Cairo University
Faculty of Engineering
Department of Electronics and Electrical
Communications Engineering

4th Year
Digital Communications
Project – Fall 2022

Digital Communications Project

Team Report:

Team Members		Section	BN
1-	محمد أيمن عبد العليم أحمد	3	34
2-	محمد حسام عثمان يسن	3	35

Single system carrier Graphs:

QPSK

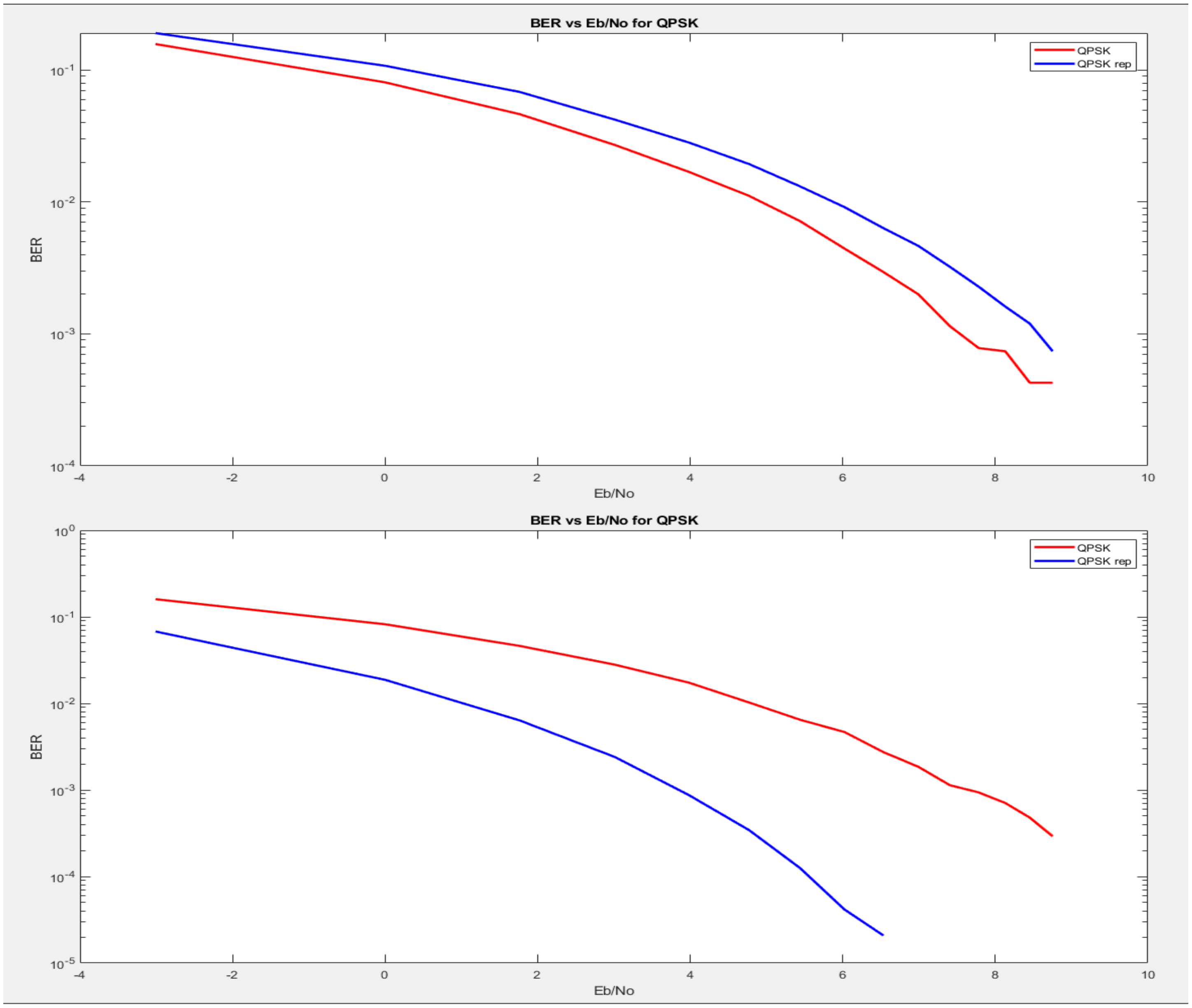


Figure 1 BER VS Eb/No(dB) for QPSK

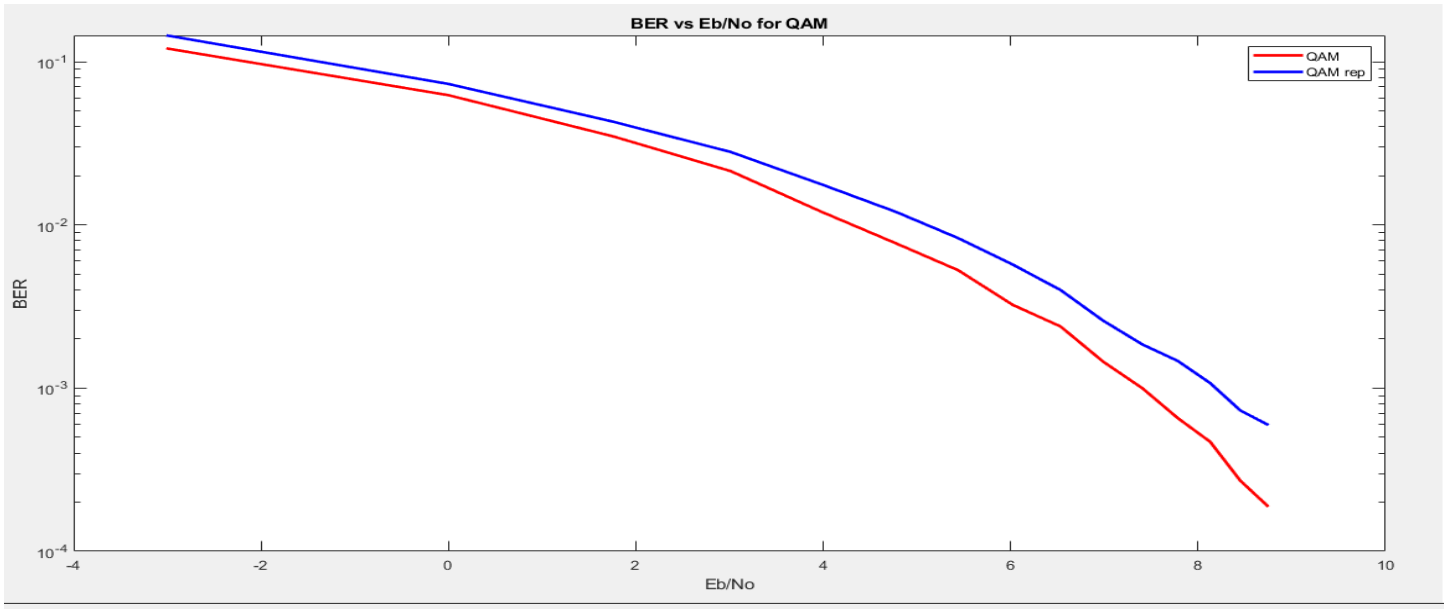
Note: First Graph the repetition code has same energy per information bit (Eb/3), second graph each bit has equal energy to the information bit.

Comments:

-The BER of the repetition code is worse than no coded in case of equal energy.
-When we increase the energy of the coded repetition bits the BER improves significantly.
-The BER of the QPSK is better than 16-QAM and we can see this also form theoretical laws.

$$BER_{QPSK} = \frac{1}{2} erfc(\frac{E_b}{N_0}), BER_{QAM} = \frac{3}{2} erfc(\sqrt{\frac{E_b}{2.5N_0}})$$

16-QAM



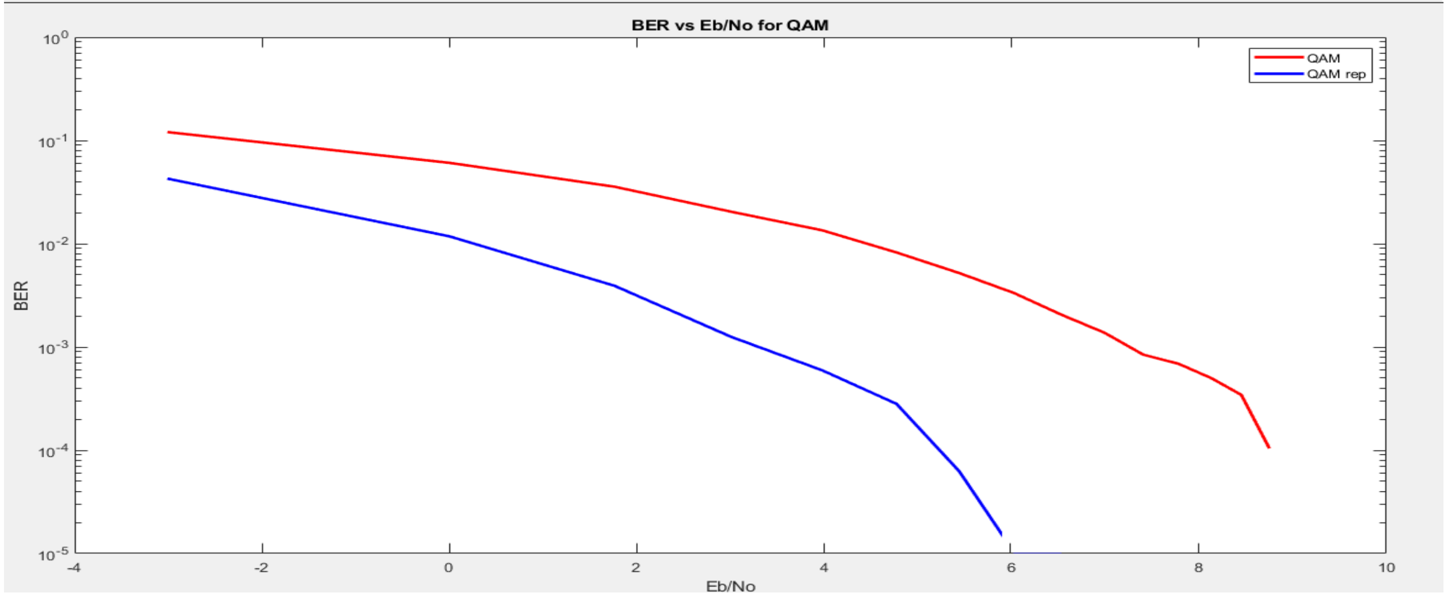


Figure 2 BER VS Eb/No (dB) for 16-QAM

Note: First Graph the repetition code has same energy per information bit (Eb/3), second graph each bit has equal energy to the information bit.

Comments:

-The BER of the repetition code is worse than no coded in case of equal energy.
-When we increase the energy of the coded repetition bits the BER improves significantly.
-The BER of the 16-QAM is worse than QPSK and we can see this also from theoretical laws.

$$BER_{QPSK} = \frac{1}{2} erfc(\frac{E_b}{N_0}), BER_{QAM} = \frac{3}{2} erfc(\sqrt{\frac{E_b}{2.5N_0}})$$

<u>OFDM system graphs:</u>

<u>QPSK</u>

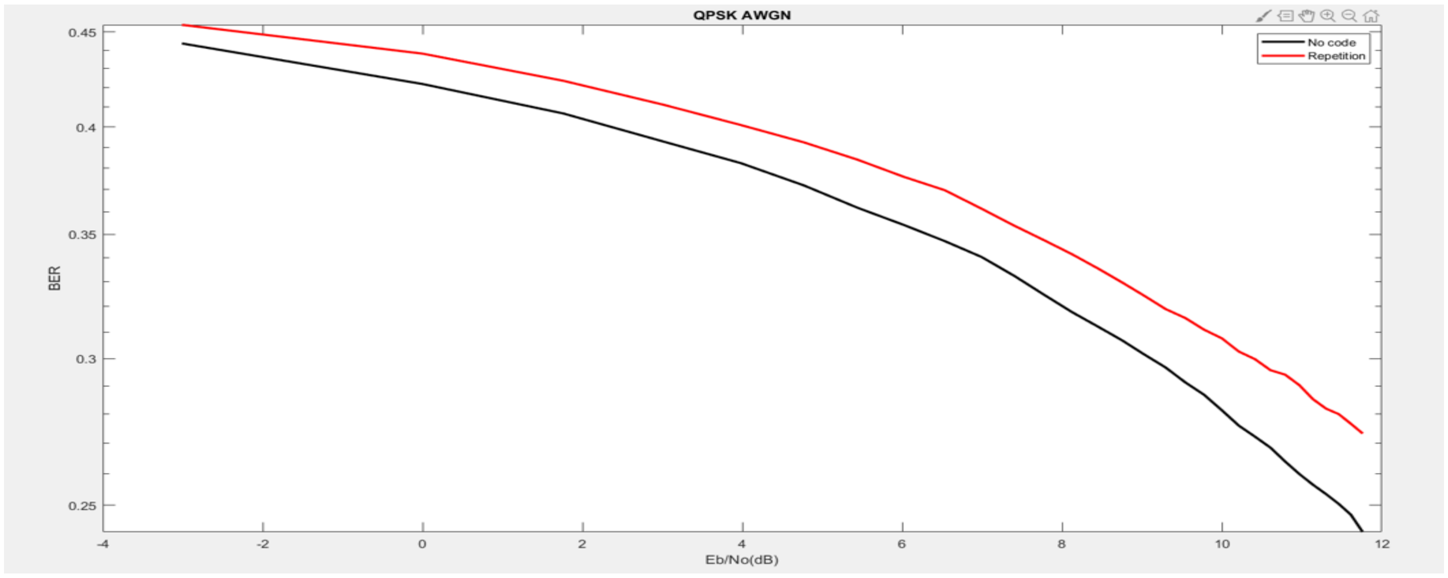


Figure 3 BER VS Eb/No (dB) for QPSK AWGN channel

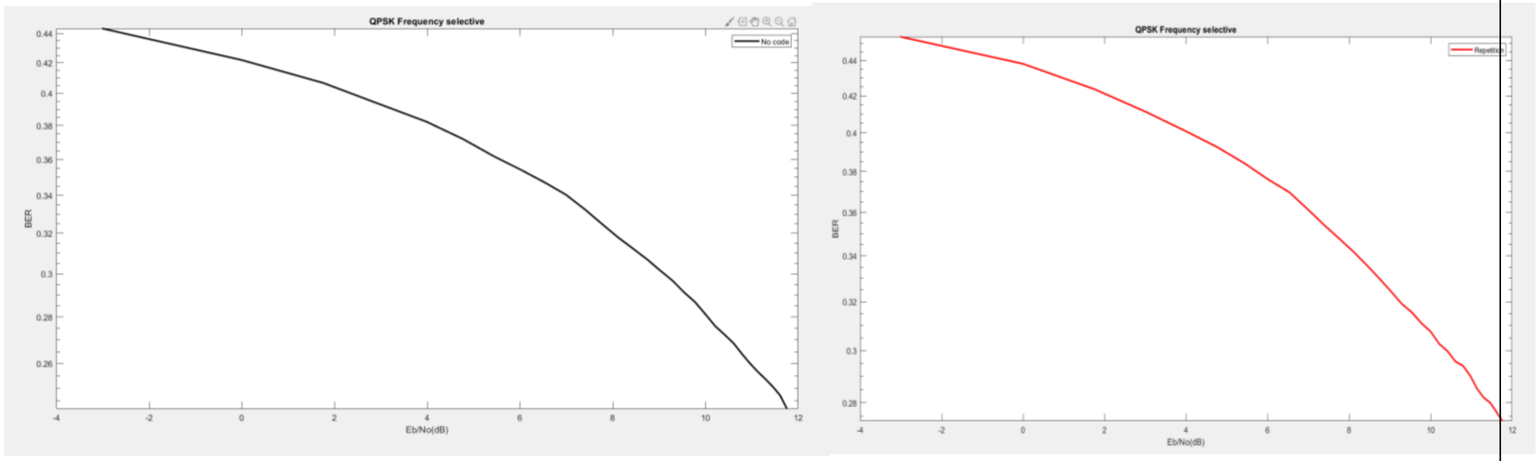


Figure 4 BER VS Eb/No (dB) for QPSK selective channel

Note: The repetition code has the same energy per information bit (Eb/3).

