Department of Electrical and Electronic Engineering
Faculty of Engineering

Modelling: Methods and Tools

(EEEE2055 UNUK) (FYR1 22-23)

Coursework 1 (Fourier Transforms)

Author:
George Downing

Student Number:
20273662

December 20, 2022

1 Task 1

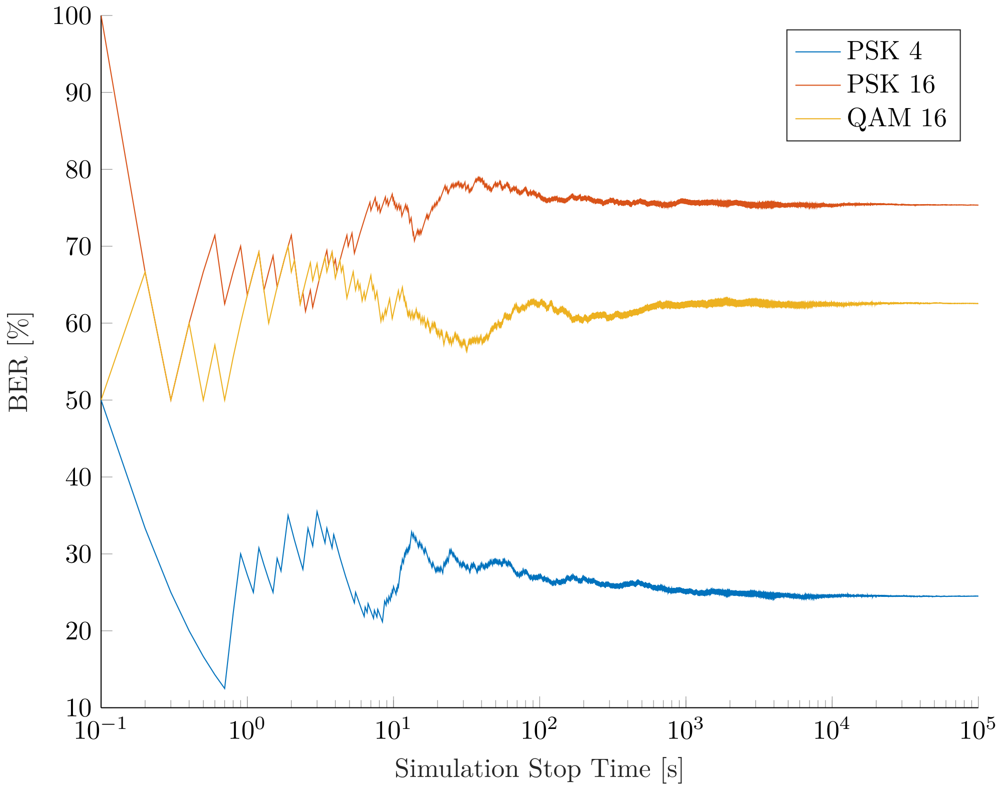

Figure 1: onvergence of BER results for QPSK, 16PSK and 16QAM modulations

Figure 1 shows the stop time vs BER for QPSK, 16-PSK and 16-QAM modulations. The results are obtained using the Matlab script as shown in A.1.

Figure 1 shows the percentage BER vs SNR for QPSK, 16-PSK and 16-QAM modulations for different stop times. The results are plotted on an x-log scale. The results are obtained using the Matlab script as shown in A.1.

The Graph shows to begin with the BER is not very precise, with lots of variation. As the simulation stop time increases, the % BER converges on a value. at around 1000 intervals, the % BER converges has converged really well. By 100000 intervals, the change in the BER is unobservable on the graph. This is because the noise model is based on a Gaussian probability distribution. As the stop time increases, the number of samples also increases, hence the noise model becomes more accurate and the BER converses on the true value.

