Faculty of Engineering
Cairo University

Cairo University

Faculty of Engineering

Electronics and Electrical Communication Department

Communication

Project 1

Name: Ahmed Mokhtar Mahfouz Emam

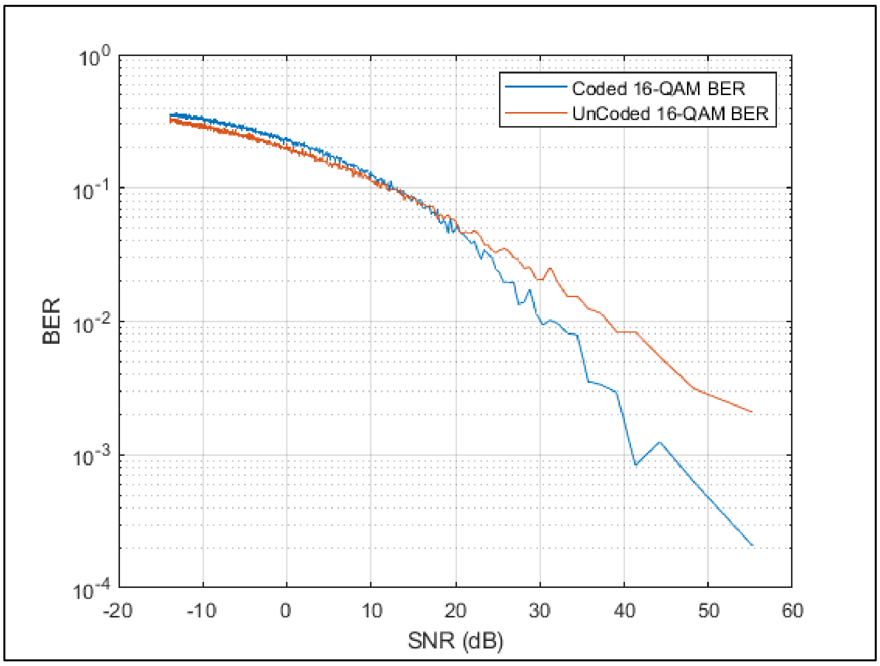
Sec: 1

BN: 30

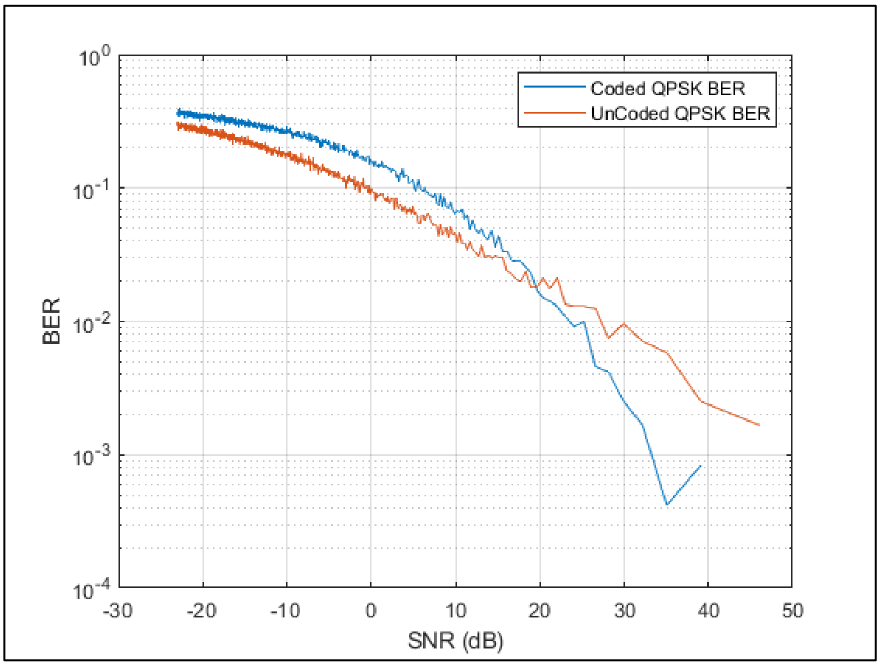
Email: ahmed.mokhtar5483@gmail.com

Part (1): Single Carrier:

1) 16-QAM:



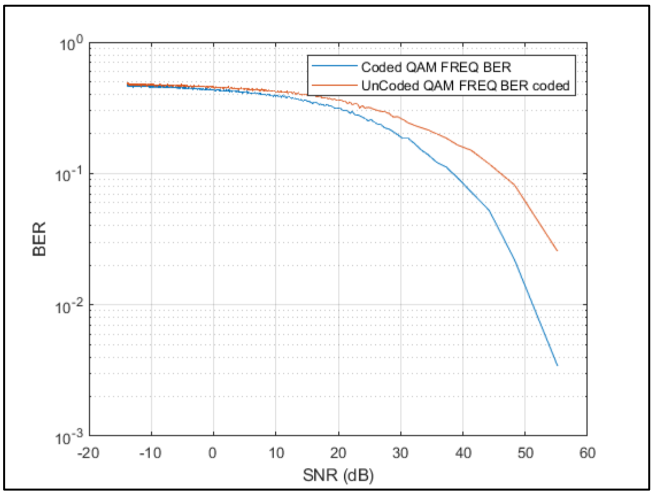
2) QPSK

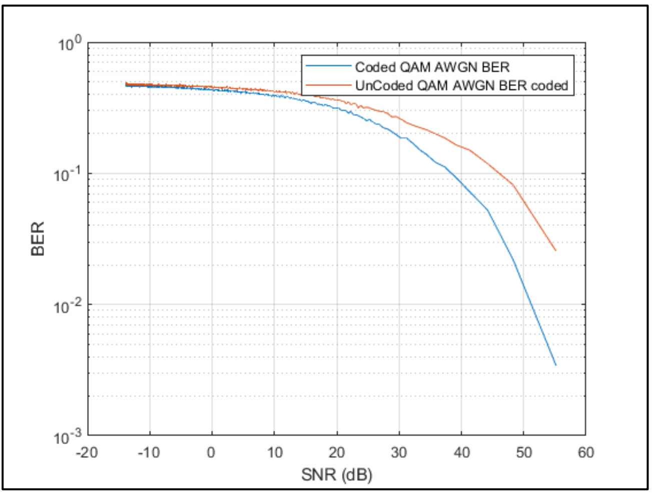


Comment: As we see from the previous figures that, the Un-coded QAM or QPSK gives us better performance at the low power of signal, this is because we use the same power for coded and the un-coded bits, which translate to the bit of the coded transmitted with the 1/3 of the power that the un-coded signal's bits transmitted with, that mean at low power of signal, the bit power in coded is too low, so you will get worth performance, but at a certain limit, the coded power become reasonable, so the coded gives us better performance.

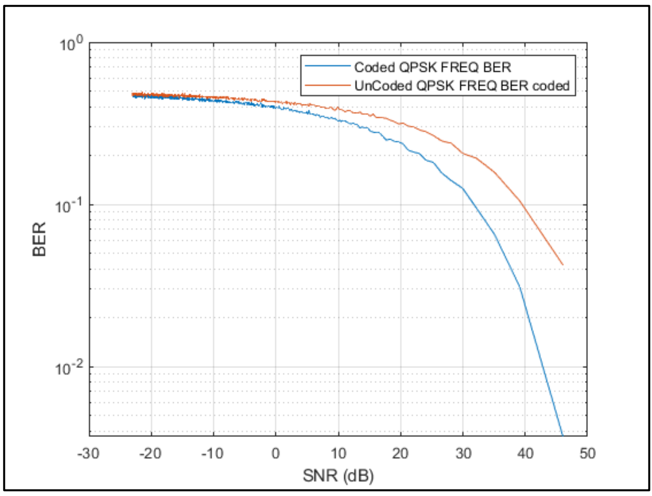
Part (2): OFDM system simulation:

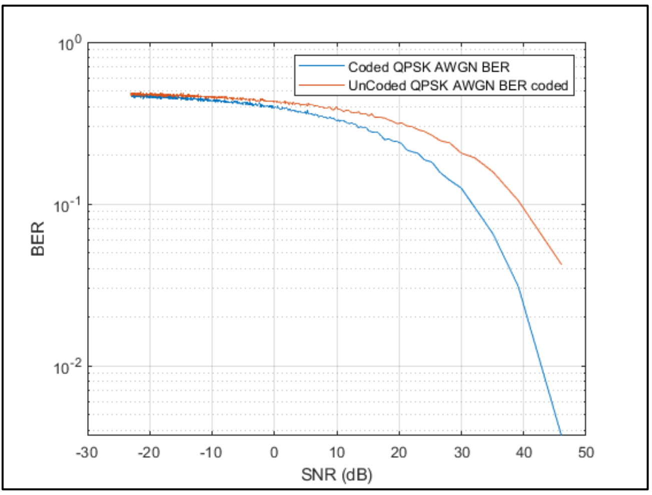
1) 16-QAM:





2) QPSK





Comment: here we have a completely different case, as shown in the previous figures the Coded is better in all cases, in Frequency selective channel or in AWGN channel, with QPSK or with QAM, and we can explain that due to the Interleaver makes an enhancement in the coded performance as it distribute the repeated bits along the channel, this will help us if the channel is bad in some bands and good in other bands, this distribution will make the good parts help the bad parts, and I still can receive it correctly, so even we use less power in codded, the idea of distribution of bits make a huge different .