<u>Comments:</u>

-The BER of the repetition code is worse than no coded in case of equal energy per information bit.
-BER of both AWGN and Frequency selective fading channels is nearly the same as we use a perfect equalizer but AWGN is slightly better.
-The BER of the 16-QAM is worse than QPSK and we can see this also from theoretical laws.

$$BER_{QPSK} = \frac{1}{2}erfc(\frac{E_b}{N_0}),\ BER_{QAM} = \frac{3}{2}erfc(\sqrt{\frac{E_b}{2.5N_0}})$$

16-QAM

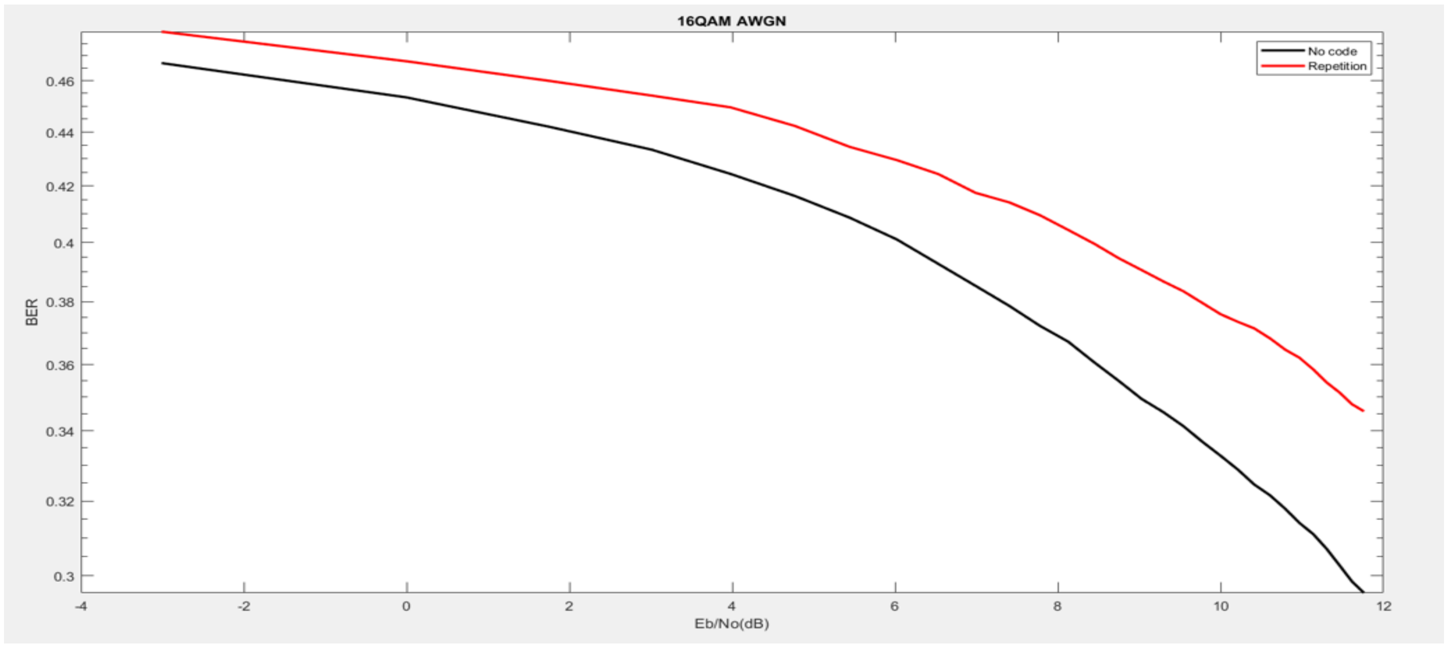
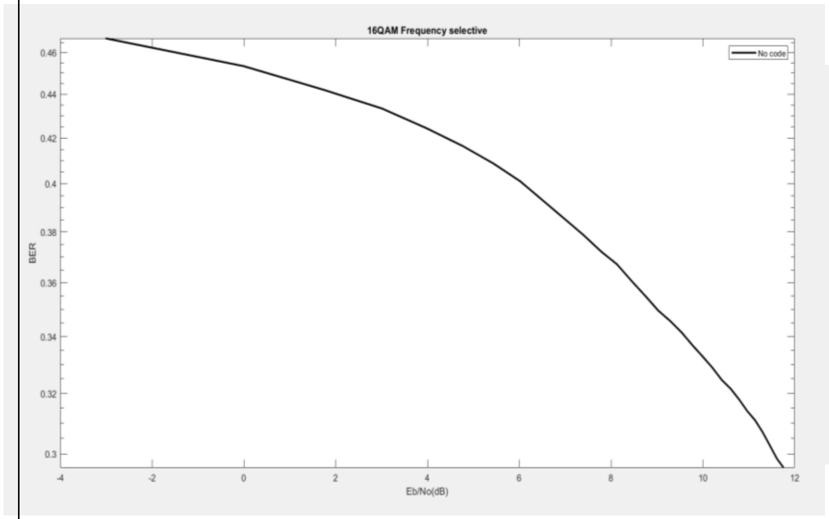


Figure 5 BER VS Eb/No (dB) for 16-QAM AWGN channel



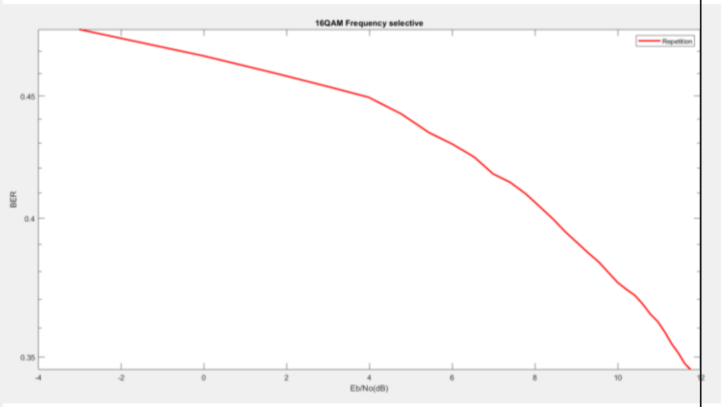


Figure 6 BER VS Eb/No (dB) for 16-QAM selective channel

Note: The repetition code has the same energy per information bit (Eb/3).

Comments:

-The BER of the repetition code is worse than no coded in case of equal energy per information bit.

-BER of both AWGN and Frequency selective fading channels is nearly the same as we use a perfect equalizer but AWGN is slightly better.

-The BER of the 16-QAM is worse than QPSK and we can see this also from theoretical laws.

$$BER_{QPSK} = \frac{1}{2} erfc(\frac{E_b}{N_0}),\ BER_{QAM} = \frac{3}{2} erfc(\sqrt{\frac{E_b}{2.5N_0}})$$

Water Filling

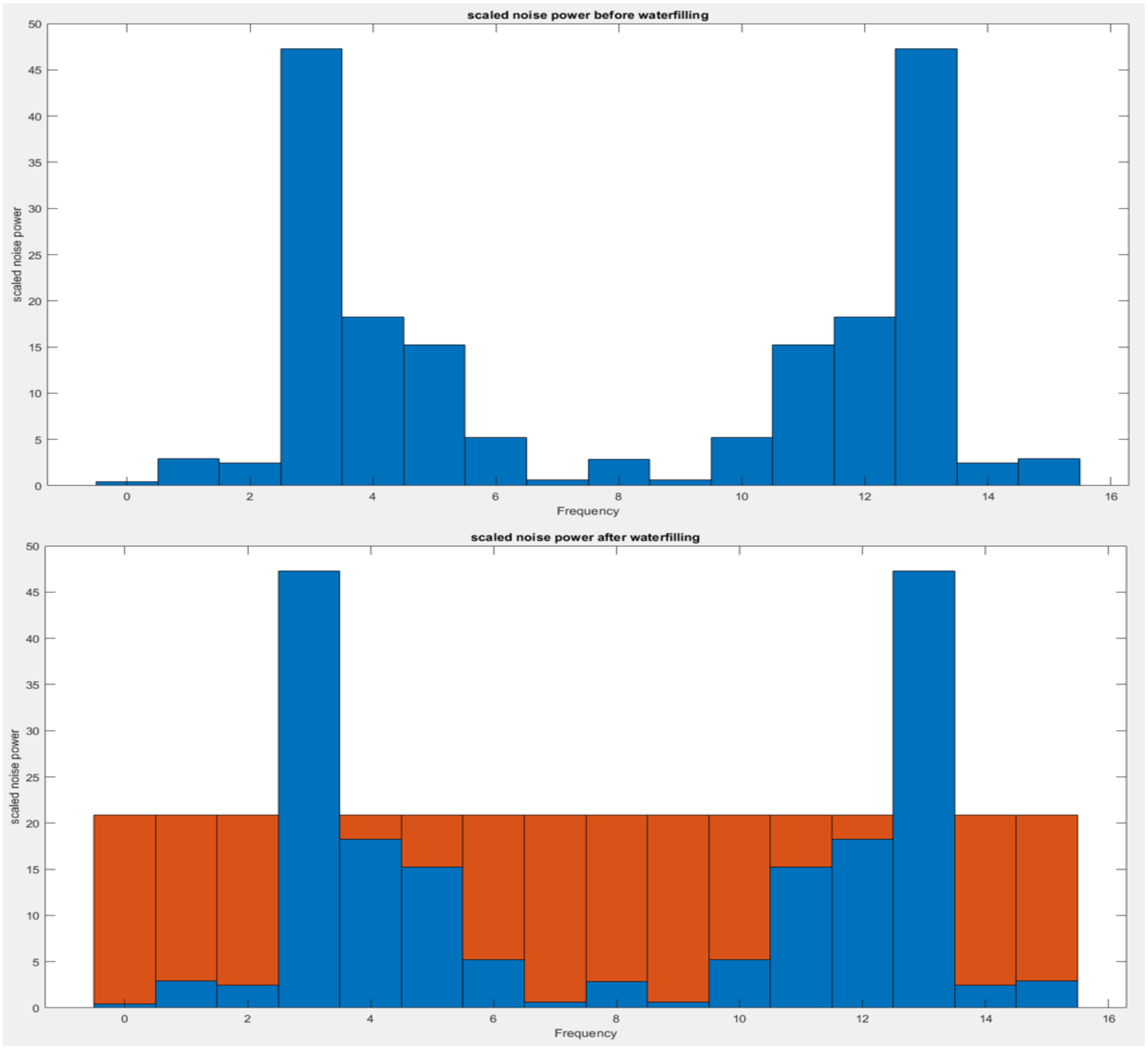


Figure 7 water falling power distribution

Comments:

Using water-filling graphical solution based on the equations : $P_n = K - \frac{\Gamma\sigma_n^2}{g_n^2} = K - \frac{2}{g_n^2}$ & $\sum_{n=1}^{16} P_n = 200$

