diff));
        T_bits = T_bits + length(uncoded_bits);

    end
    % Calculate Bit Error Rate
    BER(aa) = T_Errors / T_bits;
    fprintf('bit error probability = %f\n',BER(aa));
end

%-------------------------------------------------------------
% Finally plot the BER Vs. SNR(dB) Curve on logarithmic scale
% BER through Simulation
figure(1);
semilogy(SNR,BER,'or');
hold on;
xlabel('SNR (dB)');
ylabel('BER');
title('SNR Vs BER plot for QPSK Modualtion in Rayleigh Channel');
% Rayleigh Theoretical BER
figure(1);
EbN0Lin = 10.^(Eb_No/10);
theoryBerRay = 0.5.*(1-sqrt(EbN0Lin./(EbN0Lin+1)));
semilogy(SNR,theoryBerRay);
grid on;
% Theoretical BER
figure(1);
grid on;
legend('Simulated', 'Theoretical Raylegh', 'Theroretical AWGN');
axis([SNR(1,1) SNR(end-3) 0.00001 1]);
```

c. PSK with RS Coding on AWGN Channel:

```
M = 8;          % Modulation order
bps = log2(M);   % Bits per symbol
```

```
N = 7;          % RS codeword length
K = 5;          % RS message length

pskModulator = comm.PSKModulator('ModulationOrder',M,'BitInput',true);
pskDemodulator = comm.PSKDemodulator('ModulationOrder',M,'BitOutput',true);
awgnChannel = comm.AWGNChannel('BitsPerSymbol',bps);
errorRate = comm.ErrorRate;



rsEncoder = comm.RSEncoder('BitInput',true,'CodewordLength',N,'MessageLength',K);
rsDecoder = comm.RSDecoder('BitInput',true,'CodewordLength',N,'MessageLength',K);
%%
% Set the range of $E_b/N_0$ values. Initialize the error statistics matrix.

ebnoVec = (3:0.5:8)';
errorStats = zeros(length(ebnoVec),3);
%%
% Estimate the bit error rate for each $E_b/N_0$ value. The simulation runs
% until either 100 errors or $10^7$ bits is encountered. The main simulation loop
% processing includes encoding, modulation, demodulation, and decoding.

for i = 1:length(ebnoVec)
  awgnChannel.EbNo = ebnoVec(i);
  reset(errorRate)
  while errorStats(i,2) < 100 && errorStats(i,3) < 1e7
    data = randi([0 1],1500,1);               % Generate binary data
    encData = rsEncoder(data);                % RS encode
    modData = pskModulator(encData);          % Modulate
    rxSig = awgnChannel(modData);             % Pass signal through AWGN
    rxData = pskDemodulator(rxSig);           % Demodulate
    decData = rsDecoder(rxData);              % RS decode
    errorStats(i,:) = errorRate(data,decData);  % Collect error statistics
  end
end
%%
% Fit a curve to the BER data using |berfit|. Generate an estimate of 8-PSK
% performance without coding using the |berawgn| function.
```

```
berCurveFit = berfit(ebnoVec,errorStats(:,1));
berNoCoding = berawgn(ebnoVec,'psk',8,'nondiff');
%%
% Plot the BER data, the BER curve fit, and the estimated performance without
% RS coding.

semilogy(ebnoVec,errorStats(:,1),'b*', ...
ebnoVec,berCurveFit,'c-',ebnoVec,berNoCoding,'r')
ylabel('BER')
xlabel('Eb/No (dB)')
legend('Data','Curve Fit','No Coding')
grid
%%
% The (7,5) RS code improves the $E_b/N_0$ required to achieve a $10^{-2}$ bit
% error rate by, approximately, 1.4 dB.
%
```

d. BPSK on Rayleigh Fading Channel:

```
clear
N = 10^6 % number of bits or symbols
% Transmitter
ip = rand(1,N)>0.5; % generating 0,1 with equal probability
s = 2*ip-1; % BPSK modulation 0 -> -1; 1 -> 0

Eb_N0_dB = [-3:35]; % multiple Eb/N0 values
for ii = 1:length(Eb_N0_dB)

  n = 1/sqrt(2)*[randn(1,N) + j*randn(1,N)]; % white gaussian noise, 0dB variance
  h = 1/sqrt(2)*[randn(1,N) + j*randn(1,N)]; % Rayleigh channel