The Graph shows PSK 4 has the lowest BER. This is because PSK 4 has the least number of symbols. This means the spacing between symbols on a constellation diagram is the largest, hence is the least susceptible to noise. The PSK 16 has the highest BER. Although it has the same number of symbols as QAM 16, due to the unit circle constellation of PSK 16, vs the lattice constellation of QAM 16, the distance between symbols on the constellation diagram is smaller, therefore PSK16 is more susceptible to noise.

2 Task 2

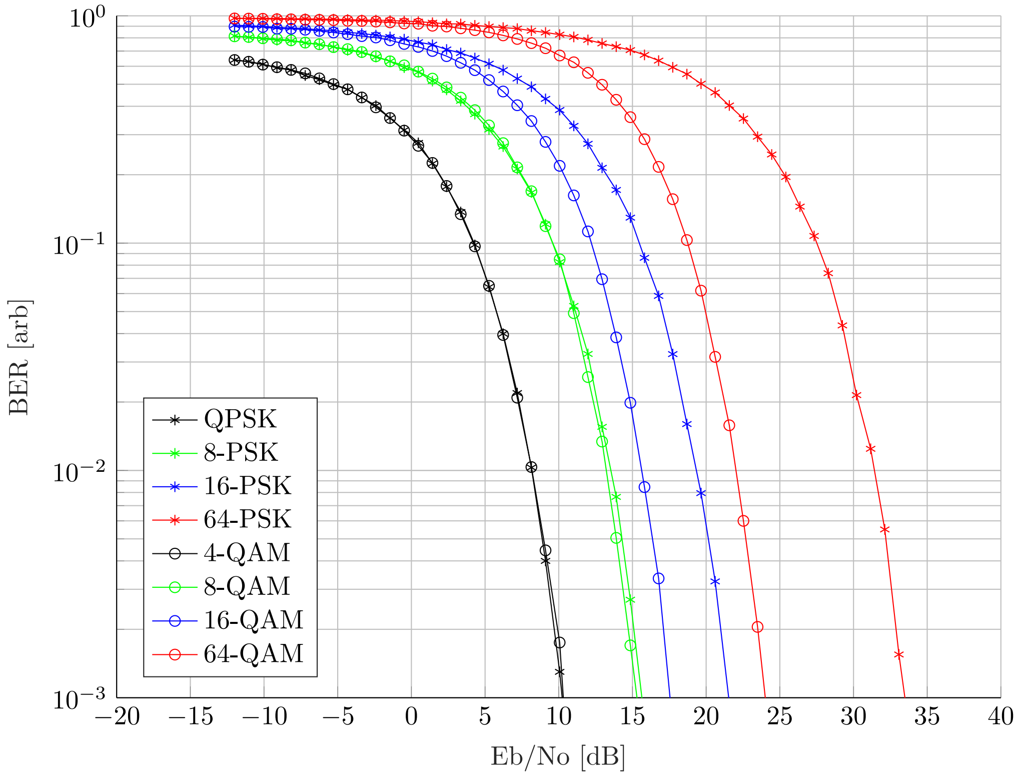

Figure 2: Comparison of BER for QPSK, 8PSK, 16PSK, 64PSK, 4QAM, 8QAM, 16QAM and 64QAM

Figure 2 shows BER for QPSK, 8PSK, 16PSK, 64PSK, 4QAM, 8QAM, 16QAM and 64QAM. The results are obtained using the Matlab script as shown in A.2. The Graph shows as the signal strength increases the bitrate error decreases. QPSK or 4PSK is identical to 4QAM. This is because the constellation is identical. This is reflected in the results. For, configurations, (m>4), the lattice constellation of QAM provides better performance than the unit circle constellation of PSK. The higher the value of m, the greater the difference. As the number of symbols increases, the signal strength must be higher to achieve the same BER. This is due to the distance between symbols being smaller and therefore less resistant to white noise.

The outlier of this graph is 8-QAM. This is because the constellation is not square. This means that the distance between symbols varies. Due to the nature of adding probabilities, the variance in distance between symbols hence, BER, means the overall BER is slightly higher than expected. This is reflected in the graph where the BER for 8-QAM is almost the same as 8-PSK, where perhaps a larger gap might be expected.

