EE3320 - WIRELESS COMMUNICATIONS

ASSIGNMENT - 4

EE18BTECH11017

---

1 BER Simulations for BPSK Modulation

1.1 BSPK Modulation

For BPSK modulation, I have mapped binary data 0 and 1 to -1 and 1 respectively

1.2 Rayleigh Fading

Assuming the Coherent Detection, that is channel impulse response is known at the reciever.We have

$$y = hx + n \implies \hat{y} = \frac{y}{h} = x + z \tag{1}$$

The h follows rayleigh distribution and X,Y follows $N(0, \sigma^2)$

$$h = X + jY = \sqrt{X^2 + Y^2} \tag{2}$$

Then,

$$E[h^2] = E[X^2 + Y^2] = E[X^2] + E[Y^2] = 2\sigma^2 = 1 = E_s$$

$$\implies \sigma^2 = \frac{1}{2}$$

So I considered X,Y to be $N(0, 1/2)$

1.3 Rician Fading

For Rician it is similar to Rayleigh with $X \sim N(s, \sigma^2)$ and $Y \sim N(0, \sigma^2)$. Then,

$$K = \frac{s^2}{2\sigma^2} \tag{3}$$

And,

$$E[h^2] = E[X^2 + Y^2] = E[X^2] + E[Y^2] = (s^2 + \sigma^2) + \sigma^2 = s^2 + 2\sigma^2$$
$$E_s = P_{tot} = E[h^2] = s^2 + 2\sigma^2 = 1 \tag{4}$$

Using (3) and (4) we get,

$$s = \frac{K}{K+1}$$
$$\sigma = \frac{1}{\sqrt{2(K+1)}}$$

K is varied such that we get different curves.

1.4 Nakagami-m Fading

Used *gamrnd* function in matlab to generate Nakagami-m Coefficients with,

$$E[X^2] = \Omega = 1$$

And m is varied to get different curves.

2 BER Simulations for 16-QAM Modulation

2.1 16-QAM Modulation

Used Inbuilt Functions in *5G - ToolBox* to modulate and demodulate in 16-QAM.

Also,remaining parameters are set same as above.