We get power allocation per subcarrier =

P1	P2	P3	P4	P5	P6	P7	P8	P9	P10
20.47	18.01	18.43	0	2.65	5.675	15.71	20.26	18.06	20.26

P11	P12	P13	P14	P15	P16
15.71	5.675	2.65	0	18.43	18.01

Single carrier system

Code:

```
num_bits=96000;
stream=randi(2,1,num_bits)-1;
stream_trans=stream';
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Eb=1:15;
E0_QPSK=Eb;
No=2;
out_norep=repelem(stream_trans,1);
out_coded=repelem(stream_trans,3);
%%%%%%%%%%%Interleaver%%%%%%%%%
inter_storage=zeros(4,4);
out_inter1=[];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% for no rep
for k=1:16:size(out_norep,1)
    f=1;
for i=k:4:k+15
    inter_storage(f,:)=out_norep(i:1:i+3);
    f=f+1;
end
temp1=reshape(inter_storage,[],1);
out_inter1=cat(1,out_inter1,temp1);
end
stream=out_inter1';
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%QPSK TRANSMITTER
QPSK_tx=zeros(num_bits/2,1,length(Eb));
for k=1:2:num_bits
    if(isequal(stream(k:k+1),[0 0]))
        QPSK_tx((k+1)/2,1,:)=-1*sqrt(E0_QPSK)-1i*sqrt(E0_QPSK);
    elseif(isequal(stream(k:k+1),[1 0]))
        QPSK_tx((k+1)/2,1,:)=1*sqrt(E0_QPSK)-1i*sqrt(E0_QPSK);
    elseif(isequal(stream(k:k+1),[1 1]))
       QPSK_tx((k+1)/2,1,:)=1*sqrt(E0_QPSK)+1i*sqrt(E0_QPSK);
    else
        QPSK_tx((k+1)/2,1,:)=-1*sqrt(E0_QPSK)+1i*sqrt(E0_QPSK);
    end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%ADDING NOISE TO TRANSMITTED SIGNALS
v1_uncoded = randn(num_bits/2,1,length(Eb)).* sqrt(No/2);
v2_uncoded = 1i.* randn(num_bits/2,1,length(Eb)).* sqrt(No/2);
channel = sqrt(pow2(v1_uncoded) + pow2(v2_uncoded)) / sqrt(2);
noise=
sqrt(No/2)*randn(num_bits/2,1,length(Eb))+1j*sqrt(No/2).*randn(num_bits/2,1,length(Eb))
;
y_QPSK=channel.*QPSK_tx+noise;
y_QPSK=y_QPSK./channel;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%QPSK RECEIVER
rec_QPSK=zeros(num_bits,1,length(Eb));
for h=1:length(Eb)
for k=1:num_bits/2
    if(real(y_QPSK(k,1,h))>0 )
        b0=1;
    else
        b0=0;
    end
    if(imag(y_QPSK(k,1,h))>0 )
       b1=1;
    else
        b1=0;
    end
    rec_QPSK((k*2)-1,1,h)=b0;
    rec_QPSK((k*2),1,h)=b1;
end
end
%%%%%%%%%%%%%De-interleeaver%%%%%%%%%%%%%%%%%%
inter_storage=zeros(4,4);
out_inter2=zeros(length(out_norep),1,length(Eb));
for q=1:1:length(Eb)
vec=[];
for k=1:16:size(out_norep,1)
    f=1;
for i=k:4:k+15
    inter_storage(f,:)=rec_QPSK(i:1:i+3,1,q);
    f=f+1;
end
temp1=reshape(inter_storage,[],1);
vec=cat(1,vec,temp1);
end
out_inter2(:,1,q)=vec;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%ERROR FOR QPSK
sum_error_QPSK=zeros(length(Eb),1);
for h=1:length(Eb)
for k=1:num_bits
    if(out_inter2(k,1,h)~=stream_trans(k,1))
        sum_error_QPSK(h,1)=sum_error_QPSK(h,1)+1;
    end
end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%BER CALC FOR ALL SIGNALS
BER_QPSK=sum_error_QPSK/num_bits;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
inter_storage=zeros(4,4);
out_inter1rep=[];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% interleaver for rep
for k=1:16:size(out_coded,1)
    f=1;
for i=k:4:k+15
    inter_storage(f,:)=out_coded(i:1:i+3);
    f=f+1;
end
temp1=reshape(inter_storage,[],1);
out_inter1rep=cat(1,out_inter1rep,temp1);
end
stream2=out_inter1rep';
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%QPSK TRANSMITTER for rep
E0_QPSK=Eb/3;
QPSK_tx2=zeros(length(out_coded)/2,1,length(Eb));
for k=1:2:length(out_coded)
    if(isequal(stream2(k:k+1),[0 0]))
        QPSK_tx2((k+1)/2,1,:)=-1*sqrt(E0_QPSK)-1i*sqrt(E0_QPSK);
    elseif(isequal(stream2(k:k+1),[1 0]))
        QPSK_tx2((k+1)/2,1,:)=1*sqrt(E0_QPSK)-1i*sqrt(E0_QPSK);
    elseif(isequal(stream2(k:k+1),[1 1]))
       QPSK_tx2((k+1)/2,1,:)=1*sqrt(E0_QPSK)+1i*sqrt(E0_QPSK);
    else
        QPSK_tx2((k+1)/2,1,:)=-1*sqrt(E0_QPSK)+1i*sqrt(E0_QPSK);
    end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%ADDING NOISE TO TRANSMITTED SIGNALS
No=2;
v1_uncoded2 = randn(length(out_coded)/2,1,length(Eb)).* sqrt(No/2);
v2_uncoded2 = 1i.* randn(length(out_coded)/2,1,length(Eb)).* sqrt(No/2);
channel2 = sqrt(pow2(v1_uncoded2) + pow2(v2_uncoded2)) / sqrt(2);
noise2=
sqrt(No/2)*randn(length(out_coded)/2,1,length(Eb))+1j*sqrt(No/2).*randn(length(out_code
d)/2,1,length(Eb));
y_QPSK2=channel2.*QPSK_tx2+noise2;
y_QPSK2=y_QPSK2./channel2;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%QPSK RECEIVER
rec_QPSK2=zeros(length(out_coded),1,length(Eb));
for h=1:length(Eb)
for k=1:length(out_coded)/2
    if(real(y_QPSK2(k,1,h))>0 )
        b0=1;
    else
        b0=0;
    end
    if(imag(y_QPSK2(k,1,h))>0 )
       b1=1;
    else
        b1=0;
    end
    rec_QPSK2((k*2)-1,1,h)=b0;
    rec_QPSK2((k*2),1,h)=b1;
end
end
%%%%%%%%%%%%%De-Interleaver%%%%%%%%%%%%%%%%%%%
inter_storage=zeros(4,4);
out_inter2rep=zeros(length(out_coded),1,length(Eb));
iwant=zeros(num_bits,1,15);
for q=1:1:length(Eb)
vec2=[];
for k=1:16:size(out_coded,1)
    f=1;
for i=k:4:k+15
    inter_storage(f,:)=rec_QPSK2(i:1:i+3,1,q);
    f=f+1;
end
temp1=reshape(inter_storage,[],1);
vec2=cat(1,vec2,temp1);
end
out_inter2rep(:,1,q)=vec2;
for z=1:3:length(out_inter2rep)
iwant((z+2)/3,1,q)=mode(out_inter2rep(z:z+2,1,q));
end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%ERROR FOR QPSK
sum_error_QPSK2=zeros(length(Eb),1);
for h=1:length(Eb)
for k=1:num_bits
    if(iwant(k,1,h)~=stream_trans(k,1))
        sum_error_QPSK2(h,1)=sum_error_QPSK2(h,1)+1;
    end
end
```

```
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%BER CALC FOR ALL SIGNALS
BER_QPSK2=sum_error_QPSK2/num_bits;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
x_axis=10*log10(Eb/2);%No=2
x_axis2=10*log10(Eb/2);%No=2
slg=semilogy(x_axis,BER_QPSK,'r',x_axis2,BER_QPSK2,'b');
slg(1).LineWidth=2;slg(2).LineWidth=2;
legend('QPSK','QPSK rep')
xlabel('Eb/No');
ylabel('BER');
title('BER vs Eb/No for QPSK');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%QAM SIGNAL%%%%%%%%%
Eb=1:15;
E0_QAM=Eb;
No=2;
out_norep=repelem(stream_trans,1);
out_coded=repelem(stream_trans,3);
%%%%%%%%%%%Interleaver%%%%%%%%%
inter_storage=zeros(4,4);
out_inter1=[];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% for no rep
for k=1:16:size(out_norep,1)
    f=1;
for i=k:4:k+15
    inter_storage(f,:)=out_norep(i:1:i+3);
    f=f+1;
end
temp1=reshape(inter_storage,[],1);
out_inter1=cat(1,out_inter1,temp1);
end
stream=out_inter1';
%%%%%%%%%%QAM%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%E0_QAM=1;
QAM_tx=zeros(num_bits/4,1,length(Eb));
for k=1:4:num_bits
    if(isequal(stream(k:k+3),[0 0 0 0]))
        QAM_tx((k+3)/4,1,:)=-3*sqrt(E0_QAM)-3i*sqrt(E0_QAM);
    elseif(isequal(stream(k:k+3),[0 1 0 0]))
        QAM_tx((k+3)/4,1,:)=-1*sqrt(E0_QAM)-3i*sqrt(E0_QAM);
    elseif(isequal(stream(k:k+3),[1 1 0 0]))
        QAM_tx((k+3)/4,1,:)=1*sqrt(E0_QAM)-3i*sqrt(E0_QAM);
    elseif(isequal(stream(k:k+3),[1 0 0 0]))
        QAM_tx((k+3)/4,1,:)=3*sqrt(E0_QAM)-3i*sqrt(E0_QAM);
    elseif(isequal(stream(k:k+3),[1 0 0 1]))
        QAM_tx((k+3)/4,1,:)=3*sqrt(E0_QAM)-1i*sqrt(E0_QAM);
    elseif(isequal(stream(k:k+3),[1 1 0 1]))
        QAM_tx((k+3)/4,1,:)=1*sqrt(E0_QAM)-1i*sqrt(E0_QAM);
    elseif(isequal(stream(k:k+3),[0 1 0 1]))
        QAM_tx((k+3)/4,1,:)=-1*sqrt(E0_QAM)-1i*sqrt(E0_QAM);
    elseif(isequal(stream(k:k+3),[0 0 0 1]))
        QAM_tx((k+3)/4,1,:)=-3*sqrt(E0_QAM)-1i*sqrt(E0_QAM);
    elseif(isequal(stream(k:k+3),[0 0 1 1]))
        QAM_tx((k+3)/4,1,:)=-3*sqrt(E0_QAM)+1i*sqrt(E0_QAM);
    elseif(isequal(stream(k:k+3),[0 1 1 1]))
        QAM_tx((k+3)/4,1,:)=-1*sqrt(E0_QAM)+1i*sqrt(E0_QAM);
    elseif(isequal(stream(k:k+3),[1 1 1 1]))
        QAM_tx((k+3)/4,1,:)=1*sqrt(E0_QAM)+1i*sqrt(E0_QAM);
    elseif(isequal(stream(k:k+3),[1 0 1 1]))
        QAM_tx((k+3)/4,1,:)=3*sqrt(E0_QAM)+1i*sqrt(E0_QAM);
    elseif(isequal(stream(k:k+3),[1 0 1 0]))