3 Task 3

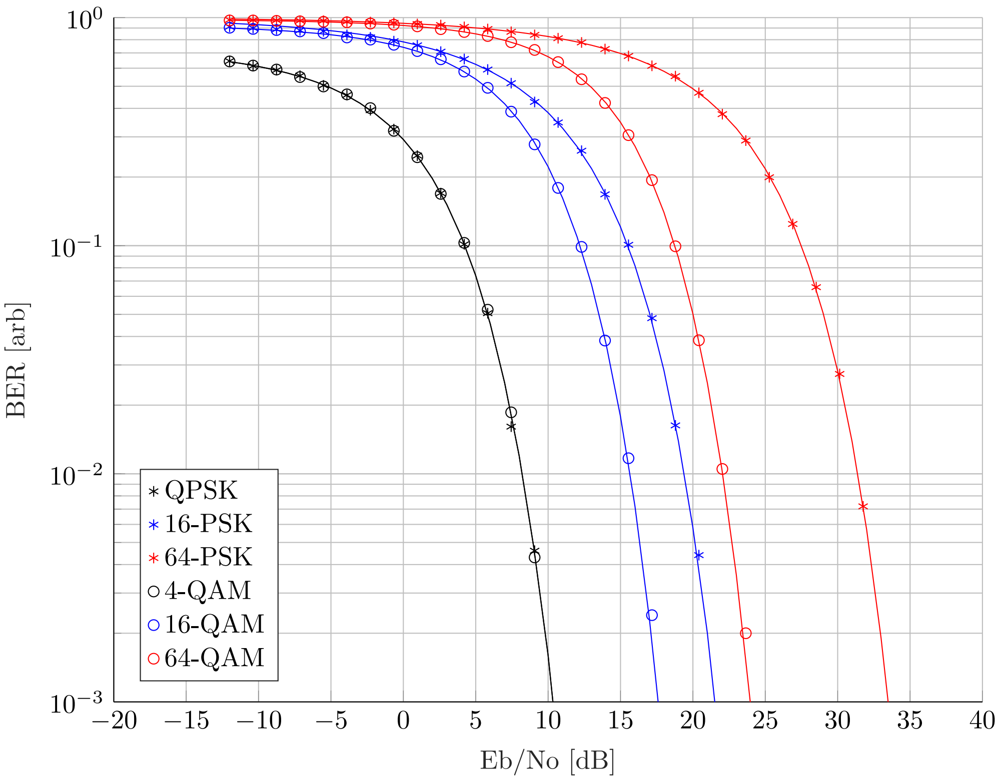

Figure 3: Comparison of BER for QPSK, 16PSK, 64PSK, 16QAM and 64QAM

$$Q(x) = \frac{1}{2} \cdot \operatorname{erf} c\left(\frac{x}{\sqrt{2 \cdot \log_2(M)}}\right) \tag{1}$$

$$\begin{aligned}
BER_{QPSK} &= 2Q\left(\sqrt{2 \cdot \frac{E_b}{N_0}}\right) - Q^2\left(\sqrt{2 \cdot \frac{E_b}{N_0}}\right) \\
BER_{MPSK}(M > 4) &= 2Q\left(\sin\left(\frac{\pi}{M}\right) \cdot \sqrt{2 \cdot \frac{E_b}{N_0}}\right) \\
BER_{MQAM} &= 1 - \left(1 - 2\left(1 - \frac{1}{\sqrt{M}}\right) \cdot Q\left(\sqrt{\frac{3 \cdot \log_2(M) \cdot \frac{E_b}{N_0}}{M - 1}}\right)\right)^2
\end{aligned} \tag{2}$$

An adaption of [1] was used to define the equations for BER as shown in eq. (2) where the Q function is defined in eq. (1).