Table of Contents

General Notation:

```
% Author : Ahmed Mokhtar Mahfouz
% Sec : 1
% BN : 30
% Created: 15/12/2021 03:44 PM
% last edit: 24/12/2021 05:28 PM
```

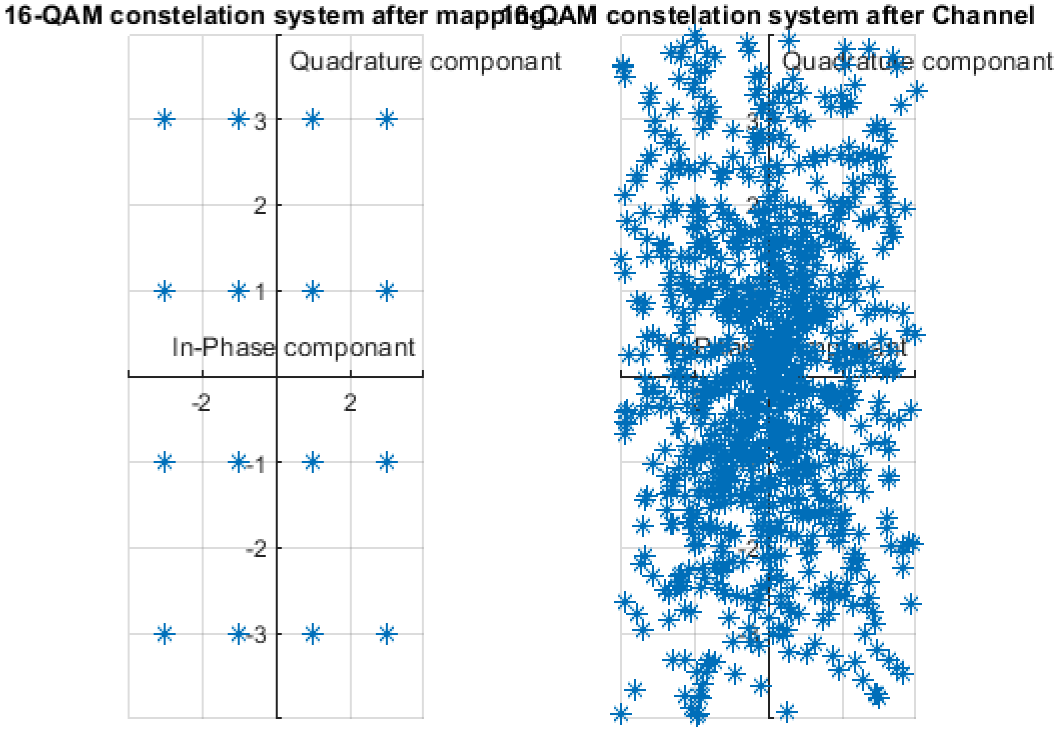
Generate the data stream:

```
clear ; clc ;close all
N = 1200;
data = randi([0,1],N,4);
N0 = 0.01:0.01:10;
N0_len = length(N0);
```

Bulit 16-QAM:

```
QAM_data = data(:,1) + data(:,2)*2 + data(:,3)*4+ data(:,4)*8;

% 1) Mapper:
%-----------
QAM_data_mapped_all = (QAM_data == 0)*(-3-3i) + (QAM_data ==
 1)*(-3-1i)...
    +(QAM_data == 2 )*(-3+3i) + (QAM_data == 3)*(-3+1i)...
    +(QAM_data == 4 )*(-1-3i) + (QAM_data == 5)*(-1-1i)...
    +(QAM_data == 6 )*(-1+3i) + (QAM_data == 7)*(-1+1i)...
    +(QAM_data == 8 )*(3-3i) + (QAM_data == 9)*(3-1i)...
    +(QAM_data == 10)*(3+3i) + (QAM_data == 11)*(3+1i)...
    +(QAM_data == 12)*(1-3i) + (QAM_data == 13)*(1-1i)...
    +(QAM_data == 14)*(1+3i) + (QAM_data == 15)*(1+1i);
```

```
QAM_data_mapped = [real(QAM_data_mapped_all) ,
 imag(QAM_data_mapped_all)];

% 2) The channel:
%----------------
v = randn(N,2);
R = sqrt(sum(v.^2,2))/sqrt(2);
QAM_data_channel = R.*QAM_data_mapped + v;

% 3) Reciver:
%------------
QAM_data_recived = zeros(N,4); % Intialize Matrix After the
 demodulator

QAM_data_recived(:,1) = ((QAM_data_channel(:,2)>-2) &
 (QAM_data_channel(:,2)<2))*1;
QAM_data_recived(:,2) = (QAM_data_channel(:,2)>0)*1;
QAM_data_recived(:,3) = ((QAM_data_channel(:,1)>-2) &
 (QAM_data_channel(:,1)<2))*1;
QAM_data_recived(:,4) = (QAM_data_channel(:,1)>0)*1;

% show the output figure for each stage:
%---------------------------------------
figure
subplot(1,2,1)
scatter(QAM_data_mapped(:,1), QAM_data_mapped(:,2),50,'*')
Ax = gca; % set origin (0,0)
Ax.XAxisLocation = 'origin';
Ax.YAxisLocation = 'origin';
axis([-4  4 -4  4])
title('16-QAM constelation system after mapping')
xlabel('In-Phase componant')
ylabel('Quadrature componant')
grid

subplot(1,2,2)
scatter(QAM_data_channel(:,1), QAM_data_channel(:,2),50,'*')
Ax = gca; % set origin (0,0)
Ax.XAxisLocation = 'origin';
Ax.YAxisLocation = 'origin';
axis([-4  4  -4  4])
title('16-QAM constelation system after Channel')
xlabel('In-Phase componant')
ylabel('Quadrature componant')
grid

BER_QAM = zeros(1,N0_len);

for i = 1:N0_len
    % 2) The channel:
    %----------------
    v = sqrt(N0(i)/2)*randn(N,2);

    QAM_data_channel = R.*QAM_data_mapped + v;
    QAM_data_channel = QAM_data_channel./R;

    % 3) Reciver:
    %------------
    QAM_data_recived(:,1) = ((QAM_data_channel(:,2)>-2) &
 (QAM_data_channel(:,2)<2))*1;
    QAM_data_recived(:,2) = (QAM_data_channel(:,2)>0)*1;
    QAM_data_recived(:,3) = ((QAM_data_channel(:,1)>-2) &
 (QAM_data_channel(:,1)<2))*1;
    QAM_data_recived(:,4) = (QAM_data_channel(:,1)>0)*1;

    % 4) BER Calculator:
    %-------------------
    BER_QAM(i) = (4*N - sum(sum(data==QAM_data_recived)))/(4*N);
end
```