3 Code

```
% Params
% Input bits
N = 1e5;

%=========================================================================
%%%%%%%%%%%%%%%%%%%%%%%%% BPSK MODULATION %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


%%%%%%%%%%%%%%%%%%%% 1. BER OF AWGN %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


% Step 1: Generating random binary data
x = randn(1,N) >= 0;

% Step2 : Map the data to the BPSK signal constellation
```

```
% This for loop maps 0 to -1 and 1 to +1 for BPSK
x_BPSK = zeros(1,N);
for i = 1:N
    if x(i) == 0
        x_BPSK(i) = -1;
    else
        x_BPSK(i) = 1;
    end
end

SNR_all = -5:1:10;
BER_all = zeros(1,length(SNR_all));

% Step 3:
% Multiply the random fading coefficient and add AWGN noise to this signal.
% Vary the noise variance to have SNR range between [-5dB, 10 dB].
j = 1;
% Generating AWGN
for SNR = -5:1:10

    % Adding noise to transmitted signal without fading
    sigma = sqrt(1/(2*(10^(SNR/10))));
    noise = sigma*randn(1,N);
    rec = x_BPSK + noise;

    % Step 4:
    % Setting the threshold and demodulating
    threshold = 0;
    rec_dec = (rec >= threshold);
    BER_count = sum(xor(x,rec_dec));
    BER_all(j) = BER_count/length(x);
    j = j + 1;
end

% Step4 : plotting
figure(1)
subplot(2,2,1)
plot(SNR_all,10*log10(BER_all),'lineWidth',1.5);
xlabel('SNR per bit (E_b/N_o) dB');
ylabel('Bit Error Rate(BER) in dB');
title(' BER vs (E_b/N_o) for BPSK in AWGN channel')
grid on;



%%%%%%%%%%%%%%%%%%%%%%% 2.BER OF RAYLEIGH FADING %%%%%%%%%%%%%%%%%%%%

BER_Rayleigh = zeros(1,length(SNR_all));
ind = 1;
for SNR = -5:1:10
    % Sigma for noise
    sigma = sqrt(1/(2*(10^(SNR/10))));

    % Rayleigh fading
    h = (1/sqrt(2))*(randn(1,N) + 1j*randn(1,N));

    % AWGN Noise
    noise = sigma*(randn(1,N)+ 1j*randn(1,N));
```

```
    % Received Signal
    rec_Rayleigh = h.*x_BPSK + noise;

    % Assuming the coherence
    recieved_Rayleigh = rec_Rayleigh./h;

    %%% THRESHOLD BASED DETECTION %%%%
    rec_rayleigh_dec = recieved_Rayleigh >= 0;

    % Calculating BER for a SNR
    BER_Rayleigh(ind) = sum(xor(x,rec_rayleigh_dec))/N;
    ind  = ind + 1;
end

subplot(2,2,2)
plot(SNR_all,10*log10(BER_Rayleigh),'lineWidth',1.5,'color','r');
xlabel('SNR per bit (E_b/N_o) dB');
ylabel('Bit Error Rate(BER) in dB');
title(' BER vs (E_b/N_o) for BPSK in Rayleigh fading channel')
grid on;


%%%%%%%%%%%%%%%%%%%% 3.BER OF RICIAN FADING %%%%%%%%%%%%%%%%%%%%%%%%%

% Same steps are followed as above instead 'h' is changed here
% k PARAMETER FOR RICIAN
colorstr = {'b','r','k','g','m','c'};
colorind = 1;
legendInfo = {0,0,0,0,0,0};
for K = [1 5 10 15 20 25];
    s = K/(K+1);
    sigma = 1/sqrt(2*(K+1));
    BER_Rician = zeros(1,length(SNR_all));
    ind = 1;
    for SNR = -5:1:10
        noise_Sigma = sqrt(1/(2*(10^(SNR/10))));

        % Rician fading coefficients
        h = ((sigma*randn(1,N)+s) + 1j*(sigma*randn(1,N)));
        noise = noise_Sigma*rand(1,N);

        % Assuming coherence
        rec_Rician = h.*x_BPSK + noise;
        recieved_Rician = rec_Rician./h;
        rec_rician_dec = recieved_Rician >= 0;
        BER_Rician(ind) = sum(xor(x,rec_rician_dec))/N;
        ind  = ind + 1;
    end
    subplot(2,2,3)
    plot(SNR_all,10*log10(BER_Rician),'lineWidth',1.5,'color',colorstr{colorind});
    legendInfo{colorind} = ['K = ' num2str(K)];
    colorind = colorind + 1;
    hold on;
end
% legend(legendInfo(1,end));
legend(legendInfo);
xlabel('SNR per bit (E_b/N_o) dB');
```

```
ylabel('Bit Error Rate(BER) in dB');
title(' BER vs (E_b/N_o) for BPSK in Rician fading channel')
grid on;
hold off;

%%%%%%%%%%%%%%%%%%%% 4. BER OF NAKAGAMI-m %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


colorstr = {'b','r','k','g','m','c'};
colorind = 1;
legendInfo = {0,0,0,0,0,0};
omega = 1;
for m = [0.5 1 1.5 2 2.5 3]
    h_nakagami = [sqrt(gamrnd(m,omega./m,1,N))];
    ind = 1;
    BER_nakagami = zeros(1,length(SNR_all));
    for SNR = -5:1:10
        noise_Sigma = sqrt(1/(2*(10^(SNR/10))));
        noise = noise_Sigma*rand(1,N);
        rec_nakagami = h_nakagami.*x_BPSK + noise;

        recieved_nakagami = rec_nakagami./h_nakagami;
        rec_nakagami_dec = recieved_nakagami >= 0;
        BER_nakagami(ind) = sum(xor(x,rec_nakagami_dec))/N;
        ind  = ind + 1;
    end
    subplot(2,2,4)
    plot(SNR_all,10*log10(BER_nakagami),'lineWidth',1.5,'color',colorstr{colorind});
    legendInfo{colorind} = ['m = ' num2str(m) ' \Omega = 1'];
    colorind = colorind + 1;
    hold on;

end
legend(legendInfo);
xlabel('SNR per bit (E_b/N_o) dB');
ylabel('Bit Error Rate(BER) in dB');
title(' BER vs (E_b/N_o) for BPSK in Nakagami-m fading channel')
grid on;
hold off;




%==========================================================================
%%%%%%%%%%%%%%%%%%%%%%%%%% 16-QAM MODULATION %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

x_16QAM = nrSymbolModulate(x','16QAM');

BER_all_16QAM = zeros(1,length(SNR_all));

j = 1;
for SNR = -5:1:10

    % Adding noise to transmitted signal without fading
    sigma = sqrt(1/(2*(10^(SNR/10))));
    noise = sigma*randn(1,length(x_16QAM));
    rec = x_16QAM' + noise;
```