        QAM_tx((k+3)/4,1,:)=3*sqrt(E0_QAM)+3i*sqrt(E0_QAM);
    elseif(isequal(stream(k:k+3),[1 1 1 0]))
        QAM_tx((k+3)/4,1,:)=1*sqrt(E0_QAM)+3i*sqrt(E0_QAM);
    elseif(isequal(stream(k:k+3),[0 1 1 0]))
        QAM_tx((k+3)/4,1,:)=-1*sqrt(E0_QAM)+3i*sqrt(E0_QAM);
    else
        QAM_tx((k+3)/4,1,:)=-3*sqrt(E0_QAM)+3i*sqrt(E0_QAM);
    end
end
%%%%%%%%%%%%%%%%%%%%%%%%%channel 1%%%%%%%%%%%%%%%%%%%%%%%%%
v1_uncoded = randn(num_bits/4,1,length(Eb)).* sqrt(No/2);
v2_uncoded = 1i.* randn(num_bits/4,1,length(Eb)).* sqrt(No/2);
channel = sqrt(pow2(v1_uncoded) + pow2(v2_uncoded)) / sqrt(2);
noise=
sqrt(No/2)*randn(num_bits/4,1,length(Eb))+1j*sqrt(No/2).*randn(num_bits/4,1,length(Eb))
;
%y_QPSK=QPSK_tx+randn(num_bits/2,1,length(Eb))+1i*randn(num_bits/2,1,length(Eb));
y_QAM=channel.*QAM_tx+noise;
y_QAM=y_QAM./channel;
%%%%%%%%%%%%%%%%%%%%%%Receiver%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
rec_QAM=zeros(num_bits,1,length(Eb));
for h=1:length(Eb)
for k=1:num_bits/4
    if(real(y_QAM(k,1,h))>2*sqrt(E0_QAM(1,h)))
        rec_QAM((k*4)-3,1,h)=1;
        rec_QAM((k*4)-2,1,h)=0;
    elseif(real(y_QAM(k,1,h))>0 && real(y_QAM(k,1,h))<2*sqrt(E0_QAM(1,h)))
        rec_QAM((k*4)-3,1,h)=1;
        rec_QAM((k*4)-2,1,h)=1;
     elseif(real(y_QAM(k,1,h))<0 && real(y_QAM(k,1,h))>-2*sqrt(E0_QAM(1,h)))
        rec_QAM((k*4)-3,1,h)=0;
        rec_QAM((k*4)-2,1,h)=1;
    else
        rec_QAM((k*4)-3,1,h)=0;
        rec_QAM((k*4)-2,1,h)=0;
        b1=0;
    end
        if(imag(y_QAM(k,1,h))>2*sqrt(E0_QAM(1,h)))
        rec_QAM((k*4)-1,1,h)=1;
        rec_QAM((k*4),1,h)=0;
    elseif(imag(y_QAM(k,1,h))>0 && imag(y_QAM(k,1,h))<2*sqrt(E0_QAM(1,h)))
        rec_QAM((k*4)-1,1,h)=1;
        rec_QAM((k*4),1,h)=1;
     elseif(imag(y_QAM(k,1,h))<0 && imag(y_QAM(k,1,h))>-2*sqrt(E0_QAM(1,h)))
        rec_QAM((k*4)-1,1,h)=0;
        rec_QAM((k*4),1,h)=1;
    else
        rec_QAM((k*4)-1,1,h)=0;
        rec_QAM((k*4),1,h)=0;
        b1=0;
        end
end
end
```

```matlab
%%%%%%%%%%%%%De interleeaver%%%%%%%%%%%%%%%%%
inter_storage=zeros(4,4);
out_inter2=zeros(length(out_norep),1,length(Eb));
for q=1:1:length(Eb)
vec=[];
for k=1:16:size(out_norep,1)
    f=1;
for i=k:4:k+15
    inter_storage(f,:)=rec_QAM(i:1:i+3,1,q);
    f=f+1;
end
temp1=reshape(inter_storage,[],1);
vec=cat(1,vec,temp1);
end
out_inter2(:,1,q)=vec;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%ERROR FOR QPSK
sum_error_QAM=zeros(length(Eb),1);
for h=1:length(Eb)
for k=1:num_bits
    if(out_inter2(k,1,h)~=stream_trans(k,1))
        sum_error_QAM(h,1)=sum_error_QAM(h,1)+1;
    end
end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%BER CALC FOR ALL SIGNALS
BER_QAM=sum_error_QAM/num_bits;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
inter_storage=zeros(4,4);
out_inter1rep=[];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% for rep
for k=1:16:size(out_coded,1)
    f=1;
for i=k:4:k+15
    inter_storage(f,:)=out_coded(i:1:i+3);
    f=f+1;
end
temp1=reshape(inter_storage,[],1);
out_inter1rep=cat(1,out_inter1rep,temp1);
end
stream2=out_inter1rep';
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%QPSK TRANSMITTER
E0_QAM=Eb/3;
QAM_tx2=zeros(length(out_coded)/4,1,length(Eb));
for k=1:4:length(out_coded)
    if(isequal(stream2(k:k+3),[0 0 0 0]))
        QAM_tx2((k+3)/4,1,:)=-3*sqrt(E0_QAM)-3i*sqrt(E0_QAM);
    elseif(isequal(stream2(k:k+3),[0 1 0 0]))
        QAM_tx2((k+3)/4,1,:)=-1*sqrt(E0_QAM)-3i*sqrt(E0_QAM);
    elseif(isequal(stream2(k:k+3),[1 1 0 0]))
        QAM_tx2((k+3)/4,1,:)=1*sqrt(E0_QAM)-3i*sqrt(E0_QAM);
    elseif(isequal(stream2(k:k+3),[1 0 0 0]))
        QAM_tx2((k+3)/4,1,:)=3*sqrt(E0_QAM)-3i*sqrt(E0_QAM);
    elseif(isequal(stream2(k:k+3),[1 0 0 1]))
        QAM_tx2((k+3)/4,1,:)=3*sqrt(E0_QAM)-1i*sqrt(E0_QAM);
    elseif(isequal(stream2(k:k+3),[1 1 0 1]))
        QAM_tx2((k+3)/4,1,:)=1*sqrt(E0_QAM)-1i*sqrt(E0_QAM);
    elseif(isequal(stream2(k:k+3),[0 1 0 1]))
        QAM_tx2((k+3)/4,1,:)=-1*sqrt(E0_QAM)-1i*sqrt(E0_QAM);
    elseif(isequal(stream2(k:k+3),[0 0 0 1]))
        QAM_tx2((k+3)/4,1,:)=-3*sqrt(E0_QAM)-1i*sqrt(E0_QAM);
    elseif(isequal(stream2(k:k+3),[0 0 1 1]))
```

```
QAM_tx2((k+3)/4,1,:)=-3*sqrt(E0_QAM)+1i*sqrt(E0_QAM);
elseif(isequal(stream2(k:k+3),[0 1 1 1]))
        QAM_tx2((k+3)/4,1,:)=-1*sqrt(E0_QAM)+1i*sqrt(E0_QAM);
    elseif(isequal(stream2(k:k+3),[1 1 1 1]))
        QAM_tx2((k+3)/4,1,:)=1*sqrt(E0_QAM)+1i*sqrt(E0_QAM);
    elseif(isequal(stream2(k:k+3),[1 0 1 1]))
        QAM_tx2((k+3)/4,1,:)=3*sqrt(E0_QAM)+1i*sqrt(E0_QAM);
    elseif(isequal(stream2(k:k+3),[1 0 1 0]))
        QAM_tx2((k+3)/4,1,:)=3*sqrt(E0_QAM)+3i*sqrt(E0_QAM);
    elseif(isequal(stream2(k:k+3),[1 1 1 0]))
        QAM_tx2((k+3)/4,1,:)=1*sqrt(E0_QAM)+3i*sqrt(E0_QAM);
    elseif(isequal(stream2(k:k+3),[0 1 1 0]))
        QAM_tx2((k+3)/4,1,:)=-1*sqrt(E0_QAM)+3i*sqrt(E0_QAM);
    else
        QAM_tx2((k+3)/4,1,:)=-3*sqrt(E0_QAM)+3i*sqrt(E0_QAM);
    end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
v1_uncoded2 = randn(length(out_coded)/4,1,length(Eb)).* sqrt(No/2);
v2_uncoded2 = 1i.* randn(length(out_coded)/4,1,length(Eb)).* sqrt(No/2);
channel2 = sqrt(pow2(v1_uncoded2) + pow2(v2_uncoded2)) / sqrt(2);
noise2=
sqrt(No/2)*randn(length(out_coded)/4,1,length(Eb))+1j*sqrt(No/2).*randn(length(out_code
d)/4,1,length(Eb));
%y_QPSK=QPSK_tx+randn(length(out_coded)/2,1,length(Eb))+1i*randn(length(out_coded)/2,1,
length(Eb));
y_QAM2=channel2.*QAM_tx2+noise2;
y_QAM2=y_QAM2./channel2;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
rec_QAM2=zeros(length(out_coded),1,length(Eb));
for h=1:length(Eb)
for k=1:length(out_coded)/4
    if(real(y_QAM2(k,1,h))>2*sqrt(E0_QAM(1,h)))
        rec_QAM2((k*4)-3,1,h)=1;
        rec_QAM2((k*4)-2,1,h)=0;
    elseif(real(y_QAM2(k,1,h))>0 && real(y_QAM2(k,1,h))<2*sqrt(E0_QAM(1,h)))
        rec_QAM2((k*4)-3,1,h)=1;
        rec_QAM2((k*4)-2,1,h)=1;
     elseif(real(y_QAM2(k,1,h))<0 && real(y_QAM2(k,1,h))>-2*sqrt(E0_QAM(1,h)))
        rec_QAM2((k*4)-3,1,h)=0;
        rec_QAM2((k*4)-2,1,h)=1;
    else
        rec_QAM2((k*4)-3,1,h)=0;
        rec_QAM2((k*4)-2,1,h)=0;
        b1=0;
    end
        if(imag(y_QAM2(k,1,h))>2*sqrt(E0_QAM(1,h)))
        rec_QAM2((k*4)-1,1,h)=1;
        rec_QAM2((k*4),1,h)=0;
    elseif(imag(y_QAM2(k,1,h))>0 && imag(y_QAM2(k,1,h))<2*sqrt(E0_QAM(1,h)))
        rec_QAM2((k*4)-1,1,h)=1;
        rec_QAM2((k*4),1,h)=1;
     elseif(imag(y_QAM2(k,1,h))<0 && imag(y_QAM2(k,1,h))>-2*sqrt(E0_QAM(1,h)))
        rec_QAM2((k*4)-1,1,h)=0;
        rec_QAM2((k*4),1,h)=1;
    else
        rec_QAM2((k*4)-1,1,h)=0;
        rec_QAM2((k*4),1,h)=0;
        b1=0;
        end
end
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%De-Interleaver%%%%%%%%%%%%%%%%%%
inter_storage=zeros(4,4);
out_inter2rep=zeros(length(out_coded),1,length(Eb));
iwant=zeros(num_bits,1,15);
for q=1:1:length(Eb)
vec2=[];
for k=1:16:size(out_coded,1)
    f=1;
for i=k:4:k+15
    inter_storage(f,:)=rec_QAM2(i:1:i+3,1,q);
    f=f+1;
end
temp1=reshape(inter_storage,[],1);
vec2=cat(1,vec2,temp1);
end
out_inter2rep(:,1,q)=vec2;
for z=1:3:length(out_inter2rep)
iwant((z+2)/3,1,q)=mode(out_inter2rep(z:z+2,1,q));
end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%ERROR FOR QPSK
sum_error_QAM2=zeros(length(Eb),1);
for h=1:length(Eb)
for k=1:num_bits
    if(iwant(k,1,h)~=stream_trans(k,1))
        sum_error_QAM2(h,1)=sum_error_QAM2(h,1)+1;
    end
end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%BER CALC FOR ALL SIGNALS
BER_QAM2=sum_error_QAM2/num_bits;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
x_axis=10*log10(Eb/2);%No=2
x_axis2=10*log10(Eb/2);%No=2
slg=semilogy(x_axis,BER_QAM,'r',x_axis2,BER_QAM2,'b');
slg(1).LineWidth=2;slg(2).LineWidth=2;
hold on
legend('QAM','QAM rep')
xlabel('Eb/No');
ylabel('BER');
title('BER vs Eb/No for QAM');
```