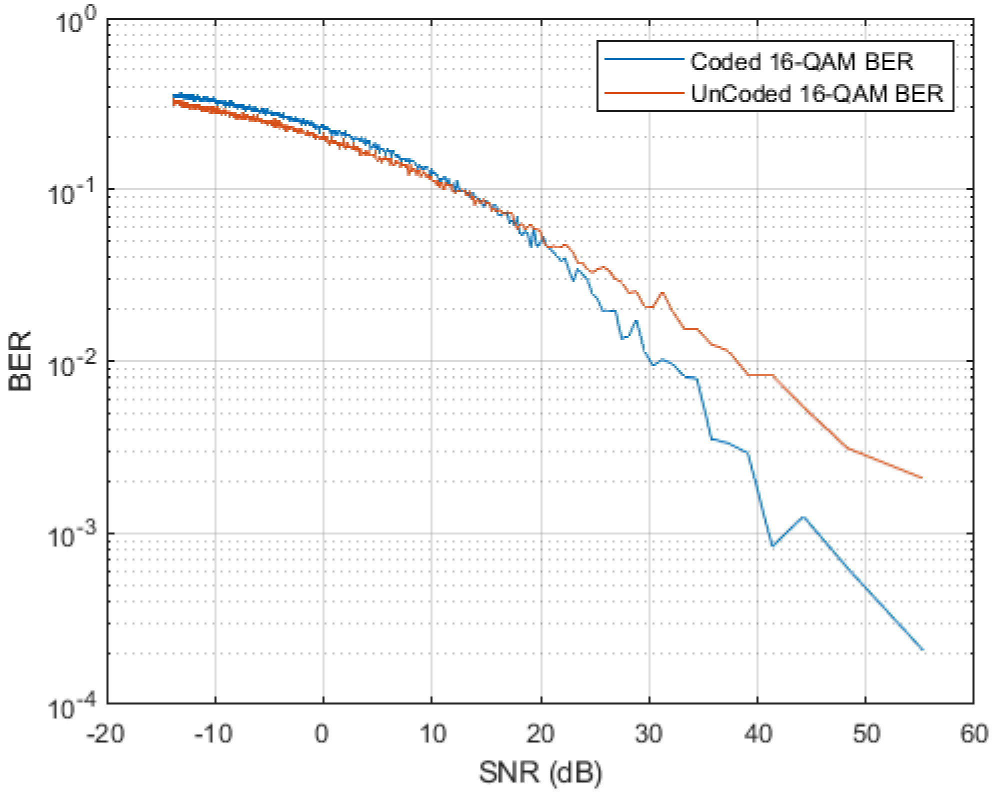


Bulit Coded encoder 16-QAM (1/3):

```
n=3;
M = n*N;
BER_coded_QAM = zeros(1,N0_len);
% 0) Encoder:
%------------
data_coded = repelem(data,3,1);

% 1) Mapper:
```

```
%-----------
QAM_data_coded = data_coded(:,1) + data_coded(:,2)*2 +
 data_coded(:,3)*4+ data_coded(:,4)*8;

QAM_data_mapped_all_coded = (QAM_data_coded == 0)*(-3-3i) +
 (QAM_data_coded == 1)*(-3-1i)...
    +(QAM_data_coded == 2 )*(-3+3i) + (QAM_data_coded == 3)*(-3+1i)...
    +(QAM_data_coded == 4 )*(-1-3i) + (QAM_data_coded == 5)*(-1-1i)...
    +(QAM_data_coded == 6 )*(-1+3i) + (QAM_data_coded == 7)*(-1+1i)...
    +(QAM_data_coded == 8 )*(3-3i) + (QAM_data_coded == 9)*(3-1i)...
    +(QAM_data_coded == 10)*(3+3i) + (QAM_data_coded == 11)*(3+1i)...
    +(QAM_data_coded == 12)*(1-3i) + (QAM_data_coded == 13)*(1-1i)...
    +(QAM_data_coded == 14)*(1+3i) + (QAM_data_coded == 15)*(1+1i);

QAM_data_mapped_coded = [real(QAM_data_mapped_all_coded)/sqrt(3) ,
 imag(QAM_data_mapped_all_coded)/sqrt(3)];
QAM_data_recived_decoded = zeros(N,4);
v = randn(M,2);
R = sqrt(sum(v.^2,2))/(sqrt(2));

for i = 1:N0_len
    % 2) The channel:
    %----------------
    v = sqrt(N0(i)/2)*randn(M,2);
    QAM_data_channel_coded = R.*QAM_data_mapped_coded + v;
    QAM_data_channel_coded = QAM_data_channel_coded./R;
    % 3) Reciver:
    %------------
    QAM_data_recived_coded(:,1) = ((QAM_data_channel_coded(:,2)>-2/
sqrt(3)) & (QAM_data_channel_coded(:,2)<2/sqrt(3)))*1;
    QAM_data_recived_coded(:,2) = (QAM_data_channel_coded(:,2)>0)*1;
    QAM_data_recived_coded(:,3) = ((QAM_data_channel_coded(:,1)>-2/
sqrt(3)) & (QAM_data_channel_coded(:,1)<2/sqrt(3)))*1;
    QAM_data_recived_coded(:,4) = (QAM_data_channel_coded(:,1)>0)*1;

    % 4) Decoder:
    %------------
    for j = 1:4
        for k =1:3:M
            QAM_data_recived_decoded(floor(k/3)+1,j) =
 (sum(QAM_data_recived_coded(k:k+2,j))>1)*1;
        end
    end
    % 5) BER Calculator:
    %-------------------
    BER_coded_QAM(i) = (4*N -
 sum(sum(data==QAM_data_recived_decoded)))/(4*N);
end

figure
semilogy(10*log(2.5./(N0)),BER_coded_QAM)
xlabel('SNR (dB)'); ylabel('BER')
hold on
semilogy(10*log(2.5./(N0)),BER_QAM)
```

```
xlabel('SNR (dB)'); ylabel('BER')
legend('Coded 16-QAM BER','UnCoded 16-QAM BER')
grid
```



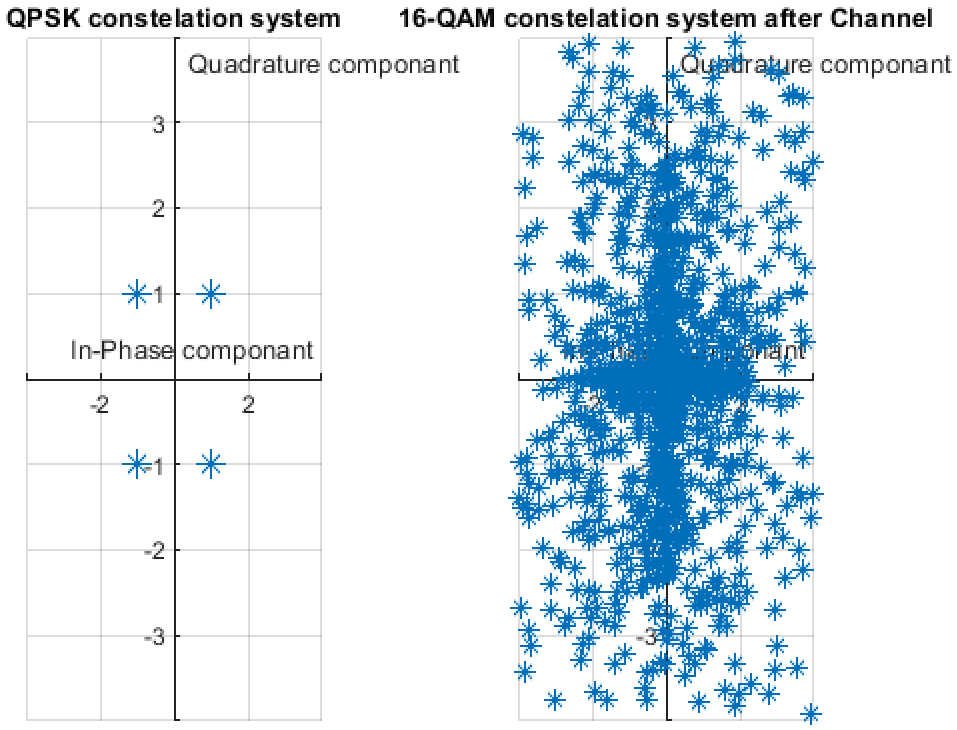
Clean variables to reduce memorey usage:

```
clear i j k b Ax v R

clear QAM_data_channel QAM_data_channel_coded QAM_data_decoded QAM_data_demaped
clear QAM_data_recived_decoded QAM_data_mapped QAM_data_mapped_coded QAM_data_reci
clear QAM_data_recived_coded QAM_data_mapped_all_coded QAM_data_mapped_all
clear QAM_data_coded QAM_data
```

Bulit QPSK:

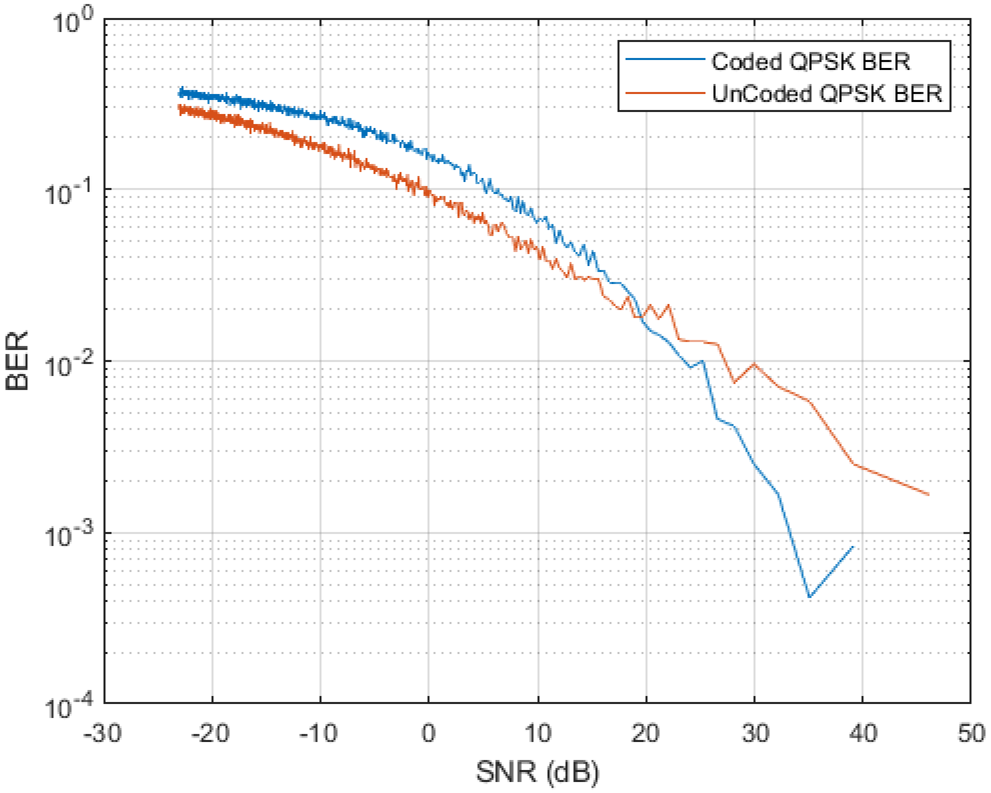
```
QPSK_data = data(:,1) + data(:,2)*2;
QPSK_data_mapped_all = (QPSK_data == 0)*(-1-1i) + (QPSK_data ==
 1)*(-1+1i)...
    +(QPSK_data == 2)*(1-1i) + (QPSK_data == 3)*(1+1i);

QPSK_data_mapped = [real(QPSK_data_mapped_all) ,
 imag(QPSK_data_mapped_all)];

% 2) The channel:
%----------------
```

```
v = randn(N,2);
R = sqrt(sum(v.^2,2));
QPSK_data_channel = R.*QPSK_data_mapped + v;

% 3) Reciver:
%------------
QPSK_data_recived = zeros(N,2);
QPSK_data_recived(:,1) = (QPSK_data_channel(:,2)>0);
QPSK_data_recived(:,2) = (QPSK_data_channel(:,1)>0);

% show the output figure for each stage:
%---------------------------------------
figure
subplot(1,2,1)
scatter(QPSK_data_mapped(:,1), QPSK_data_mapped(:,2),100,'*')
Ax = gca; % set origin (0,0)
Ax.XAxisLocation = 'origin';
Ax.YAxisLocation = 'origin';
axis([-4  4    -4  4])
title('QPSK constelation system')
xlabel('In-Phase componant')
ylabel('Quadrature componant')
grid

subplot(1,2,2)
scatter(QPSK_data_channel(:,1), QPSK_data_channel(:,2),50,'*')
Ax = gca; % set origin (0,0)
Ax.XAxisLocation = 'origin';
Ax.YAxisLocation = 'origin';
axis([-4  4    -4  4])
title('16-QAM constelation system after Channel')
xlabel('In-Phase componant')
ylabel('Quadrature componant')
grid

BER_QPSK = zeros(1,N0_len);

for i = 1:N0_len
    % 2) The channel:
    %----------------
    v = sqrt(N0(i)/2)*randn(N,2);

    QPSK_data_channel = R.*QPSK_data_mapped + v;
    QPSK_data_channel = QPSK_data_channel ./R;
    % 3) Reciver:
    %------------
    QPSK_data_recived(:,1) = (QPSK_data_channel(:,2)>0);
    QPSK_data_recived(:,2) = (QPSK_data_channel(:,1)>0);

    % 4) BER Calculator:
    %-------------------
    BER_QPSK(i) = (2*N - sum(sum(data(:,1:2)==QPSK_data_recived)))/
(2*N);
end
```