  % Channel and noise Noise addition
  y = h.*s + 10^(-Eb_N0_dB(ii)/20)*n;
  % equalization
  yHat = y./h;
  % receiver - hard decision decoding
  ipHat = real(yHat)>0;
  % counting the errors
  nErr(ii) = size(find([ip- ipHat]),2);
```

```
end
simBer = nErr/N; % simulated ber
theoryBerAWGN = 0.5*erfc(sqrt(10.^(Eb_N0_dB/10))); % theoretical ber
EbN0Lin = 10.^(Eb_N0_dB/10);
theoryBer = 0.5.*(1-sqrt(EbN0Lin./(EbN0Lin+1)));
% plot
close all
figure
hold on
semilogy(Eb_N0_dB,theoryBer,'bp-','LineWidth',2);
semilogy(Eb_N0_dB,simBer,'mx-','LineWidth',2);
axis([-3 35 10^-5 0.5])
grid on
legend('Rayleigh-Theory', 'Rayleigh-Simulation');
xlabel('Eb/No, dB');
ylabel('Bit Error Rate');
title('BER for BPSK modulation in Rayleigh channel');
```

e. M- Array PSK on Rician Fading Channel:

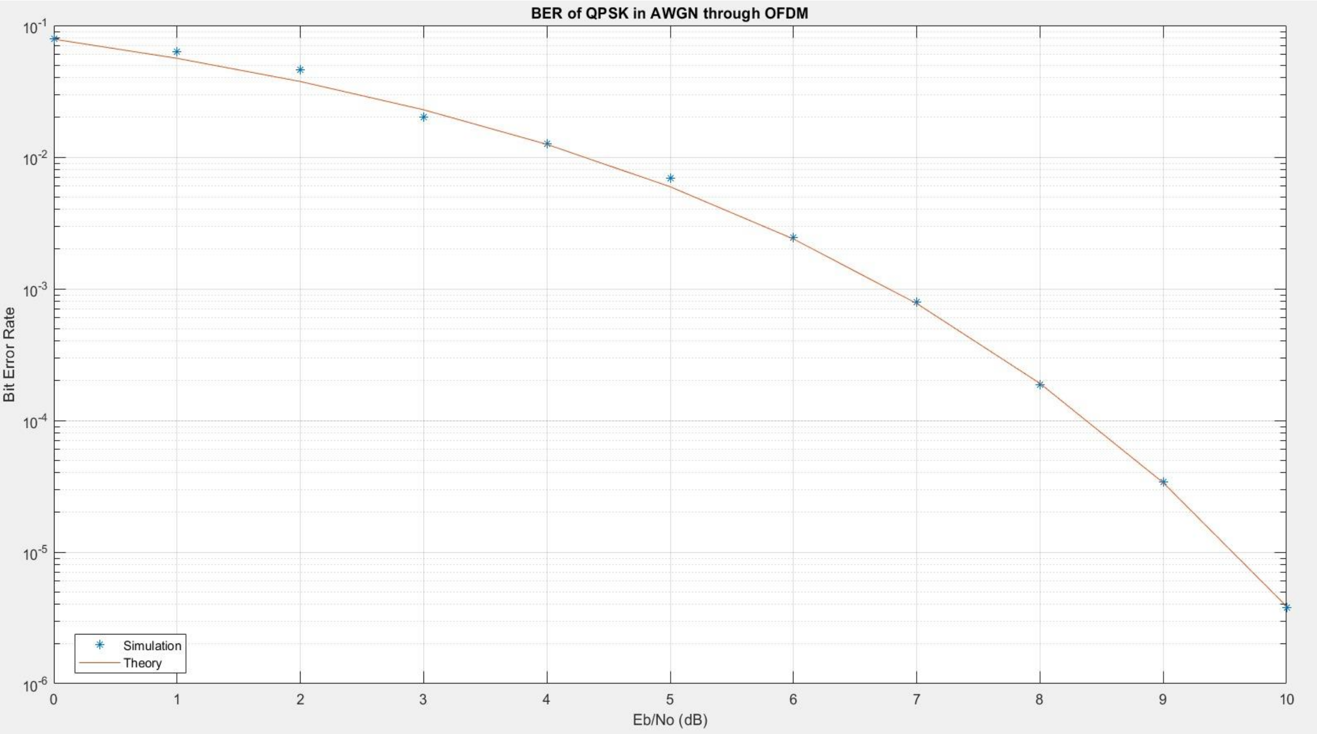
```
clear all;
close all;
clc
EbNo = 0:40;
K = [4.0; 0.6];
M = [4; 8; 16; 64; 256]; %Positions of modulation (M-PSK or M-QAM)
for k = 1:length(K)
  for m = 1:length(M)
    message = randi([0, M(m)-1], 100000, 1);
    if M(m) >= 16
      mod_msg = qammod(message, M(m), pi/4, 'gray');
      ric_ber(:, m, k) = berfading(EbNo,'qam',M(m),1,K(k));
    else
      mod_msg = pskmod(message, M(m), pi/4, 'gray');
      ric_ber(:, m, k) = berfading(EbNo, 'psk', M(m), 1, K(k));
    end
    Es = mean(abs(mod_msg).^2);
    No = Es./((10.^(EbNo./10))*log2(M(m)));
    h = sqrt( K(k)/(K(k)+1)) +...
```