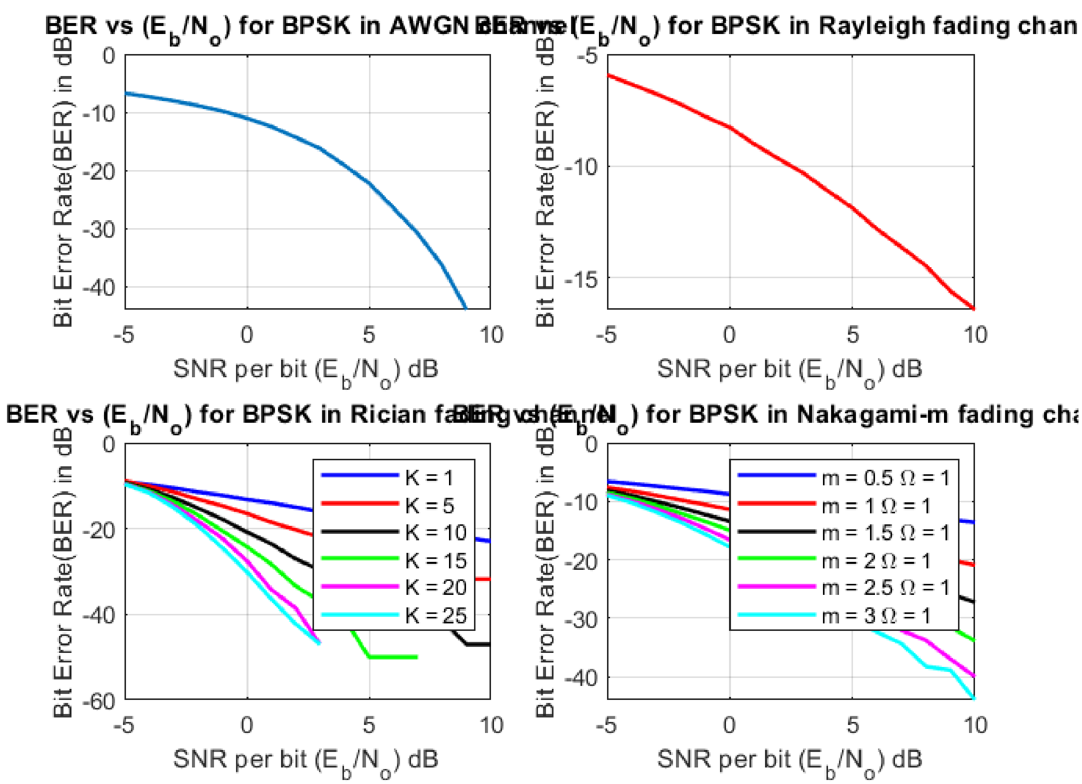
```
    % Step 4:
    % Setting the threshold and demodulating
    rec_dec = nrSymbolDemodulate(rec','16QAM','DecisionType','Hard');
    BER_count = sum(xor(x,rec_dec'));
    BER_all_16QAM(j) = BER_count/length(x);
    j = j + 1;
end

figure(2)
subplot(2,2,1)
plot(SNR_all,10*log10(BER_all_16QAM),'lineWidth',1.5);
xlabel('SNR per bit (E_b/N_o) dB');
ylabel('Bit Error Rate(BER) in dB');
title(' BER vs (E_b/N_o) for 16QAM in AWGN channel')
grid on;

%%%%%%%%%%%%%%%%%%%%%%%% 2.BER OF RAYLEIGH FADING %%%%%%%%%%%%%%%%%%%%%

BER_Rayleigh_16QAM = zeros(1,length(SNR_all));
ind = 1;
for SNR = -5:1:10
    % Sigma for noise
    sigma = sqrt(1/(2*(10^(SNR/10))));

    N_qam = length(x_16QAM);
    % Rayleigh fading
    h = (1/sqrt(2))*(randn(1,N_qam) + 1j*randn(1,N_qam));

    % AWGN Noise
    noise = sigma*(randn(1,N_qam)+ 1j*randn(1,N_qam));

    % Received Signal
    rec_Rayleigh = h.*x_16QAM' + noise;

    % Assuming the coherence
    recieved_Rayleigh = rec_Rayleigh./h;

    %%% THRESHOLD BASED DETECTION %%%%
    rec_rayleigh_dec = nrSymbolDemodulate(recieved_Rayleigh','16QAM','DecisionType','Hard');

    % Calculating BER for a SNR
    BER_Rayleigh_16QAM(ind) = sum(xor(x,rec_rayleigh_dec'))/N;
    ind  = ind + 1;
end

figure(2);
subplot(2,2,2)
plot(SNR_all,10*log10(BER_Rayleigh_16QAM),'lineWidth',1.5,'color','r');
xlabel('SNR per bit (E_b/N_o) dB');
ylabel('Bit Error Rate(BER) in dB');
title(' BER vs (E_b/N_o) for 16QAM in Rayleigh fading channel')
grid on;


%%%%%%%%%%%%%%%%%%%% 3.BER OF RICIAN FADING %%%%%%%%%%%%%%%%%%%%%%%%%%

% Same steps are followed as above instead 'h' is changed here
% k PARAMETER FOR RICIAN
```

```
colorstr = {'b','r','k','g','m','c'};
colorind = 1;
legendInfo = {0,0,0,0,0,0};