bulit Coded encoder QPSK (1/3):

```
QPSK_data_coded = data_coded(:,1) + data_coded(:,2)*2;
QPSK_data_mapped_all_coded = (QPSK_data_coded == 0)*(-1-1i) +
 (QPSK_data_coded == 1)*(-1+1i)...
    +(QPSK_data_coded == 2)*(1-1i) + (QPSK_data_coded == 3)*(1+1i);

QPSK_data_mapped_coded = [real(QPSK_data_mapped_all_coded)/sqrt(3) ,
 imag(QPSK_data_mapped_all_coded)/sqrt(3)];

QPSK_data_recived_coded = zeros(M,2);
QPSK_data_recived_decoded = zeros(N,2);
BER_coded_QPSK = zeros(1,N0_len);

v = randn(M,2);
R = sqrt(sum(v.^2,2))/sqrt(2);

for i = 1:N0_len
    % 2) The channel:
    %-----------------
    v = sqrt(N0(i)/2)*randn(M,2);
    QPSK_data_channel_coded = R.*QPSK_data_mapped_coded + v;
    QPSK_data_channel_coded = QPSK_data_channel_coded./R;
    % 3) Reciver:
    %------------
    QPSK_data_recived_coded(:,1) = (QPSK_data_channel_coded(:,2)>0);
    QPSK_data_recived_coded(:,2) = (QPSK_data_channel_coded(:,1)>0);

    % 4) Decoder:
    %------------
    for j = 1:2
        for k =1:3:M
            QPSK_data_recived_decoded(floor(k/3)+1,j) =
 (sum(QPSK_data_recived_coded(k:k+2,j))>1)*1;
        end
    end

    % 5) BER Calculator:
    %-------------------
    BER_coded_QPSK(i) = (2*N -
 sum(sum(data(:,1:2)==QPSK_data_recived_decoded)))/(2*N);
end

% Show output figure of coded encoder and uncoded encoder QAM:
%--------------------------------------------------------------
figure
semilogy(10*log(1./N0),BER_coded_QPSK)
xlabel('SNR (dB)'); ylabel('BER')
hold on
semilogy(10*log(1./(N0)),BER_QPSK)
xlabel('SNR (dB)'); ylabel('BER')
legend('Coded QPSK BER','UnCoded QPSK BER')
grid
```



Clean variables to reduce memorey usage:

```
clear i j k b Ax v R

clear QPSK_data_channel QPSK_data_channel_coded QPSK_data_decoded QPSK_data_demape
clear QPSK_data_recived_decoded QPSK_data_mapped QPSK_data_mapped_coded QPSK_data_
clear QPSK_data_recived_coded QPSK_data_mapped_all_coded QPSK_data_mapped_all
clear QPSK_data_coded QPSK_data

clear BER_QPSK BER_QAM BER_coded_QAM BER_coded_QPSK

clear data data_coded N M n N0 N0_len
```

Generate data for OFDM system simulation Part

```
N = 10000;
n = 3;
M = n*N;
data = randi([0,1],N,1);
data_coded = repelem(data,n);

N0 = 0.01:0.01:10;
N0_len = length(N0);
```

Bulit 16-QAM:

```
% 1) Interliver:
%---------------
x = 4*64;  % 4: number of bits ber symbols, 64: number of symbols
if (mod(N,x)~=0)    % Pad the data wiht zeros to make it multiple of
 64
    QAM_interliving = [data; zeros(x-mod(N,x),1)];
else
    QAM_interliving = data;
end

QAM_interliving = reshape(QAM_interliving,16,16,[]);
for i =1: size(QAM_interliving,3)
    QAM_interliving(:,:,i) = QAM_interliving(:,:,i)';
end
QAM_interliving = QAM_interliving(:);

QAM_interlived = reshape(QAM_interliving,[],4);

% 2) Mapper:
%-----------
QAM_data = QAM_interlived(:,1) + QAM_interlived(:,2)*2 ...
    + QAM_interlived(:,3)*4+ QAM_interlived(:,4)*8;

QAM_data_mapped_all = (QAM_data == 0)*(-3-3i) + (QAM_data ==
 1)*(-3-1i)...
    +(QAM_data == 2 )*(-3+3i) + (QAM_data == 3)*(-3+1i)...
    +(QAM_data == 4 )*(-1-3i) + (QAM_data == 5)*(-1-1i)...
    +(QAM_data == 6 )*(-1+3i) + (QAM_data == 7)*(-1+1i)...
    +(QAM_data == 8 )*(3-3i) + (QAM_data == 9)*(3-1i)...
    +(QAM_data == 10)*(3+3i) + (QAM_data == 11)*(3+1i)...
    +(QAM_data == 12)*(1-3i) + (QAM_data == 13)*(1-1i)...
    +(QAM_data == 14)*(1+3i) + (QAM_data == 15)*(1+1i);

QAM_data_mapped_all = reshape(QAM_data_mapped_all,64,[]);
% 3) IFFT:
%---------
QAM_IFFT = ifft(QAM_data_mapped_all,64);

% 4) Cyclic extention:
%---------------------
QAM_cyclic = [QAM_IFFT(49:64,:); QAM_IFFT];

BER_QAM_AWGN = zeros(length(N0),1);
BER_QAM_Freq = zeros(length(N0),1);

for j = 1:length(N0)
    % 5) channel:
    %------------
    %=>AWGN:
    QAM_AWGN = QAM_cyclic +
 sqrt(N0(j)/2)*(randn(size(QAM_cyclic))+1i*randn(size(QAM_cyclic)));
```

```
   %=>Frequency selective:
   h=[0.8 0 0 0 0 0 0 0 0 0 0.6];        % Filter
   QAM_Freq =
zeros(size(QAM_cyclic,1)+size(h,2)-1,size(QAM_cyclic,2));
   for i = 1:size(QAM_cyclic,2)
       QAM_Freq(:,i)= conv(QAM_AWGN(:,i), h);
   end

   % 6) Reciver
   %-----------
   %=>AWGN
   QAM_AWGN_Recived = zeros(size(QAM_IFFT));
   for i = 1:size(QAM_cyclic,2)
       QAM_AWGN_Recived(:,i) =
QAM_AWGN(size(QAM_IFFT(49:64,i),1)+1:end,i);
   end
   QAM_AWGN_FFT = fft(QAM_AWGN_Recived,64);
   QAM_AWGN_FFT= [real(QAM_AWGN_FFT(:)) imag(QAM_AWGN_FFT(:))];
   %=>Frequency selective:
   QAM_Freq_deconv = zeros(size(QAM_cyclic));
   QAM_Freq_Recived = zeros(size(QAM_IFFT));
   for i = 1:size(QAM_cyclic,2)
       QAM_Freq_deconv(:,i)= deconv(QAM_Freq(:,i), h);
   end
   for i = 1:size(QAM_cyclic,2)
       QAM_Freq_Recived(:,i) =
QAM_Freq_deconv(size(QAM_IFFT(49:64,i),1)+1:end,i);
   end
   QAM_Freq_FFT = fft(QAM_Freq_Recived,64);
   QAM_Freq_FFT= [real(QAM_Freq_FFT(:)) imag(QAM_Freq_FFT(:))];

   % 7)Demapper:
   %------------
   %=>AWGN
   QAM_AWGN_demaped = zeros(size(QAM_interlived));
   QAM_AWGN_demaped(:,1) = ((QAM_AWGN_FFT(:,2)>-2) &
(QAM_AWGN_FFT(:,2)<2))*1;
   QAM_AWGN_demaped(:,2) = (QAM_AWGN_FFT(:,2)>0)*1;
   QAM_AWGN_demaped(:,3) = ((QAM_AWGN_FFT(:,1)>-2) &
(QAM_AWGN_FFT(:,1)<2))*1;
   QAM_AWGN_demaped(:,4) = (QAM_AWGN_FFT(:,1)>0)*1;

   %=>Frequency selective:
   QAM_Freq_demaped = zeros(size(QAM_interlived));
   QAM_Freq_demaped(:,1) = ((QAM_Freq_FFT(:,2)>-2) &
(QAM_Freq_FFT(:,2)<2))*1;
   QAM_Freq_demaped(:,2) = (QAM_Freq_FFT(:,2)>0)*1;
   QAM_Freq_demaped(:,3) = ((QAM_Freq_FFT(:,1)>-2) &
(QAM_Freq_FFT(:,1)<2))*1;
   QAM_Freq_demaped(:,4) = (QAM_Freq_FFT(:,1)>0)*1;

   % deinteliver
   QAM_AWGN_dinterlived =reshape(QAM_AWGN_demaped(:),16,16,[]);
```

```
    for i = 1: size(QAM_AWGN_dinterlived,3)
        QAM_AWGN_dinterlived(:,:,i) =  QAM_AWGN_dinterlived(:,:,i)';
    end
    QAM_AWGN_final = QAM_AWGN_dinterlived(1:length(data))';

    QAM_Freq_dinterlived =reshape(QAM_Freq_demaped(:),16,16,[]);
    for i = 1: size(QAM_Freq_dinterlived,3)
        QAM_Freq_dinterlived(:,:,i) =  QAM_Freq_dinterlived(:,:,i)';
    end
    QAM_Freq_final = QAM_Freq_dinterlived(1:length(data))';

    BER_QAM_AWGN(j) = 1-sum(QAM_AWGN_final == data)/N;
    BER_QAM_Freq(j) = 1-sum(QAM_Freq_final == data)/N;
end
```

