Table of Contents

Symbol Error Rate.

Calculate the SER for different modulation types and channels, and compare with theoretical values.

```
clc; clear; close all;
```

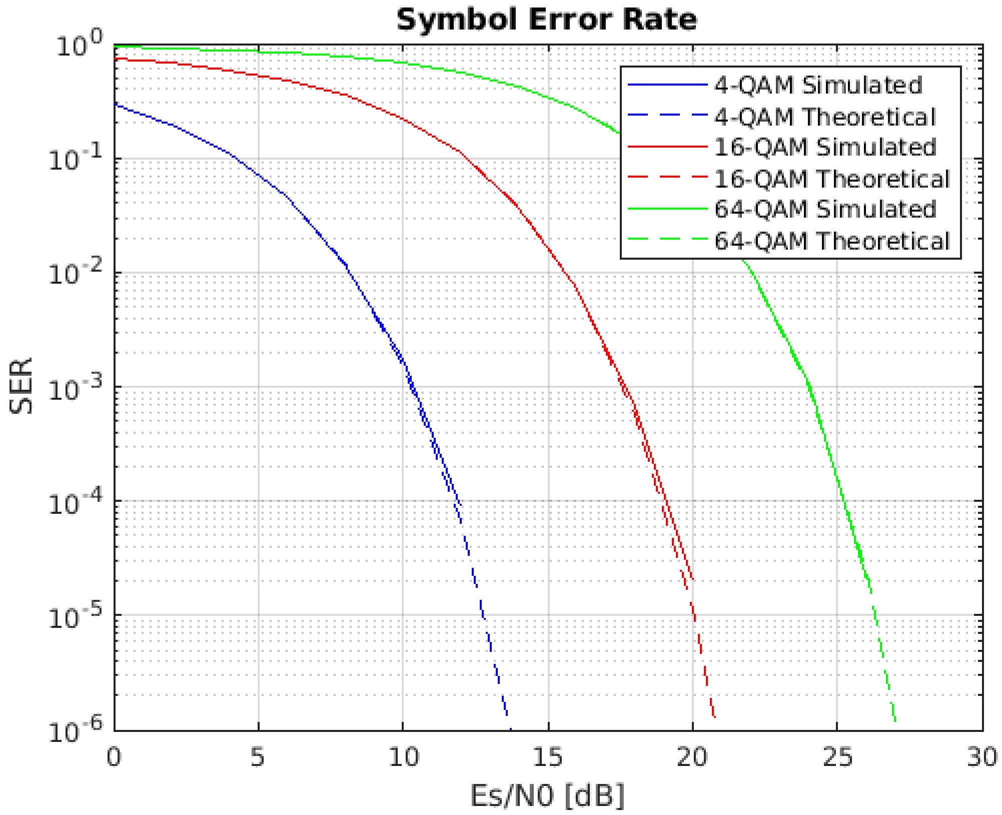
Parameters

```
sample_qtty = 1e5;                  % Number of samples.
EsNo_dB = 0:2:40;                   % EsNo.
mod_type = "QAM";                   % Modulation type.
M = [4,16,64];                      % Number of symbols.
channel_type = ChannelTypes.AWGN;   % Channel type.
PlosPnlos_dB = 20;                  % Ratio between power of LOS and NLOS.
L = 10;                             % Oversampling factor
beta = 0.8;                         % Roll-off SRRC

% Taps for FIR filters. Note: The bigger this value, the closer the
% theoretical and simulated SERs are.
nTaps = 50;
```

Calculations

```
colors = ["b", "r", "g", "c", "m", "k"];    % Colors for plotting
legendString = cell(1, 2*length(M));        % For legend in plot, as "16-QAM"

ser = zeros(1, length(EsNo_dB));            % Symbol error rate

for m=1:1:length(M)
    d = randi([0, M(m)-1], 1, sample_qtty); % Input symbols

    % Modulator
    [u, constellation] = Modulator.modulate(d, mod_type, M(m));
    v = Modulator.upsample(u, L);
    [s, ~, delay_tx] = Modulator.pulse_shaping_srrc(v, beta, L, nTaps);

    for i=1:1:length(EsNo_dB)
        % Channel
        switch channel_type
            case ChannelTypes.AWGN
                [r, h_c] = Channel.add_awgn_noise(s, EsNo_dB(i), L);
            case ChannelTypes.Rayleigh
                [r, h_c] = Channel.add_rayleigh_noise(s, EsNo_dB(i), L);
            case ChannelTypes.Ricean
```

```
                [r, h_c] = Channel.add_ricean_noise(s, EsNo_dB(i), L,
 PlosPnlos_dB);
        end

        % Demodulator
        r = Demodulator.flat_fading_equalizer(r, h_c);
        [v_r, ~, delay_rx] = Demodulator.pulse_filter_srrc(r, beta, L, nTaps);
        u_r = Demodulator.downsample(v_r, L, delay_tx + delay_rx);
        d_r = Demodulator.demodulate(u_r, mod_type, M(m), constellation);

        ser(i) = sum(d~=d_r)/sample_qtty;
    end

    switch channel_type
        case ChannelTypes.AWGN
            ser_theory = Theory.ser_AWGN(mod_type, M(m), EsNo_dB);
        case ChannelTypes.Rayleigh
            ser_theory = Theory.ser_rayleigh(mod_type, M(m), EsNo_dB);
        case ChannelTypes.Ricean
            ser_theory = Theory.ser_ricean(mod_type, M(m), EsNo_dB,
 PlosPnlos_dB);
    end

    semilogy(EsNo_dB, ser, Color=colors(m), LineStyle="-"); hold on;
    semilogy(EsNo_dB, ser_theory, Color=colors(m), LineStyle="--"); hold on;

    legendString{2*m-1} = strcat(num2str(M(m)), "-", mod_type, " Simulated");
    legendString{2*m}   = strcat(num2str(M(m)), "-", mod_type, "
 Theoretical");
end

grid on;
legend(legendString);
xlabel("Es/N0 [dB]");
ylabel("SER");
title("Symbol Error Rate");
ylim([1e-6, 1]);
```



Published with MATLAB® R2022b