```
        sqrt( 1/(K(k)+1))*(1/sqrt(2))*(randn(size(mod_msg))...
         + 1j*randn(size(mod_msg)));
        ric_msg = mod_msg.*h; % Rician flat fading
        for c = 1:100
            for jj = 1:length(EbNo)
                noisy_mod = ric_msg +...
                sqrt(No(jj)/2)*(randn(size(mod_msg))+...
                1j*randn(size(mod_msg))); %AWGN
                noisy_mod = noisy_mod ./ h; % zero-forcing equalization
                if M(m) >= 16
                    demod_msg = qamdemod(noisy_mod, M(m), pi/4, 'gray');
                else
                    demod_msg = pskdemod(noisy_mod, M(m), pi/4, 'gray');
                end
                [number,BER(c,jj)] = biterr(message,demod_msg);
            end
        end
        sum_BER(:,m, k) = sum(BER)./c;
    end
end
figure(1)
semilogy(EbNo, sum_BER(:,1,1), 'b-o', EbNo, sum_BER(:,2,1), 'r-o',...
EbNo, sum_BER(:,3,1), 'g-o', EbNo, sum_BER(:,4,1), 'c-o',...
EbNo, sum_BER(:,5,1), 'k-o',...
EbNo, ric_ber(:,1,1), 'b-', EbNo, ric_ber(:,2,1), 'r-',...
EbNo, ric_ber(:,3,1), 'g-', EbNo, ric_ber(:,4,1), 'c-',...
EbNo, ric_ber(:,5,1), 'k-', 'LineWidth', 1.5)
title('Rician model (K = 4.0)')
legend('QPSK(simulated)', '8-PSK(simulated)',...
'16-QAM(simulated)', '64-QAM(simulated)' ,'256-QAM(simulated)',...
'QPSK(theory)','8-PSK(theory)', '16-QAM(theory)',...
'64-QAM(theory)' ,'256-QAM(theory)','location','best')
xlabel('EbNo (dB)')
ylabel('BER')
grid on
figure(2)
semilogy(EbNo, sum_BER(:,1,2), 'b-o', EbNo, sum_BER(:,2,2), 'r-o',...