Bulit coded 16-QAM:

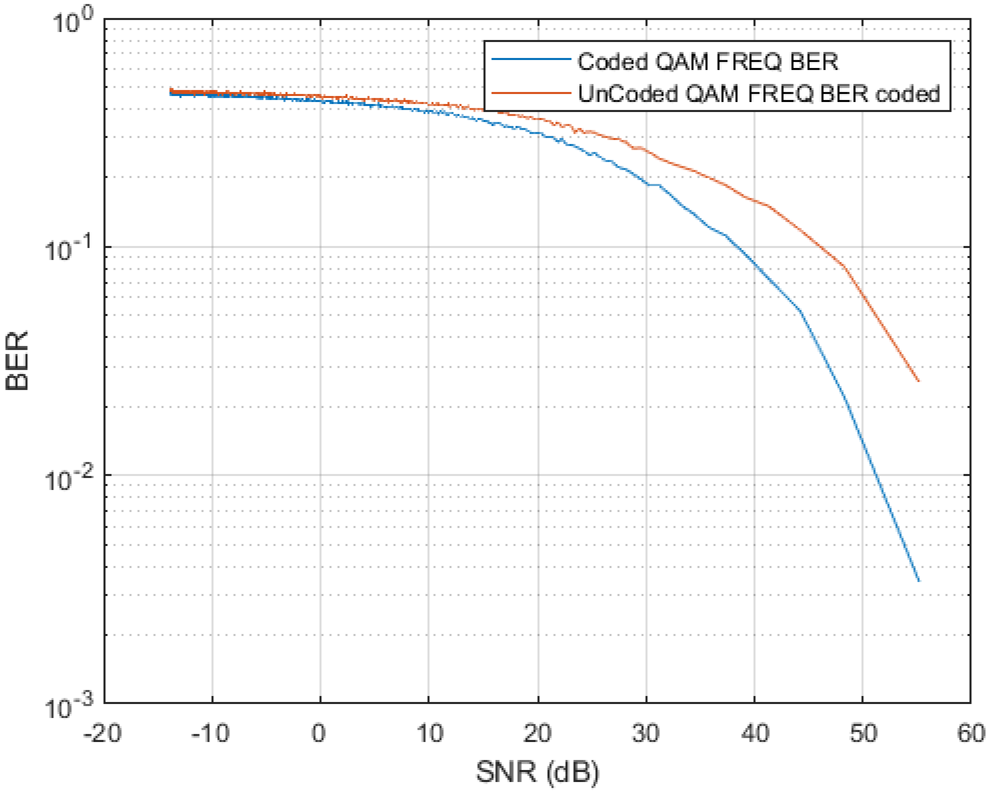
```
% 1) Interliver:
%---------------
x = 3*4*21; % 3 coded, 4 bits symbol, 21 symbol,

if (mod(M,x)~=0)  % padd with zeros to make the length of *x
    QAM_interliving_coded = [data_coded; zeros(x-mod(M,x),1)];
else
    QAM_interliving_coded = data_coded;
end

QAM_interliving_coded = reshape(QAM_interliving_coded,[], x);
QAM_interliving_coded = [QAM_interliving_coded,
 zeros(size(QAM_interliving_coded,1),256-x)];
QAM_interliving_coded = QAM_interliving_coded(:);

QAM_interliving_coded = reshape(QAM_interliving_coded,16,16,[]);
for i =1: size(QAM_interliving_coded,3)
    QAM_interliving_coded(:,:,i) = QAM_interliving_coded(:,:,i)';
end
QAM_interliving_coded = QAM_interliving_coded(:);
QAM_interliving_coded = reshape(QAM_interliving_coded,[],4);

% 2) Mapper:
%-----------
QAM_data_coded = QAM_interliving_coded(:,1) +
 QAM_interliving_coded(:,2)*2 ...
    + QAM_interliving_coded(:,3)*4+ QAM_interliving_coded(:,4)*8;

QAM_data_mapped_all_coded = (QAM_data_coded == 0)*(-3-3i) +
 (QAM_data_coded == 1)*(-3-1i)...
    +(QAM_data_coded == 2 )*(-3+3i) + (QAM_data_coded == 3)*(-3+1i)...
    +(QAM_data_coded == 4 )*(-1-3i) + (QAM_data_coded == 5)*(-1-1i)...
    +(QAM_data_coded == 6 )*(-1+3i) + (QAM_data_coded == 7)*(-1+1i)...
    +(QAM_data_coded == 8 )*(3-3i) + (QAM_data_coded == 9)*(3-1i)...
    +(QAM_data_coded == 10)*(3+3i) + (QAM_data_coded == 11)*(3+1i)...
    +(QAM_data_coded == 12)*(1-3i) + (QAM_data_coded == 13)*(1-1i)...
    +(QAM_data_coded == 14)*(1+3i) + (QAM_data_coded == 15)*(1+1i);
QAM_data_mapped_all_coded = reshape(QAM_data_mapped_all_coded,64,[])/
sqrt(3);

% 3) IFFT:
%---------
QAM_IFFT_coded = ifft(QAM_data_mapped_all_coded,64)*sqrt(3);     %
 *sqrt(3) eq to compare with -2/sqrt(3) in the demapper

% 4) Cyclic extention:
%---------------------
QAM_cyclic_coded = [QAM_IFFT_coded(49:64,:); QAM_IFFT_coded];

BER_QAM_AWGN_coded = zeros(length(N0),1);
BER_QAM_Freq_coded = zeros(length(N0),1);

for j = 1:length(N0)
    % 5) channel:
    %------------
    %=>AWGN:
    QAM_AWGN_coded = QAM_cyclic_coded +
 sqrt(N0(j)/2)*(randn(size(QAM_cyclic_coded))+1i*randn(size(QAM_cyclic_coded)));
    %=>Frequency selective:
    h=[0.8 0 0 0 0 0 0 0 0 0.6];          % Filter
    QAM_Freq_coded =
 zeros(size(QAM_cyclic_coded,1)+size(h,2)-1,size(QAM_cyclic_coded,2));
    for i = 1:size(QAM_cyclic_coded,2)
        QAM_Freq_coded(:,i)= conv(QAM_AWGN_coded(:,i), h);
    end

    % 6) Reciver
    %-----------
    %=>AWGN
    QAM_AWGN_Recived_coded = zeros(size(QAM_IFFT_coded));
    for i = 1:size(QAM_cyclic_coded,2)
        QAM_AWGN_Recived_coded(:,i) =
 QAM_AWGN_coded(size(QAM_IFFT_coded(49:64,i),1)+1:end,i);
    end
    QAM_AWGN_FFT_coded = fft(QAM_AWGN_Recived_coded,64);
    QAM_AWGN_FFT_coded= [real(QAM_AWGN_FFT_coded(:))
 imag(QAM_AWGN_FFT_coded(:))];
    %=>Frequency selective:
    QAM_Freq_deconv_coded = zeros(size(QAM_cyclic_coded));
    QAM_Freq_Recived_coded = zeros(size(QAM_IFFT_coded));
    for i = 1:size(QAM_cyclic_coded,2)
        QAM_Freq_deconv_coded(:,i)= deconv(QAM_Freq_coded(:,i), h);
    end
    for i = 1:size(QAM_cyclic_coded,2)
        QAM_Freq_Recived_coded(:,i) =
 QAM_Freq_deconv_coded(size(QAM_IFFT_coded(49:64,i),1)+1:end,i);
    end
    QAM_Freq_FFT_coded = fft(QAM_Freq_Recived_coded,64);
    QAM_Freq_FFT_coded= [real(QAM_Freq_FFT_coded(:))
 imag(QAM_Freq_FFT_coded(:))];
```