OFDM system

Code:

```
rng(5000) ;
% no coding %
x_nocodeQPSK = randi([0 1],12800,1);
x_nocode16QAM = randi([0 1],25600,1);
x_torepQPSK = randi([0 1],4200,1) ;
x_repQPSK = repelem(x_torepQPSK,3) ;
x_torep16QAM=randi([0 1],8500,1) ;
x_rep16QAM = repelem(x_torep16QAM,3) ;
for i=1:100
%Repetition Code%
x_repzpQPSK(((i-1)*128)+1:(i*128),1) =[0 ;0 ;x_repQPSK(((i-1)*126)+1:(i*126),1)];
x_repzp16QAM(((i-1)*256)+1:(i*256),1) = [0 ;x_rep16QAM(((i-1)*255)+1:(i*255),1)] ;
%QPSK interleaver %
intleav_QPSK_nocode=(reshape(x_nocodeQPSK(((i-1)*128)+1:(i*128),1),[16,8]))';
out_intleav_QPSK_nocode(((i-1)*128)+1:(i*128),1)=reshape(intleav_QPSK_nocode,128,1) ;
intleav_QPSK_rep =(reshape(x_repzpQPSK(((i-1)*128)+1:(i*128),1),[16,8]))' ;
out_intleav_QPSK_rep(((i-1)*128)+1:(i*128),1)=reshape(intleav_QPSK_rep,128,1) ;
%16QAM interleaver %
intleav_16QAM_nocode=(reshape(x_nocode16QAM(((i-1)*256)+1:(i*256),1),[16,16]))';
out_intleav_16QAM_nocode(((i-1)*256)+1:(i*256),1)=reshape(intleav_16QAM_nocode,256,1);
intleav_16QAM_rep=(reshape(x_repzp16QAM(((i-1)*256)+1:(i*256),1),[16,16]))';
out_intleav_16QAM_rep(((i-1)*256)+1:(i*256),1)=reshape(intleav_16QAM_rep,256,1);
end
Eb_nocode=1:30 ;
Eb_rep=1/3:1/3:10 ;
%%QPSK no coding%%
s_qpsk_nocode=zeros(6400,30) ;
for i=1:2:12800
if out_intleav_QPSK_nocode(i,1)==0 && out_intleav_QPSK_nocode(i+1,1)==0
    s_qpsk_nocode((i+1)/2,:)=(-1-j)*sqrt(Eb_nocode) ;
elseif out_intleav_QPSK_nocode(i,1)==0 && out_intleav_QPSK_nocode(i+1,1)==1
    s_qpsk_nocode((i+1)/2,:)=(-1+j)*sqrt(Eb_nocode) ;
elseif out_intleav_QPSK_nocode(i,1)==1 && out_intleav_QPSK_nocode(i+1,1)==1
    s_qpsk_nocode((i+1)/2,:)=(1+j)*sqrt(Eb_nocode) ;
else
    s_qpsk_nocode((i+1)/2,:)=(1-j)*sqrt(Eb_nocode) ;
end
end
%%QPSK repetition code%%
s_qpsk_rep=zeros(6400,30) ;
for i=1:2:12800
if out_intleav_QPSK_rep(i,1)==0 && out_intleav_QPSK_rep(i+1,1)==0
    s_qpsk_rep((i+1)/2,:)=(-1-j)*sqrt(Eb_rep) ;
elseif out_intleav_QPSK_rep(i,1)==0 && out_intleav_QPSK_rep(i+1,1)==1
    s_qpsk_rep((i+1)/2,:)=(-1+j)*sqrt(Eb_rep) ;
elseif out_intleav_QPSK_rep(i,1)==1 && out_intleav_QPSK_rep(i+1,1)==1
    s_qpsk_rep((i+1)/2,:)=(1+j)*sqrt(Eb_rep) ;
else
    s_qpsk_rep((i+1)/2,:)=(1-j)*sqrt(Eb_rep) ;
end
end
```

```
%%16QAM no coding%%
s_qam_nocode=zeros(6400,30) ;
for i=1:4:25600
if (out_intleav_16QAM_nocode(i,1)==0 && out_intleav_16QAM_nocode(i+1,1)==0 &&
out_intleav_16QAM_nocode(i+2,1)==0 ...
    && out_intleav_16QAM_nocode(i+3,1)==0)
    s_qam_nocode((i+3)/4,:)=(-3-3j)*sqrt(Eb_nocode/2.5) ;
elseif (out_intleav_16QAM_nocode(i,1)==0 && out_intleav_16QAM_nocode(i+1,1)==0 &&
out_intleav_16QAM_nocode(i+2,1)==0 ...
    && out_intleav_16QAM_nocode(i+3,1)==1)
    s_qam_nocode((i+3)/4,:)=(-3-j)*sqrt(Eb_nocode/2.5) ;
elseif (out_intleav_16QAM_nocode(i,1)==0 && out_intleav_16QAM_nocode(i+1,1)==0 &&
out_intleav_16QAM_nocode(i+2,1)==1 ...
    && out_intleav_16QAM_nocode(i+3,1)==0)
    s_qam_nocode((i+3)/4,:)=(-3+3j)*sqrt(Eb_nocode/2.5) ;
elseif (out_intleav_16QAM_nocode(i,1)==0 && out_intleav_16QAM_nocode(i+1,1)==0 &&
out_intleav_16QAM_nocode(i+2,1)==1 ...
    && out_intleav_16QAM_nocode(i+3,1)==1)
    s_qam_nocode((i+3)/4,:)=(-3+j)*sqrt(Eb_nocode/2.5) ;
elseif (out_intleav_16QAM_nocode(i,1)==0 && out_intleav_16QAM_nocode(i+1,1)==1 &&
out_intleav_16QAM_nocode(i+2,1)==0 ...
    && out_intleav_16QAM_nocode(i+3,1)==0)
    s_qam_nocode((i+3)/4,:)=(-1-3j)*sqrt(Eb_nocode/2.5) ;
elseif (out_intleav_16QAM_nocode(i,1)==0 && out_intleav_16QAM_nocode(i+1,1)==1 &&
out_intleav_16QAM_nocode(i+2,1)==0 ...
    && out_intleav_16QAM_nocode(i+3,1)==1)
    s_qam_nocode((i+3)/4,:)=(-1-1j)*sqrt(Eb_nocode/2.5) ;
elseif (out_intleav_16QAM_nocode(i,1)==0 && out_intleav_16QAM_nocode(i+1,1)==1 &&
out_intleav_16QAM_nocode(i+2,1)==1 ...
    && out_intleav_16QAM_nocode(i+3,1)==0)
    s_qam_nocode((i+3)/4,:)=(-1+3j)*sqrt(Eb_nocode/2.5) ;
elseif (out_intleav_16QAM_nocode(i,1)==0 && out_intleav_16QAM_nocode(i+1,1)==1 &&
out_intleav_16QAM_nocode(i+2,1)==1 ...
    && out_intleav_16QAM_nocode(i+3,1)==1)
    s_qam_nocode((i+3)/4,:)=(-1+j)*sqrt(Eb_nocode/2.5) ;
elseif (out_intleav_16QAM_nocode(i,1)==1 && out_intleav_16QAM_nocode(i+1,1)==0 &&
out_intleav_16QAM_nocode(i+2,1)==0 ...
    && out_intleav_16QAM_nocode(i+3,1)==0)
    s_qam_nocode((i+3)/4,:)=(3-3j)*sqrt(Eb_nocode/2.5) ;
elseif (out_intleav_16QAM_nocode(i,1)==1 && out_intleav_16QAM_nocode(i+1,1)==0 &&
out_intleav_16QAM_nocode(i+2,1)==0 ...
    && out_intleav_16QAM_nocode(i+3,1)==1)
    s_qam_nocode((i+3)/4,:)=(3-j)*sqrt(Eb_nocode/2.5) ;
elseif (out_intleav_16QAM_nocode(i,1)==1 && out_intleav_16QAM_nocode(i+1,1)==0 &&
out_intleav_16QAM_nocode(i+2,1)==1 ...
    && out_intleav_16QAM_nocode(i+3,1)==0)
    s_qam_nocode((i+3)/4,:)=(3+3j)*sqrt(Eb_nocode/2.5) ;
elseif (out_intleav_16QAM_nocode(i,1)==1 && out_intleav_16QAM_nocode(i+1,1)==0 &&
out_intleav_16QAM_nocode(i+2,1)==1 ...
    && out_intleav_16QAM_nocode(i+3,1)==1)
    s_qam_nocode((i+3)/4,:)=(3+j)*sqrt(Eb_nocode/2.5) ;
elseif (out_intleav_16QAM_nocode(i,1)==1 && out_intleav_16QAM_nocode(i+1,1)==1 &&
out_intleav_16QAM_nocode(i+2,1)==0 ...
    && out_intleav_16QAM_nocode(i+3,1)==0)
    s_qam_nocode((i+3)/4,:)=(1-3j)*sqrt(Eb_nocode/2.5) ;
elseif (out_intleav_16QAM_nocode(i,1)==1 && out_intleav_16QAM_nocode(i+1,1)==1 &&
out_intleav_16QAM_nocode(i+2,1)==0 ...
    && out_intleav_16QAM_nocode(i+3,1)==1)
    s_qam_nocode((i+3)/4,:)=(1-j)*sqrt(Eb_nocode/2.5) ;
```

```
elseif (out_intleav_16QAM_nocode(i,1)==1 && out_intleav_16QAM_nocode(i+1,1)==1 &&
out_intleav_16QAM_nocode(i+2,1)==1 ...
    && out_intleav_16QAM_nocode(i+3,1)==0)
    s_qam_nocode((i+3)/4,:)=(1+3j)*sqrt(Eb_nocode/2.5) ;
elseif (out_intleav_16QAM_nocode(i,1)==1 && out_intleav_16QAM_nocode(i+1,1)==1 &&
out_intleav_16QAM_nocode(i+2,1)==1 ...
    && out_intleav_16QAM_nocode(i+3,1)==1)
    s_qam_nocode((i+3)/4,:)=(1+j)*sqrt(Eb_nocode/2.5) ;
end
end
%%16QAM repetition code%%
s_qam_rep=zeros(6400,30) ;
for i=1:4:25600
if (out_intleav_16QAM_rep(i,1)==0 && out_intleav_16QAM_rep(i+1,1)==0 &&
out_intleav_16QAM_rep(i+2,1)==0 ...
    && out_intleav_16QAM_rep(i+3,1)==0)
    s_qam_rep((i+3)/4,:)=(-3-3j)*sqrt(Eb_rep/2.5) ;
elseif (out_intleav_16QAM_rep(i,1)==0 && out_intleav_16QAM_rep(i+1,1)==0 &&
out_intleav_16QAM_rep(i+2,1)==0 ...
    && out_intleav_16QAM_rep(i+3,1)==1)
    s_qam_rep((i+3)/4,:)=(-3-j)*sqrt(Eb_rep/2.5) ;
elseif (out_intleav_16QAM_rep(i,1)==0 && out_intleav_16QAM_rep(i+1,1)==0 &&
out_intleav_16QAM_rep(i+2,1)==1 ...
    && out_intleav_16QAM_rep(i+3,1)==0)
    s_qam_rep((i+3)/4,:)=(-3+3j)*sqrt(Eb_rep/2.5) ;
elseif (out_intleav_16QAM_rep(i,1)==0 && out_intleav_16QAM_rep(i+1,1)==0 &&
out_intleav_16QAM_rep(i+2,1)==1 ...
    && out_intleav_16QAM_rep(i+3,1)==1)
    s_qam_rep((i+3)/4,:)=(-3+j)*sqrt(Eb_rep/2.5) ;
elseif (out_intleav_16QAM_rep(i,1)==0 && out_intleav_16QAM_rep(i+1,1)==1 &&
out_intleav_16QAM_rep(i+2,1)==0 ...
    && out_intleav_16QAM_rep(i+3,1)==0)
    s_qam_rep((i+3)/4,:)=(-1-3j)*sqrt(Eb_rep/2.5) ;
elseif (out_intleav_16QAM_rep(i,1)==0 && out_intleav_16QAM_rep(i+1,1)==1 &&
out_intleav_16QAM_rep(i+2,1)==0 ...
    && out_intleav_16QAM_rep(i+3,1)==1)
    s_qam_rep((i+3)/4,:)=(-1-1j)*sqrt(Eb_rep/2.5) ;
elseif (out_intleav_16QAM_rep(i,1)==0 && out_intleav_16QAM_rep(i+1,1)==1 &&
out_intleav_16QAM_rep(i+2,1)==1 ...
    && out_intleav_16QAM_rep(i+3,1)==0)
    s_qam_rep((i+3)/4,:)=(-1+3j)*sqrt(Eb_rep/2.5) ;
elseif (out_intleav_16QAM_rep(i,1)==0 && out_intleav_16QAM_rep(i+1,1)==1 &&
out_intleav_16QAM_rep(i+2,1)==1 ...
    && out_intleav_16QAM_rep(i+3,1)==1)
    s_qam_rep((i+3)/4,:)=(-1+j)*sqrt(Eb_rep/2.5) ;
elseif (out_intleav_16QAM_rep(i,1)==1 && out_intleav_16QAM_rep(i+1,1)==0 &&
out_intleav_16QAM_rep(i+2,1)==0 ...
    && out_intleav_16QAM_rep(i+3,1)==0)
    s_qam_rep((i+3)/4,:)=(3-3j)*sqrt(Eb_rep/2.5) ;
elseif (out_intleav_16QAM_rep(i,1)==1 && out_intleav_16QAM_rep(i+1,1)==0 &&
out_intleav_16QAM_rep(i+2,1)==0 ...
    && out_intleav_16QAM_rep(i+3,1)==1)
    s_qam_rep((i+3)/4,:)=(3-j)*sqrt(Eb_rep/2.5) ;
elseif (out_intleav_16QAM_rep(i,1)==1 && out_intleav_16QAM_rep(i+1,1)==0 &&
out_intleav_16QAM_rep(i+2,1)==1 ...
    && out_intleav_16QAM_rep(i+3,1)==0)
    s_qam_rep((i+3)/4,:)=(3+3j)*sqrt(Eb_rep/2.5) ;
elseif (out_intleav_16QAM_rep(i,1)==1 && out_intleav_16QAM_rep(i+1,1)==0 &&
out_intleav_16QAM_rep(i+2,1)==1 ...
    && out_intleav_16QAM_rep(i+3,1)==1)
    s_qam_rep((i+3)/4,:)=(3+j)*sqrt(Eb_rep/2.5) ;
```

```
elseif (out_intleav_16QAM_rep(i,1)==1 && out_intleav_16QAM_rep(i+1,1)==1 &&
out_intleav_16QAM_rep(i+2,1)==0 ...
    && out_intleav_16QAM_rep(i+3,1)==0)
    s_qam_rep((i+3)/4,:)=(1-3j)*sqrt(Eb_rep/2.5) ;
elseif (out_intleav_16QAM_rep(i,1)==1 && out_intleav_16QAM_rep(i+1,1)==1 &&
out_intleav_16QAM_rep(i+2,1)==0 ...
    && out_intleav_16QAM_rep(i+3,1)==1)
    s_qam_rep((i+3)/4,:)=(1-j)*sqrt(Eb_rep/2.5) ;
elseif (out_intleav_16QAM_rep(i,1)==1 && out_intleav_16QAM_rep(i+1,1)==1 &&
out_intleav_16QAM_rep(i+2,1)==1 ...
    && out_intleav_16QAM_rep(i+3,1)==0)
    s_qam_rep((i+3)/4,:)=(1+3j)*sqrt(Eb_rep/2.5) ;