for K = [1 5 10 15 20 25];
    N_qam = length(x_16QAM);
    s = K/(K+1);
    sigma = 1/sqrt(2*(K+1));
    BER_Rician_16QAM = zeros(1,length(SNR_all));
    ind = 1;
    for SNR = -5:1:10
        noise_Sigma = sqrt(1/(2*(10^(SNR/10))));

        % Rician fading coefficients
        h = ((sigma*randn(1,N_qam)+s) + 1j*(sigma*randn(1,N_qam)));
        noise = noise_Sigma*rand(1,N_qam);

        % Assuming coherence
        rec_Rician = h.*x_16QAM' + noise;
        recieved_Rician = rec_Rician./h;
        rec_rician_dec = nrSymbolDemodulate(recieved_Rician','16QAM','DecisionType','Hard');
        BER_Rician_16QAM(ind) = sum(xor(x,rec_rician_dec'))/N;
        ind  = ind + 1;
    end
    figure(2);
    subplot(2,2,3)
    plot(SNR_all,10*log10(BER_Rician_16QAM),'lineWidth',1.5,'color',colorstr{colorind});
    legendInfo{colorind} = ['K = ' num2str(K)];
    colorind = colorind + 1;
    hold on;
end
% legend(legendInfo(1,end));
legend(legendInfo);
xlabel('SNR per bit (E_b/N_o) dB');
ylabel('Bit Error Rate(BER) in dB');
title(' BER vs (E_b/N_o) for 16QAM in Rician fading channel')
grid on;
hold off;


%%%%%%%%%%%%%%%%%%%% 4. BER OF NAKAGAMI-m %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


colorstr = {'b','r','k','g','m','c'};
colorind = 1;
legendInfo = {0,0,0,0,0,0};
omega = 1;
N_qam = length(x_16QAM);
for m = [0.5 1 1.5 2 2.5 3]
    h_nakagami = [sqrt(gamrnd(m,omega./m,1,N_qam))];
    ind = 1;
    BER_nakagami_16QAM = zeros(1,length(SNR_all));
    for SNR = -5:1:10
        noise_Sigma = sqrt(1/(2*(10^(SNR/10))));
        noise = noise_Sigma*rand(1,N_qam);
        rec_nakagami = h_nakagami.*x_16QAM' + noise;

        recieved_nakagami = rec_nakagami./h_nakagami;
        rec_nakagami_dec = nrSymbolDemodulate(recieved_nakagami','16QAM','DecisionType','Hard');
```