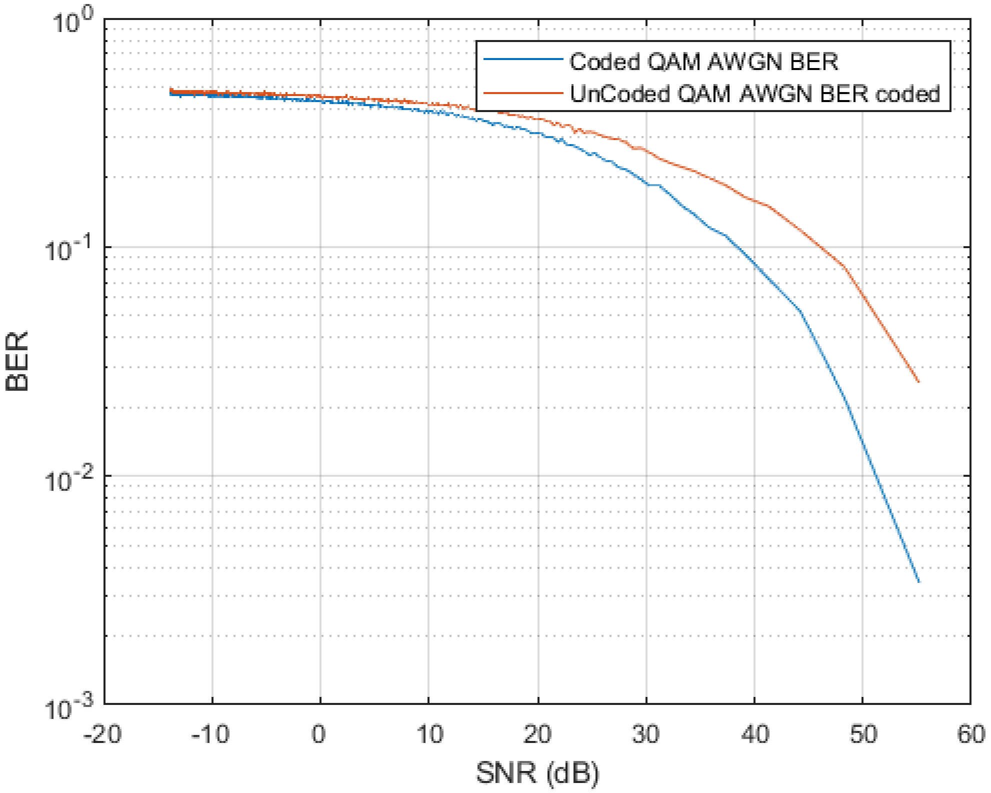
```
    % 7)Demapper:
    %------------
    %=>AWGN
    QAM_AWGN_demaped_coded = zeros(size(QAM_interliving_coded));
    QAM_AWGN_demaped_coded(:,1) = ((QAM_AWGN_FFT_coded(:,2)>-2) &
 (QAM_AWGN_FFT_coded(:,2)<2))*1;
    QAM_AWGN_demaped_coded(:,2) = (QAM_AWGN_FFT_coded(:,2)>0)*1;
    QAM_AWGN_demaped_coded(:,3) = ((QAM_AWGN_FFT_coded(:,1)>-2) &
 (QAM_AWGN_FFT_coded(:,1)<2))*1;
    QAM_AWGN_demaped_coded(:,4) = (QAM_AWGN_FFT_coded(:,1)>0)*1;

    %=>Frequency selective:
    QAM_Freq_demaped_coded = zeros(size(QAM_interliving_coded));
    QAM_Freq_demaped_coded(:,1) = ((QAM_Freq_FFT_coded(:,2)>-2) &
 (QAM_Freq_FFT_coded(:,2)<2))*1;
    QAM_Freq_demaped_coded(:,2) = (QAM_Freq_FFT_coded(:,2)>0)*1;
    QAM_Freq_demaped_coded(:,3) = ((QAM_Freq_FFT_coded(:,1)>-2) &
 (QAM_Freq_FFT_coded(:,1)<2))*1;
    QAM_Freq_demaped_coded(:,4) = (QAM_Freq_FFT_coded(:,1)>0)*1;

    % deinteliver
    QAM_AWGN_dinterlived_coded
 =reshape(QAM_AWGN_demaped_coded(:),16,16,[]);
    for i = 1: size(QAM_AWGN_dinterlived_coded,3)
        QAM_AWGN_dinterlived_coded(:,:,i) =
 QAM_AWGN_dinterlived_coded(:,:,i)';
    end
    QAM_AWGN_dinterlived_coded = reshape(QAM_AWGN_dinterlived_coded,
[],256);
    QAM_AWGN_dinterlived_coded = QAM_AWGN_dinterlived_coded(:,1:x);
    QAM_AWGN_dinterlived_coded = QAM_AWGN_dinterlived_coded(:);

    QAM_AWGN_recived_coded =
 QAM_AWGN_dinterlived_coded(1:length(data_coded))';

    QAM_Freq_dinterlived_coded
 =reshape(QAM_Freq_demaped_coded(:),16,16,[]);
    for i = 1: size(QAM_Freq_dinterlived_coded,3)
        QAM_Freq_dinterlived_coded(:,:,i) =
 QAM_Freq_dinterlived_coded(:,:,i)';
    end
 QAM_Freq_dinterlived_coded = reshape(QAM_Freq_dinterlived_coded,
[],256);
    QAM_Freq_dinterlived_coded = QAM_Freq_dinterlived_coded(:,1:x);
    QAM_Freq_dinterlived_coded = QAM_Freq_dinterlived_coded(:);

    QAM_Freq_recived_coded =
 QAM_Freq_dinterlived_coded(1:length(data_coded))';

    QAM_AWGN_final_coded = zeros(size(data));
    k=1;
    for i = 1:3:M
```

```
        QAM_AWGN_final_coded(k) = (sum(QAM_AWGN_recived_coded(i:i
+2))>1)*1;
        k=k+1;
    end

 QAM_Freq_final_coded = zeros(size(data));
    k=1;
    for i = 1:3:M
        QAM_Freq_final_coded(k) = (sum(QAM_Freq_recived_coded(i:i
+2))>1)*1;
        k=k+1;
    end

    BER_QAM_AWGN_coded(j) = 1-sum(QAM_AWGN_final_coded == data)/N;
    BER_QAM_Freq_coded(j) = 1-sum(QAM_Freq_final_coded == data)/N;
end
```

Show the output:

```
figure
semilogy(10*log(2.5./N0),BER_QAM_Freq_coded)
xlabel('SNR (dB)'); ylabel('BER')
hold on
semilogy(10*log(2.5./(N0)),BER_QAM_Freq)
xlabel('SNR (dB)'); ylabel('BER')
legend('Coded QAM FREQ BER','UnCoded QAM FREQ BER coded')
grid

figure
semilogy(10*log(2.5./N0),BER_QAM_AWGN_coded)
xlabel('SNR (dB)'); ylabel('BER')
hold on
semilogy(10*log(2.5./(N0)),BER_QAM_AWGN)
xlabel('SNR (dB)'); ylabel('BER')
legend('Coded QAM AWGN BER','UnCoded QAM AWGN BER coded')
grid
```


Coded QAM FREQ BER
UnCoded QAM FREQ BER coded
BER
SNR (dB)
10^0
10^{-1}
10^{-2}
10^{-3}
-20
-10
0
10
20
30
40
50
60


Coded QAM AWGN BER
UnCoded QAM AWGN BER coded
BER
SNR (dB)
10^0
10^{-1}
10^{-2}
10^{-3}
-20
-10
0
10
20
30
40
50
60