Figure 3 shows the theoretical (continuous) vs simulated (discrete) BER for QPSK, 16PSK, 64PSK, 16QAM and 64QAM. The results are obtained using the Matlab script as shown in A.3, where the stop time was set to 2000 for the simulated results. The Graph shows the simulated results have very little variation from the theoretical results.

The equation for QAM BER is only correct for square numbers of M. A bit error usually error occurs when a symbol is mistaken for its nearest member. The distance between symbols on the constellation diagram is variable for 8-QAM. Therefore the error for each BER for each symbol would have to be calculated and averaged, which is much more complicated than for a square number of M with a constant inter-symbol distance.

4 Task 4

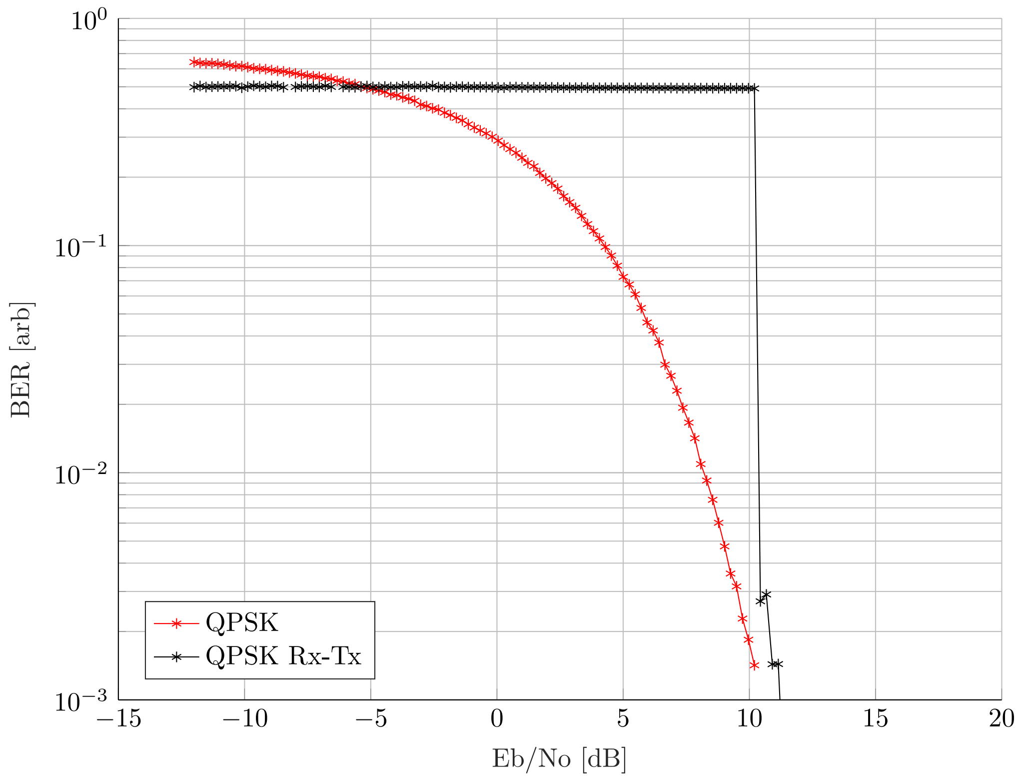

Figure 4: Comparison of BER results

Figure 4 shows Eb/N0 vs BER for a QPSK signal in the presence of AWGN noise, vs a QPSK rx/tx communication link simulating additional: time delay, frequency offset distortions and filers present in typical communication links. The results are obtained using the Matlab script as shown in A.4. The graph shows the simulated comms link is almost a perfect step whereases the simple model shows the BER decreasing smoothly as the signal strength increases. In the real world, this would make the comms link very binary, either having a connection or not.

For the comms link, the additional distortions further scramble the signal. The filters in the comms link can rebuild the signal in the presence of AWGN noise by comparing what they see to what they expect to see. However, if the noise is large enough these filters can't synchronize, and the signal becomes scrabbled. In practice, these links are either connected or not.