elseif (out_intleav_16QAM_rep(i,1)==1 && out_intleav_16QAM_rep(i+1,1)==1 &&
out_intleav_16QAM_rep(i+2,1)==1 ...
    && out_intleav_16QAM_rep(i+3,1)==1)
    s_qam_rep((i+3)/4,:)=(1+j)*sqrt(Eb_rep/2.5) ;
end
end
%%IFFT%%
for k=1:30
    for i=1:100
%%ifft QPSK no coding %%
        out_ifft_QPSK_nocode(((i-1)*64)+1:(i*64),k)=ifft(s_qpsk_nocode(((i-
1)*64)+1:(i*64),k)) ;
%%ifft QPSK repetition coding %%
        out_ifft_QPSK_rep(((i-1)*64)+1:(i*64),k)=ifft(s_qpsk_rep(((i-
1)*64)+1:(i*64),k)) ;
%%ifft 16QAM no coding%%
        out_ifft_16QAM_nocode(((i-1)*64)+1:(i*64),k)=ifft(s_qam_nocode(((i-
1)*64)+1:(i*64),k)) ;
%%ifft 16QAM repetition coding%%
        out_ifft_16QAM_rep(((i-1)*64)+1:(i*64),k)=ifft(s_qam_rep(((i-
1)*64)+1:(i*64),k)) ;
    end
end
%%Cyclic extension%%
for k=1:30
    for i=1:100
        cyclic_out_ifft_QPSK_nocode(((i-
1)*80)+1:(i*80),k)=[out_ifft_QPSK_nocode((i*64)-15:(i*64),k);out_ifft_QPSK_nocode(((i-
1)*64)+1:(i*64),k)] ;
        cyclic_out_ifft_QPSK_rep(((i-1)*80)+1:(i*80),k)=[out_ifft_QPSK_rep((i*64)-
15:(i*64),k);out_ifft_QPSK_rep(((i-1)*64)+1:(i*64),k)] ;
        cyclic_out_ifft_16QAM_nocode(((i-
1)*80)+1:(i*80),k)=[out_ifft_16QAM_nocode((i*64)-
15:(i*64),k);out_ifft_16QAM_nocode(((i-1)*64)+1:(i*64),k)] ;
        cyclic_out_ifft_16QAM_rep(((i-1)*80)+1:(i*80),k)=[out_ifft_16QAM_rep((i*64)-
15:(i*64),k);out_ifft_16QAM_rep(((i-1)*64)+1:(i*64),k)] ;
    end
end
%%%%% Signal after AWGN channel %%%%%
No=2 ;
noise=(sqrt(No/2))*randn(8000,1)+j*((sqrt(No/2))*randn(8000,1)) ;
noise=repmat(noise,1,30) ;
awgn_X_qpsk_nocode=cyclic_out_ifft_QPSK_nocode+noise ;
awgn_X_qpsk_rep=cyclic_out_ifft_QPSK_rep+noise ;
awgn_X_qam_nocode=cyclic_out_ifft_16QAM_nocode+noise ;
awgn_X_qam_rep=cyclic_out_ifft_16QAM_rep+noise ;
```

```
%%%%% Signal after Frequency selective Fading channel %%%%%
h=[0.4 ;0 ;0.26 ;0 ;0 ;0.4 ;0 ;0.6 ;0 ;0.5];
for k=1:30
    for i=1:100
        freqs_X_qpsk_nocode(((i-1)*89)+1:(i*89),k)=conv(awgn_X_qpsk_nocode(((i-
1)*80)+1:(i*80),k),h) ;
        freqs_X_qpsk_rep(((i-1)*89)+1:(i*89),k)=conv(awgn_X_qpsk_rep(((i-
1)*80)+1:(i*80),k),h) ;
        freqs_X_qam_nocode(((i-1)*89)+1:(i*89),k)=conv(awgn_X_qam_nocode(((i-
1)*80)+1:(i*80),k),h) ;
        freqs_X_qam_rep(((i-1)*89)+1:(i*89),k)=conv(awgn_X_qam_rep(((i-
1)*80)+1:(i*80),k),h) ;
    end
end
%%Receiver%%
for k=1:30
    for i=1:100
        rec_signal_freqs_qpsk_nocode(((i-
1)*80)+1:(i*80),k)=deconv(freqs_X_qpsk_nocode(((i-1)*89)+1:(i*89),k),h) ;
        rec_signal_freqs_qpsk_rep(((i-1)*80)+1:(i*80),k)=deconv(freqs_X_qpsk_rep(((i-
1)*89)+1:(i*89),k),h);
        rec_signal_freqs_qam_nocode(((i-
1)*80)+1:(i*80),k)=deconv(freqs_X_qam_nocode(((i-1)*89)+1:(i*89),k),h);
        rec_signal_freqs_qam_rep(((i-1)*80)+1:(i*80),k)=deconv(freqs_X_qam_rep(((i-
1)*89)+1:(i*89),k),h);
    end
end
%%Remove Cyclic extension%%
%%for AWGN%%
for k=1:30
    for i=1:100
        rec_nocyclic_awgn_QPSK_nocode(((i-1)*64)+1:(i*64),k)=awgn_X_qpsk_nocode(((i-
1)*80)+17:(i*80),k) ;
        rec_nocyclic_awgn_QPSK_rep(((i-1)*64)+1:(i*64),k)=awgn_X_qpsk_rep(((i-
1)*80)+17:(i*80),k) ;
        rec_nocyclic_awgn_16QAM_nocode(((i-1)*64)+1:(i*64),k)=awgn_X_qam_nocode(((i-
1)*80)+17:(i*80),k) ;
        rec_nocyclic_awgn_16QAM_rep(((i-1)*64)+1:(i*64),k)=awgn_X_qam_rep(((i-
1)*80)+17:(i*80),k) ;
    end
end
%%for Frequency selective fading%%
for k=1:30
    for i=1:100
        rec_nocyclic_freqs_QPSK_nocode(((i-
1)*64)+1:(i*64),k)=rec_signal_freqs_qpsk_nocode(((i-1)*80)+17:(i*80),k) ;
        rec_nocyclic_freqs_QPSK_rep(((i-
1)*64)+1:(i*64),k)=rec_signal_freqs_qpsk_rep(((i-1)*80)+17:(i*80),k) ;
        rec_nocyclic_freqs_16QAM_nocode(((i-
1)*64)+1:(i*64),k)=rec_signal_freqs_qam_nocode(((i-1)*80)+17:(i*80),k) ;
        rec_nocyclic_freqs_16QAM_rep(((i-
1)*64)+1:(i*64),k)=rec_signal_freqs_qam_rep(((i-1)*80)+17:(i*80),k) ;
    end
end
```

```
%%FFT%%
%%for AWGN%%
for k=1:30
    for i=1:100
        rec_fft_awgn_QPSK_nocode(((i-1)*64)+1:(i*64),k)=
fft(rec_nocyclic_awgn_QPSK_nocode(((i-1)*64)+1:(i*64),k)) ;
        rec_fft_awgn_QPSK_rep(((i-
1)*64)+1:(i*64),k)=fft(rec_nocyclic_awgn_QPSK_rep(((i-1)*64)+1:(i*64),k)) ;
        rec_fft_awgn_16QAM_nocode(((i-
1)*64)+1:(i*64),k)=fft(rec_nocyclic_awgn_16QAM_nocode(((i-1)*64)+1:(i*64),k)) ;
        rec_fft_awgn_16QAM_rep(((i-
1)*64)+1:(i*64),k)=fft(rec_nocyclic_awgn_16QAM_rep(((i-1)*64)+1:(i*64),k)) ;
    end
end
%%for Frequency selective fading%%
for k=1:30
    for i=1:100
        rec_fft_freqs_QPSK_nocode(((i-1)*64)+1:(i*64),k)=
fft(rec_nocyclic_freqs_QPSK_nocode(((i-1)*64)+1:(i*64),k)) ;
        rec_fft_freqs_QPSK_rep(((i-
1)*64)+1:(i*64),k)=fft(rec_nocyclic_freqs_QPSK_rep(((i-1)*64)+1:(i*64),k)) ;
        rec_fft_freqs_16QAM_nocode(((i-
1)*64)+1:(i*64),k)=fft(rec_nocyclic_freqs_16QAM_nocode(((i-1)*64)+1:(i*64),k)) ;
        rec_fft_freqs_16QAM_rep(((i-
1)*64)+1:(i*64),k)=fft(rec_nocyclic_freqs_16QAM_rep(((i-1)*64)+1:(i*64),k)) ;
    end
end
%%Demapper%%
%%%%%%%%% Qpsk %%%%%%%%
%%no coding AWGN%%
for i=1:6400
    for k=1:30
if real(rec_fft_awgn_QPSK_nocode(i,k))>0 && imag(rec_fft_awgn_QPSK_nocode(i,k))>0
    rec_bits_awgn_nocoding_qpsk(2*i-1,k)=1 ; rec_bits_awgn_nocoding_qpsk(2*i,k)=1 ;
elseif real(rec_fft_awgn_QPSK_nocode(i,k))<0 && imag(rec_fft_awgn_QPSK_nocode(i,k))>0
    rec_bits_awgn_nocoding_qpsk(2*i-1,k)=0 ; rec_bits_awgn_nocoding_qpsk(2*i,k)=1 ;
elseif real(rec_fft_awgn_QPSK_nocode(i,k))<0 && imag(rec_fft_awgn_QPSK_nocode(i,k))<0
    rec_bits_awgn_nocoding_qpsk(2*i-1,k)=0 ; rec_bits_awgn_nocoding_qpsk(2*i,k)=0 ;
elseif real(rec_fft_awgn_QPSK_nocode(i,k))>0 && imag(rec_fft_awgn_QPSK_nocode(i,k))<0
    rec_bits_awgn_nocoding_qpsk(2*i-1,k)=1 ; rec_bits_awgn_nocoding_qpsk(2*i,k)=0 ;
end
   end
end
%%no coding Freqs%%
for i=1:6400
    for k=1:30
if real(rec_fft_freqs_QPSK_nocode(i,k))>0 && imag(rec_fft_freqs_QPSK_nocode(i,k))>0
    rec_bits_freqs_nocoding_qpsk(2*i-1,k)=1 ; rec_bits_freqs_nocoding_qpsk(2*i,k)=1 ;
elseif real(rec_fft_freqs_QPSK_nocode(i,k))<0 && imag(rec_fft_freqs_QPSK_nocode(i,k))>0
    rec_bits_freqs_nocoding_qpsk(2*i-1,k)=0 ; rec_bits_freqs_nocoding_qpsk(2*i,k)=1 ;
elseif real(rec_fft_freqs_QPSK_nocode(i,k))<0 && imag(rec_fft_freqs_QPSK_nocode(i,k))<0
    rec_bits_freqs_nocoding_qpsk(2*i-1,k)=0 ; rec_bits_freqs_nocoding_qpsk(2*i,k)=0 ;
elseif real(rec_fft_freqs_QPSK_nocode(i,k))>0 && imag(rec_fft_freqs_QPSK_nocode(i,k))<0
    rec_bits_freqs_nocoding_qpsk(2*i-1,k)=1 ; rec_bits_freqs_nocoding_qpsk(2*i,k)=0 ;
end
   end
end
```

```
%%repetition coding AWGN%%
for i=1:6400
    for k=1:30
if real(rec_fft_awgn_QPSK_rep(i,k))>0 && imag(rec_fft_awgn_QPSK_rep(i,k))>0
    rec_bits_awgn_rep_qpsk(2*i-1,k)=1 ; rec_bits_awgn_rep_qpsk(2*i,k)=1 ;
elseif real(rec_fft_awgn_QPSK_rep(i,k))<0 && imag(rec_fft_awgn_QPSK_rep(i,k))>0
    rec_bits_awgn_rep_qpsk(2*i-1,k)=0 ; rec_bits_awgn_rep_qpsk(2*i,k)=1 ;
elseif real(rec_fft_awgn_QPSK_rep(i,k))<0 && imag(rec_fft_awgn_QPSK_rep(i,k))<0
    rec_bits_awgn_rep_qpsk(2*i-1,k)=0 ; rec_bits_awgn_rep_qpsk(2*i,k)=0 ;
elseif real(rec_fft_awgn_QPSK_rep(i,k))>0 && imag(rec_fft_awgn_QPSK_rep(i,k))<0
    rec_bits_awgn_rep_qpsk(2*i-1,k)=1 ; rec_bits_awgn_rep_qpsk(2*i,k)=0 ;
end
   end
end
%%repetition coding Freqs%%
for i=1:6400
    for k=1:30
if real(rec_fft_freqs_QPSK_rep(i,k))>0 && imag(rec_fft_freqs_QPSK_rep(i,k))>0
    rec_bits_freqs_rep_qpsk(2*i-1,k)=1 ; rec_bits_freqs_rep_qpsk(2*i,k)=1 ;
elseif real(rec_fft_freqs_QPSK_rep(i,k))<0 && imag(rec_fft_freqs_QPSK_rep(i,k))>0
    rec_bits_freqs_rep_qpsk(2*i-1,k)=0 ; rec_bits_freqs_rep_qpsk(2*i,k)=1 ;
elseif real(rec_fft_freqs_QPSK_rep(i,k))<0 && imag(rec_fft_freqs_QPSK_rep(i,k))<0
    rec_bits_freqs_rep_qpsk(2*i-1,k)=0 ; rec_bits_freqs_rep_qpsk(2*i,k)=0 ;
elseif real(rec_fft_freqs_QPSK_rep(i,k))>0 && imag(rec_fft_freqs_QPSK_rep(i,k))<0
    rec_bits_freqs_rep_qpsk(2*i-1,k)=1 ; rec_bits_freqs_rep_qpsk(2*i,k)=0 ;
end
   end
end
%%16QAM%%
%%no coding AWGN%%
for i=1:6400
    for k=1:30
if (real(rec_fft_awgn_16QAM_nocode(i,k))>0 &&
real(rec_fft_awgn_16QAM_nocode(i,k)<2*sqrt(Eb_nocode(1,k)/2.5)) && ...
 imag(rec_fft_awgn_16QAM_nocode(i,k))>0 &&
imag(rec_fft_awgn_16QAM_nocode(i,k))<2*sqrt(Eb_nocode(1,k)/2.5))
rec_bits_awgn_nocode_qam(4*i-3,k)=1;rec_bits_awgn_nocode_qam(4*i-
2,k)=1;rec_bits_awgn_nocode_qam(4*i-1,k)=1;...
    rec_bits_awgn_nocode_qam(4*i,k)=1 ;
elseif (real(rec_fft_awgn_16QAM_nocode(i,k))>0 &&
real(rec_fft_awgn_16QAM_nocode(i,k))<2*sqrt(Eb_nocode(1,k)/2.5) && ...
        imag(rec_fft_awgn_16QAM_nocode(i,k))>2*sqrt(Eb_nocode(1,k)/2.5))
rec_bits_awgn_nocode_qam(4*i-3,k)=1;rec_bits_awgn_nocode_qam(4*i-
2,k)=1;rec_bits_awgn_nocode_qam(4*i-1,k)=1;...
    rec_bits_awgn_nocode_qam(4*i,k)=0 ;
elseif (real(rec_fft_awgn_16QAM_nocode(i,k))>0 &&
real(rec_fft_awgn_16QAM_nocode(i,k))<2*sqrt(Eb_nocode(1,k)/2.5) && ...
    imag(rec_fft_awgn_16QAM_nocode(i,k))<0 && imag(rec_fft_awgn_16QAM_nocode(i,k))>-
2*sqrt(Eb_nocode(1,k)/2.5))
rec_bits_awgn_nocode_qam(4*i-3,k)=1;rec_bits_awgn_nocode_qam(4*i-
2,k)=1;rec_bits_awgn_nocode_qam(4*i-1,k)=0;...
    rec_bits_awgn_nocode_qam(4*i,k)=1 ;