Bulit QPSK:

```
% 1) Interliver:
%---------------
x = 2*64;

if (mod(N,x)~=0)    % Pad the data wiht zeros to make it multiable of
 64
    QPSK_interliving = [data; zeros(x-mod(N,x),1)];
else
    QPSK_interliving = data;
end

QPSK_interliving= reshape(QPSK_interliving,8,16,[]);
QPSK_interlived
 =zeros(size(QPSK_interliving,2),size(QPSK_interliving,1),size(QPSK_interliving,3)
for i =1: size(QPSK_interliving,3)
    QPSK_interlived(:,:,i) = QPSK_interliving(:,:,i)';
end
QPSK_interlived = QPSK_interlived(:);
QPSK_interlived = reshape(QPSK_interlived,[],2);

% 2) Mapper:
%-----------
QPSK_data = QPSK_interlived(:,1) + QPSK_interlived(:,2)*2;

QPSK_data_mapped_all = (QPSK_data == 0)*(-1-1i) + (QPSK_data ==
 1)*(-1+1i)...
    +(QPSK_data == 2)*(1-1i) + (QPSK_data == 3)*(1+1i);
QPSK_data_mapped_all = reshape(QPSK_data_mapped_all,64,[]);

% 3) IFFT:
%---------
QPSK_IFFT = ifft(QPSK_data_mapped_all,64);

% 4) Cyclic extention:
%---------------------
QPSK_cyclic = [QPSK_IFFT(49:64,:); QPSK_IFFT];

BER_QPSK_AWGN = zeros(length(N0),1);
BER_QPSK_Freq = zeros(length(N0),1);
for j = 1:length(N0)
    % 5) channel:
    %------------
    %=>AWGN:
    QPSK_AWGN = QPSK_cyclic +
 sqrt(N0(j)/2)*(randn(size(QPSK_cyclic))+1i*randn(size(QPSK_cyclic)));

    %=>Frequency selective:
    h=[0.8 0 0 0 0 0 0 0 0 0 0.6];        % Filter
    QPSK_Freq =
 zeros(size(QPSK_cyclic,1)+size(h,2)-1,size(QPSK_cyclic,2));
    for i = 1:size(QPSK_cyclic,2)
```

```
       QPSK_Freq(:,i)= conv(QPSK_AWGN(:,i), h);
   end

   % 6) Reciver
   %-----------
   %=>AWGN
   QPSK_AWGN_Recived = zeros(size(QPSK_IFFT));
   for i = 1:size(QPSK_cyclic,2)
       QPSK_AWGN_Recived(:,i) =
QPSK_AWGN(size(QPSK_IFFT(49:64,i),1)+1:end,i);
   end
   QPSK_AWGN_FFT = fft(QPSK_AWGN_Recived,64);
   QPSK_AWGN_FFT= [real(QPSK_AWGN_FFT(:)) imag(QPSK_AWGN_FFT(:))];
   %=>Frequency selective:
   QPSK_Freq_deconv = zeros(size(QPSK_cyclic));
   QPSK_Freq_Recived = zeros(size(QPSK_IFFT));
   for i = 1:size(QPSK_cyclic,2)
       QPSK_Freq_deconv(:,i)= deconv(QPSK_Freq(:,i), h);
   end
   for i = 1:size(QPSK_cyclic,2)
       QPSK_Freq_Recived(:,i) =
QPSK_Freq_deconv(size(QPSK_IFFT(49:64,i),1)+1:end,i);
   end
   QPSK_Freq_FFT = fft(QPSK_Freq_Recived,64);
   QPSK_Freq_FFT= [real(QPSK_Freq_FFT(:)) imag(QPSK_Freq_FFT(:))];

   % 7)Demapper:
   %------------
   %=>AWGN
   QPSK_AWGN_demaped = zeros(size(QPSK_interlived));
   QPSK_AWGN_demaped(:,1) = (QPSK_AWGN_FFT(:,2)>0);
   QPSK_AWGN_demaped(:,2) = (QPSK_AWGN_FFT(:,1)>0);
   %=>Frequency selective:
   QPSK_Freq_demaped = zeros(size(QPSK_interlived));
   QPSK_Freq_demaped(:,1) = (QPSK_Freq_FFT(:,2)>0);
   QPSK_Freq_demaped(:,2) = (QPSK_Freq_FFT(:,1)>0);

   % deinteliver
   QPSK_AWGN_dinterliving =reshape(QPSK_AWGN_demaped(:),16,8,[]);
   QPSK_AWGN_dinterlived =
zeros(size(QPSK_AWGN_dinterliving,2),size(QPSK_AWGN_dinterliving,1),size(QPSK_AWG
   for i = 1: size(QPSK_AWGN_dinterlived,3)
       QPSK_AWGN_dinterlived(:,:,i) =
QPSK_AWGN_dinterliving(:,:,i)';
   end
   QPSK_AWGN_final = QPSK_AWGN_dinterlived(1:length(data))';

   QPSK_Freq_dinterliving =reshape(QPSK_Freq_demaped(:),16,8,[]);
   QPSK_Freq_dinterlived =
zeros(size(QPSK_Freq_dinterliving,2),size(QPSK_Freq_dinterliving,1),size(QPSK_Fre
   for i = 1: size(QPSK_Freq_dinterlived,3)
       QPSK_Freq_dinterlived(:,:,i) =
QPSK_Freq_dinterliving(:,:,i)';
   end
```

```
    QPSK_Freq_final = QPSK_Freq_dinterlived(1:length(data))';

    BER_QPSK_AWGN(j) = 1-sum(QPSK_AWGN_final == data)/N;
    BER_QPSK_Freq(j) = 1-sum(QPSK_Freq_final == data)/N;
end
```

Bulit QPSK coded:

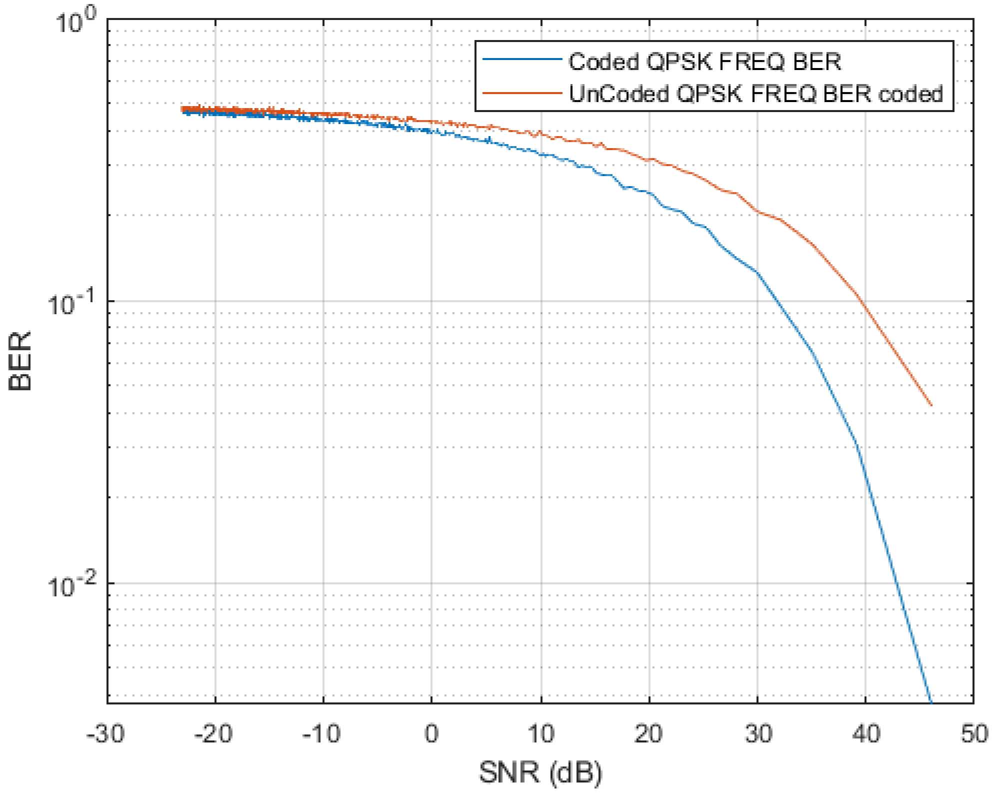
```
% 1) Interliver:
%----------------
x = 2*21*3;

if (mod(M,x)~=0)    % Pad the data wiht zeros to make it multiable of
 64
    QPSK_interliving_coded = [data_coded; zeros(x-mod(M,x),1)];
else
    QPSK_interliving_coded = data_coded;
end
QPSK_interliving_coded = reshape(QPSK_interliving_coded,[],x);  %
 padding with zero at the end of 64 IFFT
QPSK_interliving_coded = [QPSK_interliving_coded,
 zeros(size(QPSK_interliving_coded,1),128-x)];
QPSK_interliving_coded=QPSK_interliving_coded(:);

QPSK_interliving_coded= reshape(QPSK_interliving_coded,8,16,[]);
QPSK_interlived_coded
 =zeros(size(QPSK_interliving_coded,2),size(QPSK_interliving_coded,1),size(QPSK_in
for i =1: size(QPSK_interliving_coded,3)
    QPSK_interlived_coded(:,:,i) = QPSK_interliving_coded(:,:,i)';
end
QPSK_interlived_coded = QPSK_interlived_coded(:);
QPSK_interlived_coded = reshape(QPSK_interlived_coded,[],2);

% 2) Mapper:
%------------
QPSK_data_coded = QPSK_interlived_coded(:,1) +
 QPSK_interlived_coded(:,2)*2;

QPSK_data_mapped_all_coded = (QPSK_data_coded == 0)*(-1-1i) +
 (QPSK_data_coded == 1)*(-1+1i)...
    +(QPSK_data_coded == 2)*(1-1i) + (QPSK_data_coded == 3)*(1+1i);
QPSK_data_mapped_all_coded = reshape(QPSK_data_mapped_all_coded,64,
[])/sqrt(3);

% 3) IFFT:
%----------
QPSK_IFFT_coded = ifft(QPSK_data_mapped_all_coded,64)*sqrt(3); %
 *sqrt(3) eq to compare with -2/sqrt(3) in the demapper

% 4) Cyclic extention:
%----------------------
QPSK_cyclic_coded = [QPSK_IFFT_coded(49:64,:); QPSK_IFFT_coded];
```