References

[1] Mathuranathan, "Simulate additive white Gaussian noise (AWGN) channel - GaussianWaves," GaussianWaves, Nov. 2020. [Online]. Available: https://www.gaussianwaves.com/2015/06/how-to-generate-awgn-noise-in-matlaboctave-without-using-in-built-awgn-function

Appendices

A Matlab Code

A.1 Task 1

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Matlab script to generate BER vs Simulation Stop Time plot
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear all;
close all;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                      % Define Constants
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

stop_time = 100000 % simulation stop time
Eb_No_db = 1 % Eb/No in dB
phaseOffset = 0 % phase offset in radians

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                       % Run simulation
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

m_ary = 4 % M-ary number
out1 = sim('../Simlink/PSK.slx') % run simulation

m_ary = 16 % M-ary number
out2 = sim('../Simlink/PSK.slx') % run simulation

m_ary = 16 % M-ary number
out3 = sim('../Simlink/QAM.slx') % run simulation

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                      % Create Plot
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

hold on;
plot(out1.yout{1}.Values.Time(:), out1.yout{1}.Values.Data(:, 1) * 100) %plot simulation results QPSK
plot(out2.yout{1}.Values.Time(:), out2.yout{1}.Values.Data(:, 1) * 100) %plot simulation results 16-PSK
plot(out3.yout{1}.Values.Time(:), out3.yout{1}.Values.Data(:, 1) * 100) %plot simulation results 16-QAM
hold off;

legend('PSK 4', 'PSK 16', 'QAM 16') % add legend to plot
ylabel('BER [%]') % add y-axis label
xlabel('Simulation Stop Time [s]') % add x-axis label

set(gca, 'XScale', 'log') % set x-axis to logarithmic scale

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                  % Convert to text format
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

cleanfigure; % remove unnecessary lines from plot
matlab2tikz('../Figures/fig1.tex'); % export plot to LaTeX
```

A.2 Task 2

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%         Script to run the simulation and plot BER vs Signal Strengh
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
clear all;  % Clear all variables
close all; % Close all figures

global stop_time; % Simulation time gobal to use in functions
global Eb_No_db; % Eb/No in dB global to use in functions
global phaseOffset; % Phase offset global to use in functions
global m_ary; % M-ary global to use in functions
global idx; % Index for color and line style

idx = 0; % Index for color and line style
stop_time = 2000 % Simulation time
phaseOffset = 0 % Phase offset
max_EBN = 35 % Max Eb/No in dB
resulution = 50 % Resolution of the plot

figure; % Create a new figure
hold on; % Hold the plot

BER = EbNo_BER('../Simlink/PSK.slx', 4, -12, max_EBN, resulution, '-*') % Run the simulation and plot the BER
BER = EbNo_BER('../Simlink/PSK.slx', 8, -12, max_EBN, resulution, '-o') % Run the simulation and plot the BER
BER = EbNo_BER('../Simlink/PSK.slx', 16, -12, max_EBN, resulution, '-x') % Run the simulation and plot the BER
BER = EbNo_BER('../Simlink/PSK.slx', 64, -12, max_EBN, resulution, '-+') % Run the simulation and plot the BER

BER = EbNo_BER('../Simlink/QAM.slx', 4, -12, max_EBN, resulution, '-*') % Run the simulation and plot the BER
BER = EbNo_BER('../Simlink/QAM.slx', 8, -12, max_EBN, resulution, '-o') % Run the simulation and plot the BER
BER = EbNo_BER('../Simlink/QAM.slx', 16, -12, max_EBN, resulution, '-x') % Run the simulation and plot the BER
BER = EbNo_BER('../Simlink/QAM.slx', 64, -12, max_EBN, resulution, '-+') % Run the simulation and plot the BER

hold off; % Release the plot
set(gca, 'YScale', 'log') % Set the Y axis to log scale
legend('QPSK', '8-PSK', '16-PSK', '64-PSK', '4-QAM', '8-QAM', '16-QAM', '64-QAM', Location = 'southwest') % Add a legend
xlabel('Eb/No [dB]') % Add a label to the X axis
ylabel('BER [arb]') % Add a label to the Y axis
grid on; % Add a grid to the plot
xlim([-20 40]); % Set the X axis limits
ylim([10 ^ -3 1]); % Set the Y axis limits

cleanfigure; % Clean the figure
matlab2tikz('../Figures/fig2.tex'); % Export the figure to LaTeX

function BER = EbNo_BER(model, m, start, stop, resolution, line)
    global m_ary; % M-ary global to use in functions
    global Eb_No_db; % Eb/No in dB global to use in functions
    global idx; % Index for color and line style
    m_ary = m % M-ary global to use in functions
    idx = idx + 1; % Index for color and line style

    col = {'k-*', 'g-*', 'b-*', 'r-*', 'k-o', 'g-o', 'b-o', 'r-o'}; % Colors and line styles

    Eb_N = linspace(start, stop, resolution); % Eb/No in dB

    for i = 1:length(Eb_N) % Loop over all Eb/No values
        Eb_No_db = Eb_N(i); % Eb/No in dB global to use in functions
        res = sim(model) % Run the simulation
        BER(i) = res.yout{1}.Values.Data(end, 1) % Get the BER from the simulation
    end

    plot(Eb_N(:), BER(:), char(col(idx))); % Plot the BER vs Eb/No
end
```