EbNo, sum_BER(:,3,2), 'g-o', EbNo, sum_BER(:,4,2), 'c-o',...
```

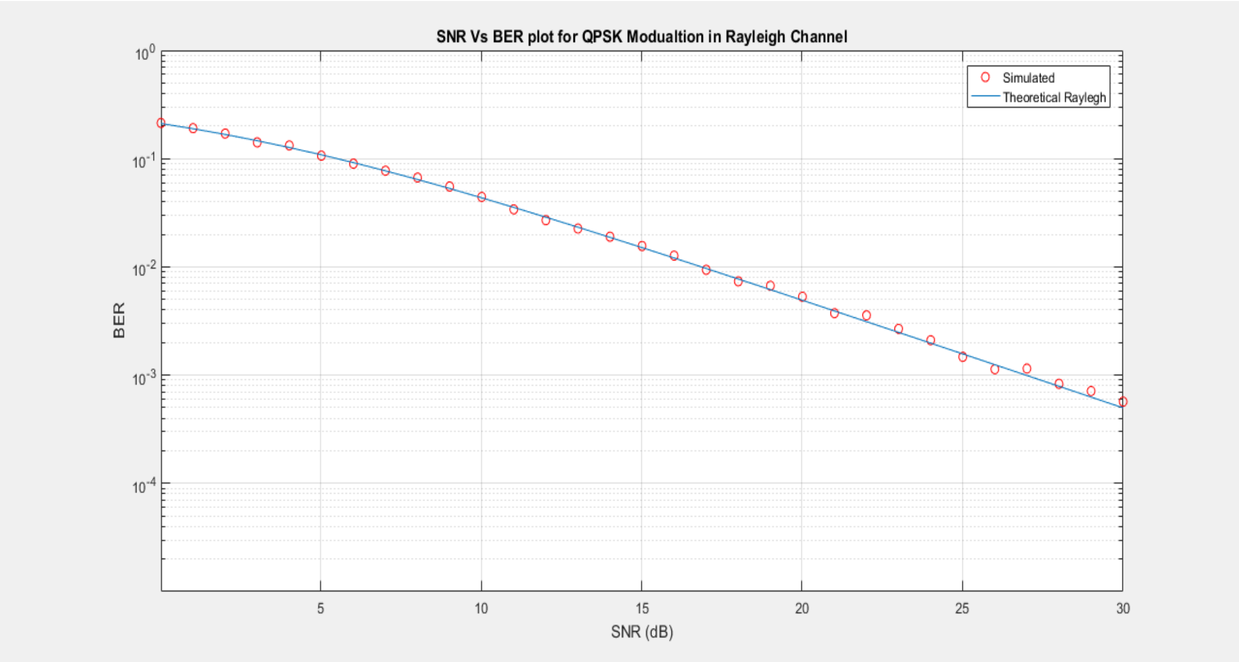
```
EbNo, sum_BER(:,5,2), 'k-o',...
EbNo, ric_ber(:,1,2), 'b-', EbNo, ric_ber(:,2,2), 'r-',...
EbNo, ric_ber(:,3,2), 'g-', EbNo,ric_ber(:,4,2), 'c-',...
EbNo, ric_ber(:,5,2), 'k-','LineWidth', 1.5)
title('Rician model (K = 0.6)')
legend('QPSK(simulated)', '8-PSK(simulated)',...
'16-QAM(simulated)', '64-QAM(simulated)' ,'256-QAM(simulated)',...
'QPSK(theory)','8-PSK(theory)',...
'16-QAM(theory)', '64-QAM(theory)' ,'256-QAM(theory)','location','best')
xlabel('EbNo (dB)')
ylabel('BER')
grid on
```

Results:

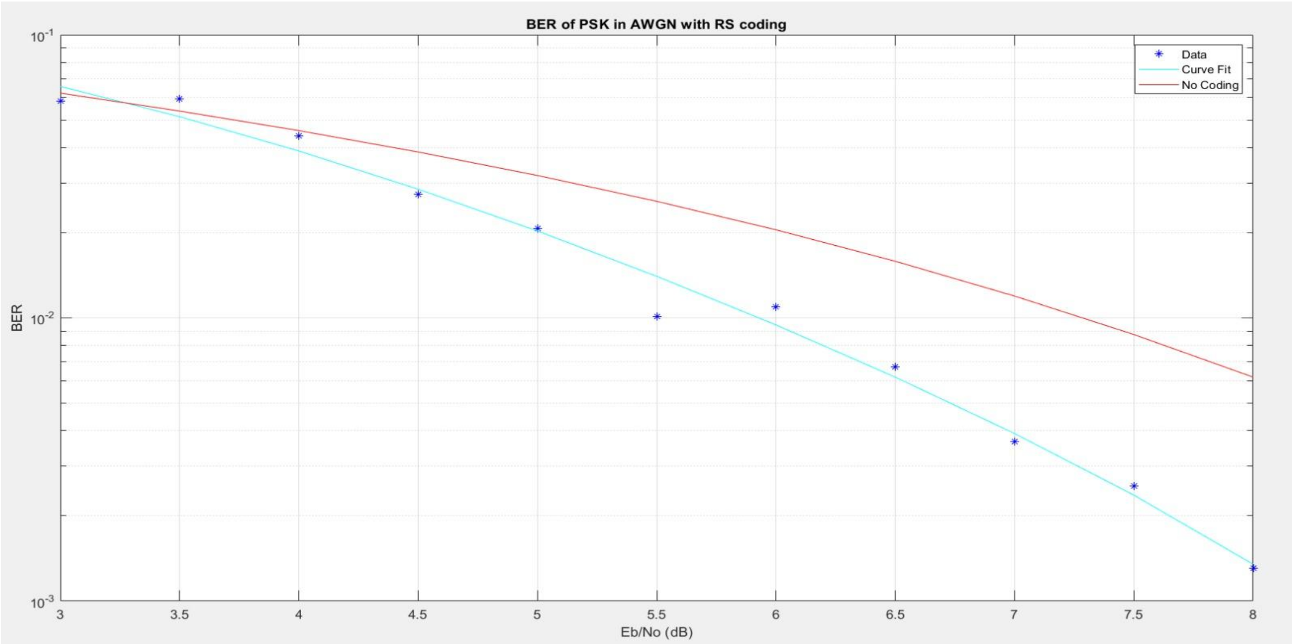
1. QPSK with OFDM on AWGN Channel:



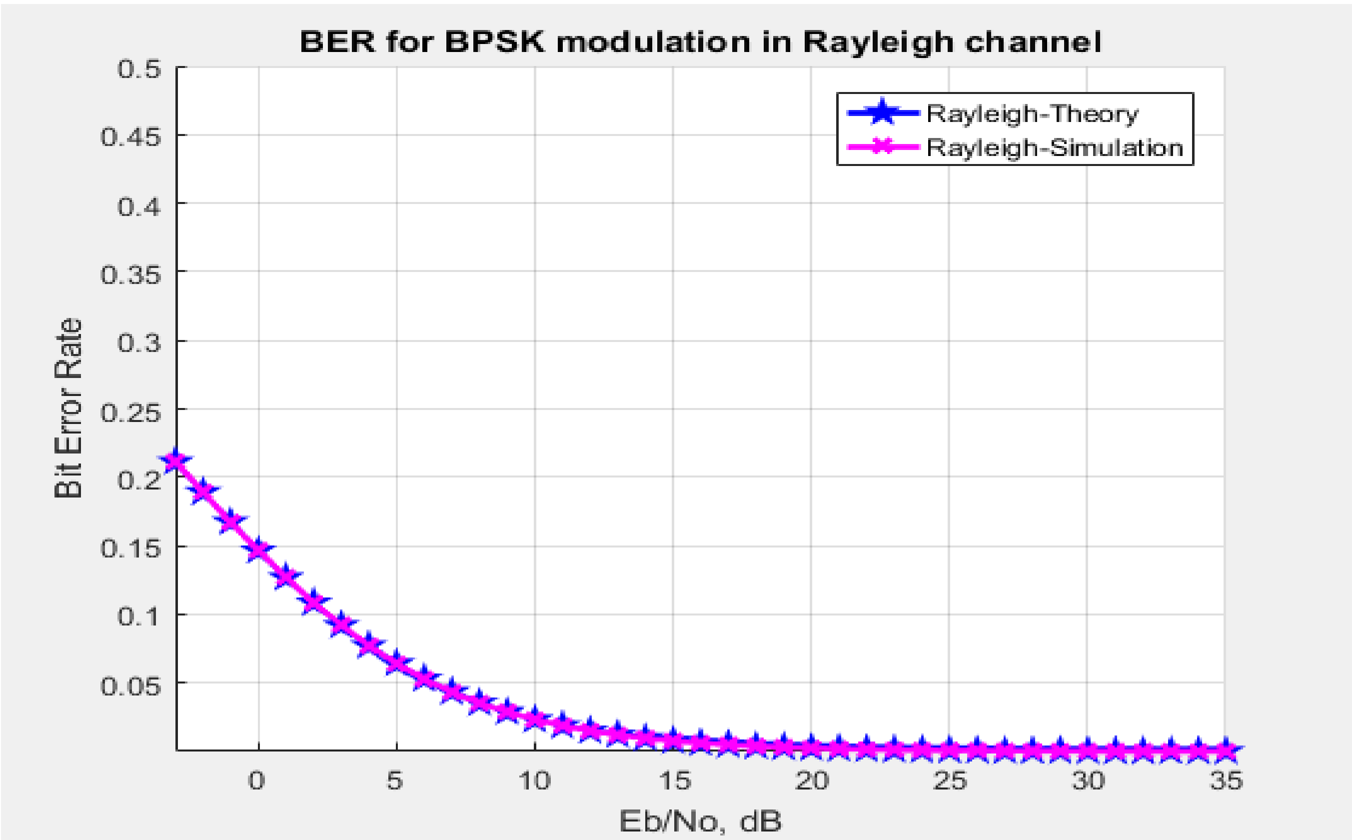
2. QPSK with Gray Coding on Rayleigh Channel:



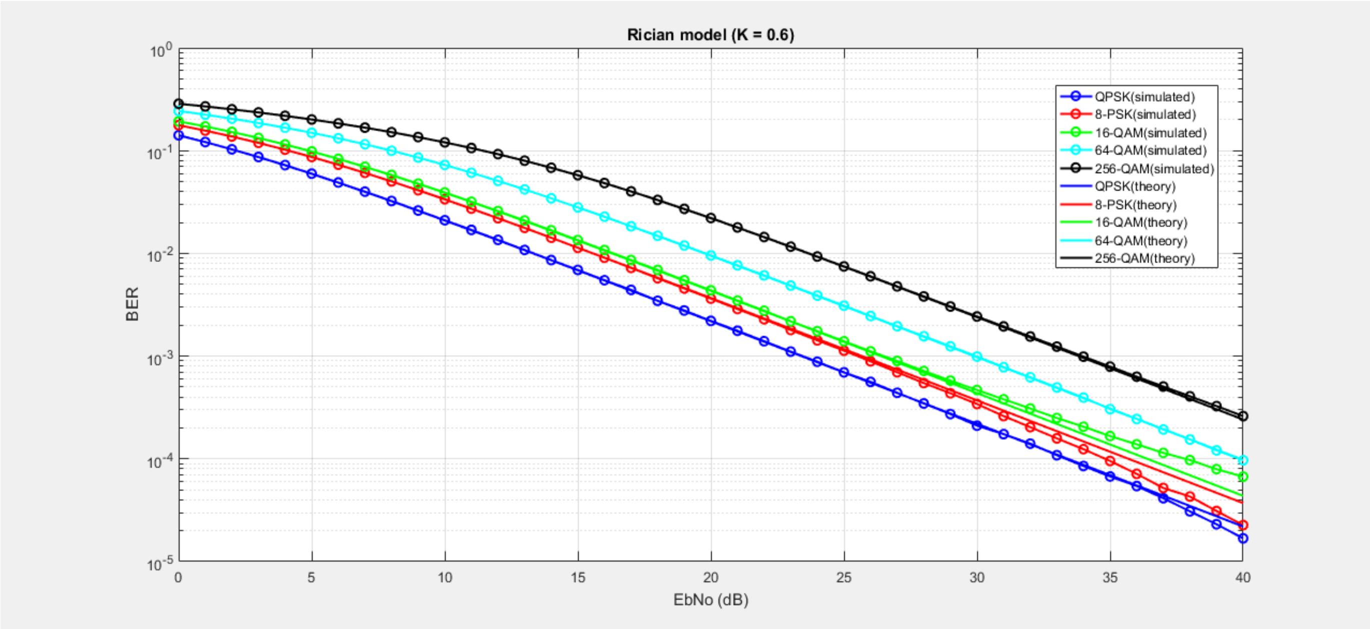
3. PSK with RS Coding on AWGN Channel:



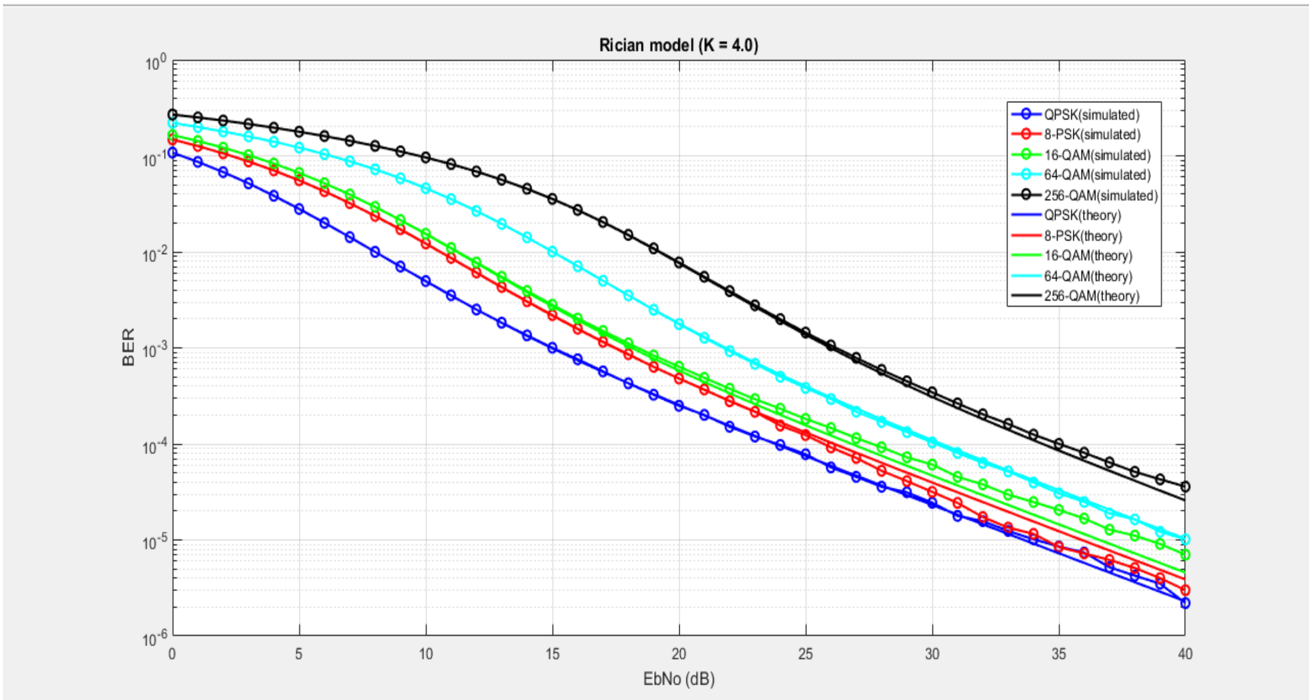
4. BPSK on Rayleigh Fading Channel:



5. M- Array PSK on Rician Fading Channel(k=0.6):



6. M- Array PSK on Rician Fading Channel(k=4.0):



Conclusion:

We know that everything which is useful to us will have some possible set of advantages and disadvantages. The above mentioned results also show that these techniques have their own set of advantages and disadvantages. We have to precisely select the suitable modulation scheme and multiplexing scheme which will satisfy the needs of application for which we are using this communication. Even though many new modulating schemes have been implemented in the recent years, we are still having some possible disadvantages when coming to practical applications. In this project I have simulated noisy channels by assuming different noise distribution functions but when this project will be implemented in real world, based on the climatic conditions and atmospheric effects the real-time results may vary.

References:

1. https://matlab.mathworks.com/
2. https://in.mathworks.com/help/comm/index.html?s_tid=CRUX_lftnav
3. https://in.mathworks.com/help/comm/examples.html?s_tid=CRUX_gn_example
4. https://in.mathworks.com/help/comm/gs/qpsk-and-ofdm-with-matlab-system-objects-1.html
5. https://in.mathworks.com/matlabcentral/fileexchange/
6. https://en.wikipedia.org/wiki
7. https://in.mathworks.com/matlabcentral/fileexchange?q=Rician+channel&s_tid=srch_dym_sug