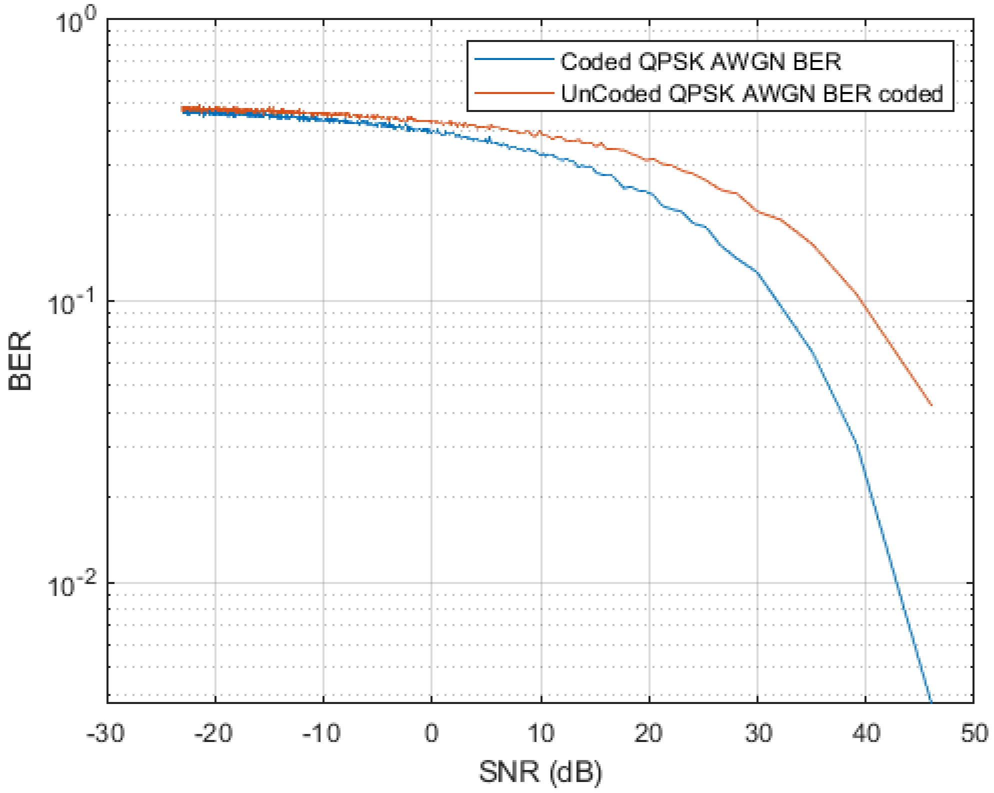
```
BER_QPSK_AWGN_coded = zeros(length(N0),1);
BER_QPSK_Freq_coded = zeros(length(N0),1);
for j = 1:length(N0)
    % 5) channel:
    %------------
    %=>AWGN:
    QPSK_AWGN_coded = QPSK_cyclic_coded +
 sqrt(N0(j)/2)*(randn(size(QPSK_cyclic_coded))+1i*randn(size(QPSK_cyclic_coded)));
    %=>Frequency selective:
    h=[0.8 0 0 0 0 0 0 0 0 0 0.6];        % Filter
    QPSK_Freq_coded =
 zeros(size(QPSK_cyclic_coded,1)+size(h,2)-1,size(QPSK_cyclic_coded,2));
    for i = 1:size(QPSK_cyclic_coded,2)
        QPSK_Freq_coded(:,i)= conv(QPSK_AWGN_coded(:,i), h);
    end

    % 6) Reciver
    %-----------
    %=>AWGN
    QPSK_AWGN_Recived_coded = zeros(size(QPSK_IFFT_coded));
    for i = 1:size(QPSK_cyclic_coded,2)
        QPSK_AWGN_Recived_coded(:,i) =
 QPSK_AWGN_coded(size(QPSK_IFFT_coded(49:64,i),1)+1:end,i);
    end
    QPSK_AWGN_FFT_coded = fft(QPSK_AWGN_Recived_coded,64);
    QPSK_AWGN_FFT_coded= [real(QPSK_AWGN_FFT_coded(:))
 imag(QPSK_AWGN_FFT_coded(:))];
    %=>Frequency selective:
    QPSK_Freq_deconv_coded = zeros(size(QPSK_cyclic_coded));
    QPSK_Freq_Recived_coded = zeros(size(QPSK_IFFT_coded));
    for i = 1:size(QPSK_cyclic_coded,2)
        QPSK_Freq_deconv_coded(:,i)= deconv(QPSK_Freq_coded(:,i), h);
    end
    for i = 1:size(QPSK_cyclic_coded,2)
        QPSK_Freq_Recived_coded(:,i) =
 QPSK_Freq_deconv_coded(size(QPSK_IFFT_coded(49:64,i),1)+1:end,i);
    end
    QPSK_Freq_FFT_coded = fft(QPSK_Freq_Recived_coded,64);
    QPSK_Freq_FFT_coded= [real(QPSK_Freq_FFT_coded(:))
 imag(QPSK_Freq_FFT_coded(:))];

    % 7)Demapper:
    %------------
    %=>AWGN
    QPSK_AWGN_demaped_coded = zeros(size(QPSK_interlived_coded));
    QPSK_AWGN_demaped_coded(:,1) = (QPSK_AWGN_FFT_coded(:,2)>0);
    QPSK_AWGN_demaped_coded(:,2) = (QPSK_AWGN_FFT_coded(:,1)>0);

    %=>Frequency selective:
    QPSK_Freq_demaped_coded = zeros(size(QPSK_interlived_coded));
    QPSK_Freq_demaped_coded(:,1) = (QPSK_Freq_FFT_coded(:,2)>0);
    QPSK_Freq_demaped_coded(:,2) = (QPSK_Freq_FFT_coded(:,1)>0);

    % deinteliver
    QPSK_AWGN_dinterliving_coded
 =reshape(QPSK_AWGN_demaped_coded(:),16,8,[]);
    QPSK_AWGN_dinterlived_coded =
 zeros(size(QPSK_AWGN_dinterliving_coded,2),size(QPSK_AWGN_dinterliving_coded,1),s
    for i = 1: size(QPSK_AWGN_dinterlived_coded,3)
        QPSK_AWGN_dinterlived_coded(:,:,i) =
 QPSK_AWGN_dinterliving_coded(:,:,i)';
    end
    QPSK_AWGN_dinterlived_coded = QPSK_AWGN_dinterlived_coded(:);  %
 remove the zero padded of the 64 IFFT
    QPSK_AWGN_dinterlived_coded = reshape(QPSK_AWGN_dinterlived_coded,
[],128);
    QPSK_AWGN_dinterlived_coded = QPSK_AWGN_dinterlived_coded(:,1:x);
    QPSK_AWGN_recived_coded =
 QPSK_AWGN_dinterlived_coded(1:length(data_coded))';

    QPSK_Freq_dinterliving_coded
 =reshape(QPSK_Freq_demaped_coded(:),16,8,[]);
    QPSK_Freq_dinterlived_coded =
 zeros(size(QPSK_Freq_dinterliving_coded,2),size(QPSK_Freq_dinterliving_coded,1),s
    for i = 1: size(QPSK_Freq_dinterlived_coded,3)
        QPSK_Freq_dinterlived_coded(:,:,i) =
 QPSK_Freq_dinterliving_coded(:,:,i)';
    end
    QPSK_Freq_dinterlived_coded = QPSK_Freq_dinterlived_coded(:);  %
 remove the zero padded of the 64 IFFT
    QPSK_Freq_dinterlived_coded = reshape(QPSK_Freq_dinterlived_coded,
[],128);
    QPSK_Freq_dinterlived_coded = QPSK_Freq_dinterlived_coded(:,1:x);
    QPSK_Freq_recived_coded =
 QPSK_Freq_dinterlived_coded(1:length(data_coded))';

    QPSK_AWGN_final_coded = zeros(size(data));
    k=1;
    for i = 1:3:M
        QPSK_AWGN_final_coded(k) = (sum(QPSK_AWGN_recived_coded(i:i
+2))>1)*1;
        k=k+1;
    end
    QPSK_Freq_final_coded = zeros(size(data));
    k=1;
    for i = 1:3:M
        QPSK_Freq_final_coded(k) = (sum(QPSK_Freq_recived_coded(i:i
+2))>1)*1;
        k=k+1;
    end
    BER_QPSK_AWGN_coded(j) = 1-sum(QPSK_AWGN_final_coded == data)/N;
    BER_QPSK_Freq_coded(j) = 1-sum(QPSK_Freq_final_coded == data)/N;
end
```

Show the output:

```
figure
```

```
semilogy(10*log(1./N0),BER_QPSK_Freq_coded)
xlabel('SNR (dB)'); ylabel('BER')
hold on
semilogy(10*log(1./(N0)),BER_QPSK_Freq)
xlabel('SNR (dB)'); ylabel('BER')
legend('Coded QPSK FREQ BER','UnCoded QPSK FREQ BER coded')
grid

figure
semilogy(10*log(1./N0),BER_QPSK_AWGN_coded)
xlabel('SNR (dB)'); ylabel('BER')
hold on
semilogy(10*log(1./(N0)),BER_QPSK_AWGN)
xlabel('SNR (dB)'); ylabel('BER')
legend('Coded QPSK AWGN BER','UnCoded QPSK AWGN BER coded')
grid
```





Published with MATLAB® R2020a