elseif (real(rec_fft_awgn_16QAM_nocode(i,k))>0 &&
real(rec_fft_awgn_16QAM_nocode(i,k))<2*sqrt(Eb_nocode(1,k)/2.5) && ...
    imag(rec_fft_awgn_16QAM_nocode(i,k))<-2*sqrt(Eb_nocode(1,k)/2.5))
rec_bits_awgn_nocode_qam(4*i-3,k)=1;rec_bits_awgn_nocode_qam(4*i-
2,k)=1;rec_bits_awgn_nocode_qam(4*i-1,k)=0;...
    rec_bits_awgn_nocode_qam(4*i,k)=0 ;
```

```
elseif (real(rec_fft_awgn_16QAM_nocode(i,k))>2*sqrt(Eb_nocode(1,k)/2.5)
&&imag(rec_fft_awgn_16QAM_nocode(i,k))>0 && ...
    imag(rec_fft_awgn_16QAM_nocode(i,k))<2*sqrt(Eb_nocode(1,k)/2.5))
rec_bits_awgn_nocode_qam(4*i-3,k)=1;rec_bits_awgn_nocode_qam(4*i-
2,k)=0;rec_bits_awgn_nocode_qam(4*i-1,k)=1;...
    rec_bits_awgn_nocode_qam(4*i,k)=1 ;
elseif (real(rec_fft_awgn_16QAM_nocode(i,k))>2*sqrt(Eb_nocode(1,k)/2.5) && ...
    imag(rec_fft_awgn_16QAM_nocode(i,k))>2*sqrt(Eb_nocode(1,k)/2.5))
rec_bits_awgn_nocode_qam(4*i-3,k)=1;rec_bits_awgn_nocode_qam(4*i-
2,k)=0;rec_bits_awgn_nocode_qam(4*i-1,k)=1;...
    rec_bits_awgn_nocode_qam(4*i,k)=0 ;
elseif (real(rec_fft_awgn_16QAM_nocode(i,k))>2*sqrt(Eb_nocode(1,k)/2.5)
&&imag(rec_fft_awgn_16QAM_nocode(i,k))<0 && ...
    imag(rec_fft_awgn_16QAM_nocode(i,k))>-2*sqrt(Eb_nocode(1,k)/2.5))
rec_bits_awgn_nocode_qam(4*i-3,k)=1;rec_bits_awgn_nocode_qam(4*i-
2,k)=0;rec_bits_awgn_nocode_qam(4*i-1,k)=0;...
    rec_bits_awgn_nocode_qam(4*i,k)=1 ;
elseif (real(rec_fft_awgn_16QAM_nocode(i,k))>2*sqrt(Eb_nocode(1,k)/2.5) && ...
    imag(rec_fft_awgn_16QAM_nocode(i,k))<-2*sqrt(Eb_nocode(1,k)/2.5))
rec_bits_awgn_nocode_qam(4*i-3,k)=1;rec_bits_awgn_nocode_qam(4*i-
2,k)=0;rec_bits_awgn_nocode_qam(4*i-1,k)=0;...
    rec_bits_awgn_nocode_qam(4*i,k)=0 ;
elseif (real(rec_fft_awgn_16QAM_nocode(i,k))<0 &&
real(rec_fft_awgn_16QAM_nocode(i,k))>-2*sqrt(Eb_nocode(1,k)/2.5) && ...
    imag(rec_fft_awgn_16QAM_nocode(i,k))>0&&
imag(rec_fft_awgn_16QAM_nocode(i,k))<2*sqrt(Eb_nocode(1,k)/2.5))
rec_bits_awgn_nocode_qam(4*i-3,k)=0;rec_bits_awgn_nocode_qam(4*i-
2,k)=1;rec_bits_awgn_nocode_qam(4*i-1,k)=1;...
    rec_bits_awgn_nocode_qam(4*i,k)=1 ;
elseif (real(rec_fft_awgn_16QAM_nocode(i,k))<0 &&
real(rec_fft_awgn_16QAM_nocode(i,k))>-2*sqrt(Eb_nocode(1,k)/2.5) && ...
    imag(rec_fft_awgn_16QAM_nocode(i,k))>2*sqrt(Eb_nocode(1,k)/2.5))
rec_bits_awgn_nocode_qam(4*i-3,k)=0;rec_bits_awgn_nocode_qam(4*i-
2,k)=1;rec_bits_awgn_nocode_qam(4*i-1,k)=1;...
    rec_bits_awgn_nocode_qam(4*i,k)=0 ;
elseif (real(rec_fft_awgn_16QAM_nocode(i,k))<0 &&
real(rec_fft_awgn_16QAM_nocode(i,k))>-2*sqrt(Eb_nocode(1,k)/2.5) && ...
    imag(rec_fft_awgn_16QAM_nocode(i,k))<0&& imag(rec_fft_awgn_16QAM_nocode(i,k))>-
2*sqrt(Eb_nocode(1,k)/2.5))
rec_bits_awgn_nocode_qam(4*i-3,k)=0;rec_bits_awgn_nocode_qam(4*i-
2,k)=1;rec_bits_awgn_nocode_qam(4*i-1,k)=0;...
    rec_bits_awgn_nocode_qam(4*i,k)=1 ;
elseif (real(rec_fft_awgn_16QAM_nocode(i,k))<0 &&
real(rec_fft_awgn_16QAM_nocode(i,k))>-2*sqrt(Eb_nocode(1,k)/2.5) && ...
    imag(rec_fft_awgn_16QAM_nocode(i,k))<-2*sqrt(Eb_nocode(1,k)/2.5))
rec_bits_awgn_nocode_qam(4*i-3,k)=0;rec_bits_awgn_nocode_qam(4*i-
2,k)=1;rec_bits_awgn_nocode_qam(4*i-1,k)=0;...
    rec_bits_awgn_nocode_qam(4*i,k)=0 ;
elseif (real(rec_fft_awgn_16QAM_nocode(i,k))<-2*sqrt(Eb_nocode(1,k)/2.5) &&
imag(rec_fft_awgn_16QAM_nocode(i,k))>0 && ...
    imag(rec_fft_awgn_16QAM_nocode(i,k))<2*sqrt(Eb_nocode(1,k)/2.5))
rec_bits_awgn_nocode_qam(4*i-3,k)=0;rec_bits_awgn_nocode_qam(4*i-
2,k)=0;rec_bits_awgn_nocode_qam(4*i-1,k)=1;...
    rec_bits_awgn_nocode_qam(4*i,k)=1 ;
elseif (real(rec_fft_awgn_16QAM_nocode(i,k))<-2*sqrt(Eb_nocode(1,k)/2.5) && ...
    imag(rec_fft_awgn_16QAM_nocode(i,k))>2*sqrt(Eb_nocode(1,k)/2.5))
rec_bits_awgn_nocode_qam(4*i-3,k)=0;rec_bits_awgn_nocode_qam(4*i-
2,k)=0;rec_bits_awgn_nocode_qam(4*i-1,k)=1;...
    rec_bits_awgn_nocode_qam(4*i,k)=0 ;
```

```
elseif (real(rec_fft_awgn_16QAM_nocode(i,k))<-2*sqrt(Eb_nocode(1,k)/2.5) &&
imag(rec_fft_awgn_16QAM_nocode(i,k))<0 && ...
    imag(rec_fft_awgn_16QAM_nocode(i,k))>-2*sqrt(Eb_nocode(1,k)/2.5))
rec_bits_awgn_nocode_qam(4*i-3,k)=0;rec_bits_awgn_nocode_qam(4*i-
2,k)=0;rec_bits_awgn_nocode_qam(4*i-1,k)=0;...
    rec_bits_awgn_nocode_qam(4*i,k)=1 ;
elseif (real(rec_fft_awgn_16QAM_nocode(i,k))<-2*sqrt(Eb_nocode(1,k)/2.5)&& ...
    imag(rec_fft_awgn_16QAM_nocode(i,k))<-2*sqrt(Eb_nocode(1,k)/2.5))
rec_bits_awgn_nocode_qam(4*i-3,k)=0;rec_bits_awgn_nocode_qam(4*i-
2,k)=0;rec_bits_awgn_nocode_qam(4*i-1,k)=0;...
    rec_bits_awgn_nocode_qam(4*i,k)=0 ;
end
    end
end
%%no coding Freqs%%
for i=1:6400
    for k=1:30
if (real(rec_fft_freqs_16QAM_nocode(i,k))>0 &&
real(rec_fft_freqs_16QAM_nocode(i,k)<2*sqrt(Eb_nocode(1,k)/2.5)) && ...
 imag(rec_fft_freqs_16QAM_nocode(i,k))>0 &&
imag(rec_fft_freqs_16QAM_nocode(i,k))<2*sqrt(Eb_nocode(1,k)/2.5))
rec_bits_freqs_nocode_qam(4*i-3,k)=1;rec_bits_freqs_nocode_qam(4*i-
2,k)=1;rec_bits_freqs_nocode_qam(4*i-1,k)=1;...
    rec_bits_freqs_nocode_qam(4*i,k)=1 ;
elseif (real(rec_fft_freqs_16QAM_nocode(i,k))>0 &&
real(rec_fft_freqs_16QAM_nocode(i,k))<2*sqrt(Eb_nocode(1,k)/2.5) && ...
        imag(rec_fft_freqs_16QAM_nocode(i,k))>2*sqrt(Eb_nocode(1,k)/2.5))
rec_bits_freqs_nocode_qam(4*i-3,k)=1;rec_bits_freqs_nocode_qam(4*i-
2,k)=1;rec_bits_freqs_nocode_qam(4*i-1,k)=1;...
    rec_bits_freqs_nocode_qam(4*i,k)=0 ;
elseif (real(rec_fft_freqs_16QAM_nocode(i,k))>0 &&
real(rec_fft_freqs_16QAM_nocode(i,k))<2*sqrt(Eb_nocode(1,k)/2.5) && ...
    imag(rec_fft_freqs_16QAM_nocode(i,k))<0 && imag(rec_fft_freqs_16QAM_nocode(i,k))>-
2*sqrt(Eb_nocode(1,k)/2.5))
rec_bits_freqs_nocode_qam(4*i-3,k)=1;rec_bits_freqs_nocode_qam(4*i-
2,k)=1;rec_bits_freqs_nocode_qam(4*i-1,k)=0;...
    rec_bits_freqs_nocode_qam(4*i,k)=1 ;
elseif (real(rec_fft_freqs_16QAM_nocode(i,k))>0 &&
real(rec_fft_freqs_16QAM_nocode(i,k))<2*sqrt(Eb_nocode(1,k)/2.5) && ...
    imag(rec_fft_freqs_16QAM_nocode(i,k))<-2*sqrt(Eb_nocode(1,k)/2.5))
rec_bits_freqs_nocode_qam(4*i-3,k)=1;rec_bits_freqs_nocode_qam(4*i-
2,k)=1;rec_bits_freqs_nocode_qam(4*i-1,k)=0;...
    rec_bits_freqs_nocode_qam(4*i,k)=0 ;
elseif (real(rec_fft_freqs_16QAM_nocode(i,k))>2*sqrt(Eb_nocode(1,k)/2.5)
&&imag(rec_fft_freqs_16QAM_nocode(i,k))>0 && ...
    imag(rec_fft_freqs_16QAM_nocode(i,k))<2*sqrt(Eb_nocode(1,k)/2.5))
rec_bits_freqs_nocode_qam(4*i-3,k)=1;rec_bits_freqs_nocode_qam(4*i-
2,k)=0;rec_bits_freqs_nocode_qam(4*i-1,k)=1;...
    rec_bits_freqs_nocode_qam(4*i,k)=1 ;
elseif (real(rec_fft_freqs_16QAM_nocode(i,k))>2*sqrt(Eb_nocode(1,k)/2.5) && ...
    imag(rec_fft_freqs_16QAM_nocode(i,k))>2*sqrt(Eb_nocode(1,k)/2.5))
rec_bits_freqs_nocode_qam(4*i-3,k)=1;rec_bits_freqs_nocode_qam(4*i-
2,k)=0;rec_bits_freqs_nocode_qam(4*i-1,k)=1;...
    rec_bits_freqs_nocode_qam(4*i,k)=0 ;
elseif (real(rec_fft_freqs_16QAM_nocode(i,k))>2*sqrt(Eb_nocode(1,k)/2.5)
&&imag(rec_fft_freqs_16QAM_nocode(i,k))<0 && ...
    imag(rec_fft_freqs_16QAM_nocode(i,k))>-2*sqrt(Eb_nocode(1,k)/2.5))
rec_bits_freqs_nocode_qam(4*i-3,k)=1;rec_bits_freqs_nocode_qam(4*i-
2,k)=0;rec_bits_freqs_nocode_qam(4*i-1,k)=0;...
    rec_bits_freqs_nocode_qam(4*i,k)=1 ;
```

```
elseif (real(rec_fft_freqs_16QAM_nocode(i,k))>2*sqrt(Eb_nocode(1,k)/2.5) && ...
    imag(rec_fft_freqs_16QAM_nocode(i,k))<-2*sqrt(Eb_nocode(1,k)/2.5))
rec_bits_freqs_nocode_qam(4*i-3,k)=1;rec_bits_freqs_nocode_qam(4*i-
2,k)=0;rec_bits_freqs_nocode_qam(4*i-1,k)=0;...
    rec_bits_freqs_nocode_qam(4*i,k)=0 ;
elseif (real(rec_fft_freqs_16QAM_nocode(i,k))<0 &&
real(rec_fft_freqs_16QAM_nocode(i,k))>-2*sqrt(Eb_nocode(1,k)/2.5) && ...
    imag(rec_fft_freqs_16QAM_nocode(i,k))>0&&
imag(rec_fft_freqs_16QAM_nocode(i,k))<2*sqrt(Eb_nocode(1,k)/2.5))
rec_bits_freqs_nocode_qam(4*i-3,k)=0;rec_bits_freqs_nocode_qam(4*i-
2,k)=1;rec_bits_freqs_nocode_qam(4*i-1,k)=1;...
    rec_bits_freqs_nocode_qam(4*i,k)=1 ;
elseif (real(rec_fft_freqs_16QAM_nocode(i,k))<0 &&
real(rec_fft_freqs_16QAM_nocode(i,k))>-2*sqrt(Eb_nocode(1,k)/2.5) && ...
    imag(rec_fft_freqs_16QAM_nocode(i,k))>2*sqrt(Eb_nocode(1,k)/2.5))
rec_bits_freqs_nocode_qam(4*i-3,k)=0;rec_bits_freqs_nocode_qam(4*i-
2,k)=1;rec_bits_freqs_nocode_qam(4*i-1,k)=1;...
    rec_bits_freqs_nocode_qam(4*i,k)=0 ;
elseif (real(rec_fft_freqs_16QAM_nocode(i,k))<0 &&
real(rec_fft_freqs_16QAM_nocode(i,k))>-2*sqrt(Eb_nocode(1,k)/2.5) && ...
    imag(rec_fft_freqs_16QAM_nocode(i,k))<0&& imag(rec_fft_freqs_16QAM_nocode(i,k))>-
2*sqrt(Eb_nocode(1,k)/2.5))
rec_bits_freqs_nocode_qam(4*i-3,k)=0;rec_bits_freqs_nocode_qam(4*i-
2,k)=1;rec_bits_freqs_nocode_qam(4*i-1,k)=0;...
    rec_bits_freqs_nocode_qam(4*i,k)=1 ;
elseif (real(rec_fft_freqs_16QAM_nocode(i,k))<0 &&
real(rec_fft_freqs_16QAM_nocode(i,k))>-2*sqrt(Eb_nocode(1,k)/2.5) && ...
    imag(rec_fft_freqs_16QAM_nocode(i,k))<-2*sqrt(Eb_nocode(1,k)/2.5))
rec_bits_freqs_nocode_qam(4*i-3,k)=0;rec_bits_freqs_nocode_qam(4*i-
2,k)=1;rec_bits_freqs_nocode_qam(4*i-1,k)=0;...
    rec_bits_freqs_nocode_qam(4*i,k)=0 ;
elseif (real(rec_fft_freqs_16QAM_nocode(i,k))<-2*sqrt(Eb_nocode(1,k)/2.5) &&
imag(rec_fft_freqs_16QAM_nocode(i,k))>0 && ...
    imag(rec_fft_freqs_16QAM_nocode(i,k))<2*sqrt(Eb_nocode(1,k)/2.5))