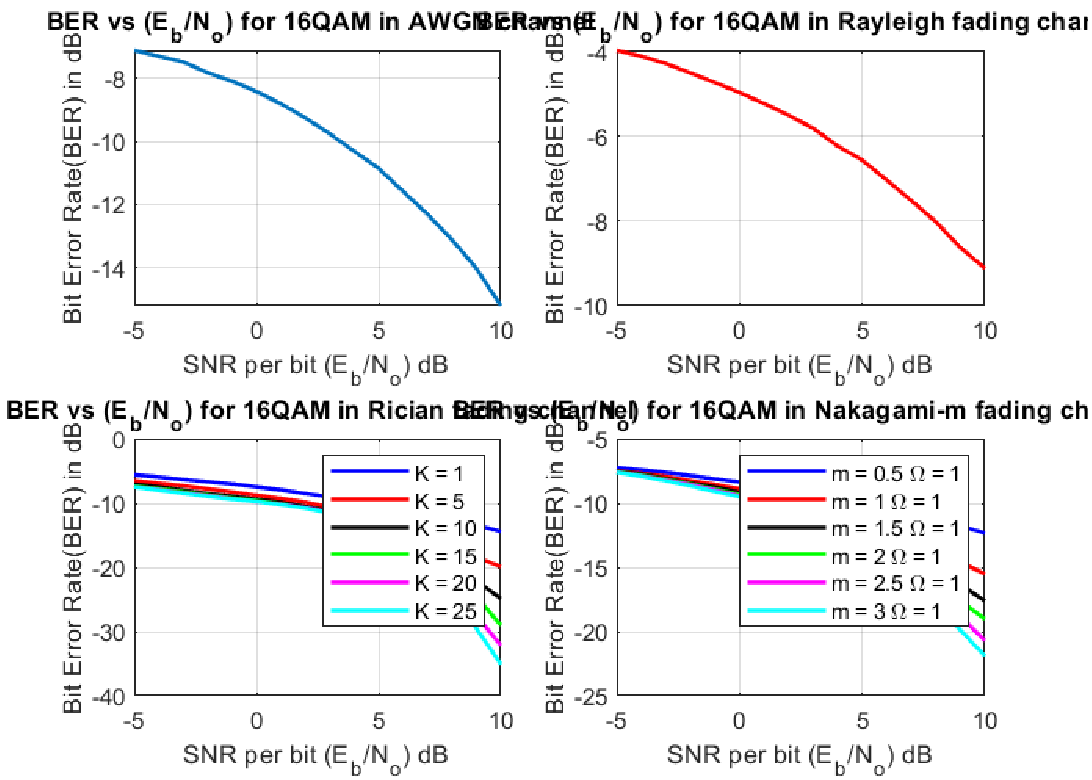
```
        BER_nakagami_16QAM(ind) = sum(xor(x,rec_nakagami_dec'))/N;
        ind  = ind + 1;
    end
    subplot(2,2,4)
    plot(SNR_all,10*log10(BER_nakagami_16QAM),'lineWidth',1.5,'color',colorstr{colorind});
    legendInfo{colorind} = ['m = ' num2str(m) ' \Omega = 1'];
    colorind = colorind + 1;
    hold on;

end
legend(legendInfo);
xlabel('SNR per bit (E_b/N_o) dB');
ylabel('Bit Error Rate(BER) in dB');
title(' BER vs (E_b/N_o) for 16QAM in Nakagami-m fading channel')
grid on;
hold off;
```

4 Plots

4.1 For BPSK



4.2 For 16 QAM



5 Observations

- For AWGN channel without fading has lowest BER followed by Rician Fading.
- As K increases BER decreases for Rician Fading
- As m increases BER decreases for Nakagami Fading
- Rayleigh Fading has the highest BER's
- For Low SNR's BER for 16-QAM is low and for high SNR's BER for 16-QAM is high compared to BPSK
- For high SNR's Rician has lowest BER's for both BPSK and 16-QAM