A.3 Task 3

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   Script to run the simulation and plot theoretical BER vs Signal Strengh
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear all;  % Clear all variables
close all; % Close all figures

global stop_time; % Simulation time global to use in funciton
global Eb_No_db; % Eb/No in dB global to use in funciton
global phaseOffset; % Phase offset global to use in funciton
```

```
global m_ary; % M-ary global to use in funciton
global idx; % Index global to use in funciton

idx = 0; % Index global to use in funciton
stop_time = 1000 % Simulation time
phaseOffset = 0 % Phase offset
max_EBN = 35 % Max Eb/No
resulution = 30 % Resulution of Eb/No

figure; % Create new figure
hold on; % Hold figure

if (1) %debug

    BER = EbNo_BER('../Simlink/PSK.slx', 4, -12, max_EBN, resulution, '-*') % Run simulation
    %BER = EbNo_BER('../Simlink/PSK.slx', 8, -12, max_EBN, resulution, '-o') % Run simulation
    BER = EbNo_BER('../Simlink/PSK.slx', 16, -12, max_EBN, resulution, '-x') % Run simulation
    BER = EbNo_BER('../Simlink/PSK.slx', 64, -12, max_EBN, resulution, '-+') % Run simulation

    BER = EbNo_BER('../Simlink/QAM.slx', 4, -12, max_EBN, resulution, '-*') % Run simulation
    %BER = EbNo_BER('../Simlink/QAM.slx', 8, -12, max_EBN, resulution, '-o') % Run simulation
    BER = EbNo_BER('../Simlink/QAM.slx', 16, -12, max_EBN, resulution, '-x') % Run simulation
    BER = EbNo_BER('../Simlink/QAM.slx', 64, -12, max_EBN, resulution, '-+') % Run simulation

end

%BPSK BER
EbN0dB = -12:1:40; % Eb/N0 range in dB
EbN0lin = 10 .^ (EbN0dB / 10) % Eb/N0 range in linear scale
colours = {'k-', 'k-', 'b-', 'b-', 'r-', 'r-'}; % Colours for the plot
index = 0 % Index for the colours

index = index + 1; % Index for the colours
M = [4 16 64]; % M-ary

for i = M,
    k = log2(i) % Bits per symbol
    y_s = k * EbN0lin % gamma_s

    syms x % Symbol for the Q function
    Q(x) = (1/2) * erfc(x / sqrt(2 * k)); % Q function

    if i == 4 % QPSK
        berErr = 2 * Q(sqrt(2 * EbN0lin)) - Q(sqrt(2 * EbN0lin)) .^ 2 % BER for QPSK
    else
        berErr = 2 * Q(sin(pi / i) * sqrt(2 * y_s)); % BER for M-PSK
    end

    berErr2 = 1 - (1 - (2 * (1 - 1 / sqrt(i))) * Q(sqrt((3 * y_s) / (i - 1)))) .^ 2 % BER for M-QAM

    plotHandle = plot(EbN0dB, (berErr), char(colours(index))); % Plot PSK BER
    index = index +1; % Index for the colours

    plotHandle2 = plot(EbN0dB, (berErr2), char(colours(index))); % Plot QAM BER
    index = index +1; % Index for the colours
end

set(gca, 'YScale', 'log') % Set Y axis to log scale
legend('QPSK', '16-PSK', '64-PSK', '4-QAM', '16-QAM', '64-QAM', Location = 'southwest') % Legend
xlabel('Eb/No [dB]') % X label
ylabel('BER [arb]') % Y label
xlim([-20 40]); % X axis limits
ylim([10 ^ -3 1]); % Y axis limits
hold off % Hold figure
grid on; % Grid on

cleanfigure; % Clean figure
matlab2tikz('../Figures/fig3.tex'); % Export figure to LaTeX

function BER = EbNo_BER(model, m, start, stop, resolution, line)

    global m_ary; % M-ary global to use in funciton
    global Eb_No_db; % Eb/No in dB global to use in funciton
```

```
    global idx; % Index global to use in funciton

    m_ary = m % M-ary global to use in funciton
    idx = idx + 1; % Index global to use in funciton

    col = {'k*', 'b*', 'r*', 'ko', 'bo', 'ro'}; % Colours for the plot

    Eb_N = linspace(start, stop, resolution); % Eb/No range

    for i = 1:length(Eb_N) % Loop through Eb/No
        Eb_No_db = Eb_N(i); % Eb/No in dB global to use in funciton
        res = sim(model) % Run simulation
        BER(i) = res.yout{1}.Values.Data(end, 1) % Get BER
    end

    plot(Eb_N(:), BER(:), char(col(idx))); % Plot BER

end
```

A.4 Task 4

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                        PSK rx tx vs PSK sim
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear all;  % clear all variables
close all; % close all figures

idx = 0; % index for the number of simulations
stop_time = 5000 % simulation time
phaseOffset = 0 % phase offset
m_ary = 4 % m-ary PSK
max_EBN = 35 % maximum Eb/No
resulution = 200 % number of points in the plot

figure; % figure
hold on; % hold on for multiple plots

Eb_N = linspace(-12, 35, resolution); % Eb/No vector

for i = 1:length(Eb_N) % loop over Eb/No
    Eb_No_db = Eb_N(i); % set Eb/No
    res = sim('../Simlink/PSK.slx'); % run simulation
    BER(i) = res.yout{1}.Values.Data(end, 1) % save BER
end

plot(Eb_N(:), BER(:), 'r-*'); % plot BER vs Eb/No

for i = 1:length(Eb_N) % loop over Eb/No
    Eb_No_db = Eb_N(i); % set Eb/No
    res = sim('commqpsktxrx.slx'); % run simulation
    BER(i) = out.Data(end, 1) % save BER
end

plot(Eb_N(:), BER(:), 'k-*'); % plot BER vs Eb/No

hold off; % hold off for multiple plots
set(gca, 'YScale', 'log') % set y-axis to log scale
legend('QPSK', 'QPSK Rx-Tx', Location = 'southwest') % legend
xlabel('Eb/No [dB]') % x-axis label
ylabel('BER [arb]') % y-axis label
grid on; % grid on
xlim([-15 20]); % x-axis limits
ylim([10 ^ -3 1]); % y-axis limits

cleanfigure; % clean figure
matlab2tikz('../Figures/fig4.tex'); % save figure as tex file
```

B LTSpice Code