rec_bits_freqs_nocode_qam(4*i-3,k)=0;rec_bits_freqs_nocode_qam(4*i-
2,k)=0;rec_bits_freqs_nocode_qam(4*i-1,k)=1;...
    rec_bits_freqs_nocode_qam(4*i,k)=1 ;
elseif (real(rec_fft_freqs_16QAM_nocode(i,k))<-2*sqrt(Eb_nocode(1,k)/2.5) && ...
    imag(rec_fft_freqs_16QAM_nocode(i,k))>2*sqrt(Eb_nocode(1,k)/2.5))
rec_bits_freqs_nocode_qam(4*i-3,k)=0;rec_bits_freqs_nocode_qam(4*i-
2,k)=0;rec_bits_freqs_nocode_qam(4*i-1,k)=1;...
    rec_bits_freqs_nocode_qam(4*i,k)=0 ;
elseif (real(rec_fft_freqs_16QAM_nocode(i,k))<-2*sqrt(Eb_nocode(1,k)/2.5) &&
imag(rec_fft_freqs_16QAM_nocode(i,k))<0 && ...
    imag(rec_fft_freqs_16QAM_nocode(i,k))>-2*sqrt(Eb_nocode(1,k)/2.5))
rec_bits_freqs_nocode_qam(4*i-3,k)=0;rec_bits_freqs_nocode_qam(4*i-
2,k)=0;rec_bits_freqs_nocode_qam(4*i-1,k)=0;...
    rec_bits_freqs_nocode_qam(4*i,k)=1 ;
elseif (real(rec_fft_freqs_16QAM_nocode(i,k))<-2*sqrt(Eb_nocode(1,k)/2.5)&& ...
    imag(rec_fft_freqs_16QAM_nocode(i,k))<-2*sqrt(Eb_nocode(1,k)/2.5))
rec_bits_freqs_nocode_qam(4*i-3,k)=0;rec_bits_freqs_nocode_qam(4*i-
2,k)=0;rec_bits_freqs_nocode_qam(4*i-1,k)=0;...
    rec_bits_freqs_nocode_qam(4*i,k)=0 ;
end
    end
end
 %%repetition coding AWGN%%
for i=1:6400
    for k=1:30
```

```
if (real(rec_fft_awgn_16QAM_rep(i,k))>0 &&
real(rec_fft_awgn_16QAM_rep(i,k)<2*sqrt(Eb_rep(1,k)/2.5)) && ...
 imag(rec_fft_awgn_16QAM_rep(i,k))>0 &&
imag(rec_fft_awgn_16QAM_rep(i,k))<2*sqrt(Eb_rep(1,k)/2.5))
rec_bits_awgn_rep_qam(4*i-3,k)=1;rec_bits_awgn_rep_qam(4*i-
2,k)=1;rec_bits_awgn_rep_qam(4*i-1,k)=1;...
    rec_bits_awgn_rep_qam(4*i,k)=1 ;
elseif (real(rec_fft_awgn_16QAM_rep(i,k))>0 &&
real(rec_fft_awgn_16QAM_rep(i,k))<2*sqrt(Eb_rep(1,k)/2.5) && ...
        imag(rec_fft_awgn_16QAM_rep(i,k))>2*sqrt(Eb_rep(1,k)/2.5))
rec_bits_awgn_rep_qam(4*i-3,k)=1;rec_bits_awgn_rep_qam(4*i-
2,k)=1;rec_bits_awgn_rep_qam(4*i-1,k)=1;...
    rec_bits_awgn_rep_qam(4*i,k)=0 ;
elseif (real(rec_fft_awgn_16QAM_rep(i,k))>0 &&
real(rec_fft_awgn_16QAM_rep(i,k))<2*sqrt(Eb_rep(1,k)/2.5) && ...
    imag(rec_fft_awgn_16QAM_rep(i,k))<0 && imag(rec_fft_awgn_16QAM_rep(i,k))>-
2*sqrt(Eb_rep(1,k)/2.5))
rec_bits_awgn_rep_qam(4*i-3,k)=1;rec_bits_awgn_rep_qam(4*i-
2,k)=1;rec_bits_awgn_rep_qam(4*i-1,k)=0;...
    rec_bits_awgn_rep_qam(4*i,k)=1 ;
elseif (real(rec_fft_awgn_16QAM_rep(i,k))>0 &&
real(rec_fft_awgn_16QAM_rep(i,k))<2*sqrt(Eb_rep(1,k)/2.5) && ...
    imag(rec_fft_awgn_16QAM_rep(i,k))<-2*sqrt(Eb_rep(1,k)/2.5))
rec_bits_awgn_rep_qam(4*i-3,k)=1;rec_bits_awgn_rep_qam(4*i-
2,k)=1;rec_bits_awgn_rep_qam(4*i-1,k)=0;...
    rec_bits_awgn_rep_qam(4*i,k)=0 ;
elseif (real(rec_fft_awgn_16QAM_rep(i,k))>2*sqrt(Eb_rep(1,k)/2.5)
&&imag(rec_fft_awgn_16QAM_rep(i,k))>0 && ...
    imag(rec_fft_awgn_16QAM_rep(i,k))<2*sqrt(Eb_rep(1,k)/2.5))
rec_bits_awgn_rep_qam(4*i-3,k)=1;rec_bits_awgn_rep_qam(4*i-
2,k)=0;rec_bits_awgn_rep_qam(4*i-1,k)=1;...
    rec_bits_awgn_rep_qam(4*i,k)=1 ;
elseif (real(rec_fft_awgn_16QAM_rep(i,k))>2*sqrt(Eb_rep(1,k)/2.5) && ...
    imag(rec_fft_awgn_16QAM_rep(i,k))>2*sqrt(Eb_rep(1,k)/2.5))
rec_bits_awgn_rep_qam(4*i-3,k)=1;rec_bits_awgn_rep_qam(4*i-
2,k)=0;rec_bits_awgn_rep_qam(4*i-1,k)=1;...
    rec_bits_awgn_rep_qam(4*i,k)=0 ;
elseif (real(rec_fft_awgn_16QAM_rep(i,k))>2*sqrt(Eb_rep(1,k)/2.5)
&&imag(rec_fft_awgn_16QAM_rep(i,k))<0 && ...
    imag(rec_fft_awgn_16QAM_rep(i,k))>-2*sqrt(Eb_rep(1,k)/2.5))
rec_bits_awgn_rep_qam(4*i-3,k)=1;rec_bits_awgn_rep_qam(4*i-
2,k)=0;rec_bits_awgn_rep_qam(4*i-1,k)=0;...
    rec_bits_awgn_rep_qam(4*i,k)=1 ;
elseif (real(rec_fft_awgn_16QAM_rep(i,k))>2*sqrt(Eb_rep(1,k)/2.5) && ...
    imag(rec_fft_awgn_16QAM_rep(i,k))<-2*sqrt(Eb_rep(1,k)/2.5))
rec_bits_awgn_rep_qam(4*i-3,k)=1;rec_bits_awgn_rep_qam(4*i-
2,k)=0;rec_bits_awgn_rep_qam(4*i-1,k)=0;...
    rec_bits_awgn_rep_qam(4*i,k)=0 ;
elseif (real(rec_fft_awgn_16QAM_rep(i,k))<0 && real(rec_fft_awgn_16QAM_rep(i,k))>-
2*sqrt(Eb_rep(1,k)/2.5) && ...
    imag(rec_fft_awgn_16QAM_rep(i,k))>0&&
imag(rec_fft_awgn_16QAM_rep(i,k))<2*sqrt(Eb_rep(1,k)/2.5))
rec_bits_awgn_rep_qam(4*i-3,k)=0;rec_bits_awgn_rep_qam(4*i-
2,k)=1;rec_bits_awgn_rep_qam(4*i-1,k)=1;...
    rec_bits_awgn_rep_qam(4*i,k)=1 ;
```

```
elseif (real(rec_fft_awgn_16QAM_rep(i,k))<0 && real(rec_fft_awgn_16QAM_rep(i,k))>-
2*sqrt(Eb_rep(1,k)/2.5) && ...
    imag(rec_fft_awgn_16QAM_rep(i,k))>2*sqrt(Eb_rep(1,k)/2.5))
rec_bits_awgn_rep_qam(4*i-3,k)=0;rec_bits_awgn_rep_qam(4*i-
2,k)=1;rec_bits_awgn_rep_qam(4*i-1,k)=1;...
    rec_bits_awgn_rep_qam(4*i,k)=0 ;
elseif (real(rec_fft_awgn_16QAM_rep(i,k))<0 && real(rec_fft_awgn_16QAM_rep(i,k))>-
2*sqrt(Eb_rep(1,k)/2.5) && ...
    imag(rec_fft_awgn_16QAM_rep(i,k))<0&& imag(rec_fft_awgn_16QAM_rep(i,k))>-
2*sqrt(Eb_rep(1,k)/2.5))
rec_bits_awgn_rep_qam(4*i-3,k)=0;rec_bits_awgn_rep_qam(4*i-
2,k)=1;rec_bits_awgn_rep_qam(4*i-1,k)=0;...
    rec_bits_awgn_rep_qam(4*i,k)=1 ;
elseif (real(rec_fft_awgn_16QAM_rep(i,k))<0 && real(rec_fft_awgn_16QAM_rep(i,k))>-
2*sqrt(Eb_rep(1,k)/2.5) && ...
    imag(rec_fft_awgn_16QAM_rep(i,k))<-2*sqrt(Eb_rep(1,k)/2.5))
rec_bits_awgn_rep_qam(4*i-3,k)=0;rec_bits_awgn_rep_qam(4*i-
2,k)=1;rec_bits_awgn_rep_qam(4*i-1,k)=0;...
    rec_bits_awgn_rep_qam(4*i,k)=0 ;
elseif (real(rec_fft_awgn_16QAM_rep(i,k))<-2*sqrt(Eb_rep(1,k)/2.5) &&
imag(rec_fft_awgn_16QAM_rep(i,k))>0 && ...
    imag(rec_fft_awgn_16QAM_rep(i,k))<2*sqrt(Eb_rep(1,k)/2.5))
rec_bits_awgn_rep_qam(4*i-3,k)=0;rec_bits_awgn_rep_qam(4*i-
2,k)=0;rec_bits_awgn_rep_qam(4*i-1,k)=1;...
    rec_bits_awgn_rep_qam(4*i,k)=1 ;
elseif (real(rec_fft_awgn_16QAM_rep(i,k))<-2*sqrt(Eb_rep(1,k)/2.5) && ...
    imag(rec_fft_awgn_16QAM_rep(i,k))>2*sqrt(Eb_rep(1,k)/2.5))
rec_bits_awgn_rep_qam(4*i-3,k)=0;rec_bits_awgn_rep_qam(4*i-
2,k)=0;rec_bits_awgn_rep_qam(4*i-1,k)=1;...
    rec_bits_awgn_rep_qam(4*i,k)=0 ;
elseif (real(rec_fft_awgn_16QAM_rep(i,k))<-2*sqrt(Eb_rep(1,k)/2.5) &&
imag(rec_fft_awgn_16QAM_rep(i,k))<0 && ...
    imag(rec_fft_awgn_16QAM_rep(i,k))>-2*sqrt(Eb_rep(1,k)/2.5))
rec_bits_awgn_rep_qam(4*i-3,k)=0;rec_bits_awgn_rep_qam(4*i-
2,k)=0;rec_bits_awgn_rep_qam(4*i-1,k)=0;...
    rec_bits_awgn_rep_qam(4*i,k)=1 ;
elseif (real(rec_fft_awgn_16QAM_rep(i,k))<-2*sqrt(Eb_rep(1,k)/2.5)&& ...
    imag(rec_fft_awgn_16QAM_rep(i,k))<-2*sqrt(Eb_rep(1,k)/2.5))
rec_bits_awgn_rep_qam(4*i-3,k)=0;rec_bits_awgn_rep_qam(4*i-
2,k)=0;rec_bits_awgn_rep_qam(4*i-1,k)=0;...
    rec_bits_awgn_rep_qam(4*i,k)=0 ;
end
    end
end
%%repetition coding Freqs%%
for i=1:6400
    for k=1:30
if (real(rec_fft_freqs_16QAM_rep(i,k))>0 &&
real(rec_fft_freqs_16QAM_rep(i,k)<2*sqrt(Eb_rep(1,k)/2.5)) && ...
 imag(rec_fft_freqs_16QAM_rep(i,k))>0 &&
imag(rec_fft_freqs_16QAM_rep(i,k))<2*sqrt(Eb_rep(1,k)/2.5))
rec_bits_freqs_rep_qam(4*i-3,k)=1;rec_bits_freqs_rep_qam(4*i-
2,k)=1;rec_bits_freqs_rep_qam(4*i-1,k)=1;...
    rec_bits_freqs_rep_qam(4*i,k)=1 ;
elseif (real(rec_fft_freqs_16QAM_rep(i,k))>0 &&
real(rec_fft_freqs_16QAM_rep(i,k))<2*sqrt(Eb_rep(1,k)/2.5) && ...
        imag(rec_fft_freqs_16QAM_rep(i,k))>2*sqrt(Eb_rep(1,k)/2.5))
rec_bits_freqs_rep_qam(4*i-3,k)=1;rec_bits_freqs_rep_qam(4*i-
2,k)=1;rec_bits_freqs_rep_qam(4*i-1,k)=1;...
    rec_bits_freqs_rep_qam(4*i,k)=0 ;
```

```
elseif (real(rec_fft_freqs_16QAM_rep(i,k))>0 &&
real(rec_fft_freqs_16QAM_rep(i,k))<2*sqrt(Eb_rep(1,k)/2.5) && ...
    imag(rec_fft_freqs_16QAM_rep(i,k))<0 && imag(rec_fft_freqs_16QAM_rep(i,k))>-
2*sqrt(Eb_rep(1,k)/2.5))
rec_bits_freqs_rep_qam(4*i-3,k)=1;rec_bits_freqs_rep_qam(4*i-
2,k)=1;rec_bits_freqs_rep_qam(4*i-1,k)=0;...
    rec_bits_freqs_rep_qam(4*i,k)=1 ;
elseif (real(rec_fft_freqs_16QAM_rep(i,k))>0 &&
real(rec_fft_freqs_16QAM_rep(i,k))<2*sqrt(Eb_rep(1,k)/2.5) && ...
    imag(rec_fft_freqs_16QAM_rep(i,k))<-2*sqrt(Eb_rep(1,k)/2.5))
rec_bits_freqs_rep_qam(4*i-3,k)=1;rec_bits_freqs_rep_qam(4*i-
2,k)=1;rec_bits_freqs_rep_qam(4*i-1,k)=0;...
    rec_bits_freqs_rep_qam(4*i,k)=0 ;
elseif (real(rec_fft_freqs_16QAM_rep(i,k))>2*sqrt(Eb_rep(1,k)/2.5)
&&imag(rec_fft_freqs_16QAM_rep(i,k))>0 && ...
    imag(rec_fft_freqs_16QAM_rep(i,k))<2*sqrt(Eb_rep(1,k)/2.5))
rec_bits_freqs_rep_qam(4*i-3,k)=1;rec_bits_freqs_rep_qam(4*i-
2,k)=0;rec_bits_freqs_rep_qam(4*i-1,k)=1;...
    rec_bits_freqs_rep_qam(4*i,k)=1 ;
elseif (real(rec_fft_freqs_16QAM_rep(i,k))>2*sqrt(Eb_rep(1,k)/2.5) && ...
    imag(rec_fft_freqs_16QAM_rep(i,k))>2*sqrt(Eb_rep(1,k)/2.5))
rec_bits_freqs_rep_qam(4*i-3,k)=1;rec_bits_freqs_rep_qam(4*i-
2,k)=0;rec_bits_freqs_rep_qam(4*i-1,k)=1;...
    rec_bits_freqs_rep_qam(4*i,k)=0 ;
elseif (real(rec_fft_freqs_16QAM_rep(i,k))>2*sqrt(Eb_rep(1,k)/2.5)
&&imag(rec_fft_freqs_16QAM_rep(i,k))<0 && ...
    imag(rec_fft_freqs_16QAM_rep(i,k))>-2*sqrt(Eb_rep(1,k)/2.5))
rec_bits_freqs_rep_qam(4*i-3,k)=1;rec_bits_freqs_rep_qam(4*i-
2,k)=0;rec_bits_freqs_rep_qam(4*i-1,k)=0;...
    rec_bits_freqs_rep_qam(4*i,k)=1 ;
elseif (real(rec_fft_freqs_16QAM_rep(i,k))>2*sqrt(Eb_rep(1,k)/2.5) && ...
    imag(rec_fft_freqs_16QAM_rep(i,k))<-2*sqrt(Eb_rep(1,k)/2.5))
rec_bits_freqs_rep_qam(4*i-3,k)=1;rec_bits_freqs_rep_qam(4*i-
2,k)=0;rec_bits_freqs_rep_qam(4*i-1,k)=0;...
    rec_bits_freqs_rep_qam(4*i,k)=0 ;
elseif (real(rec_fft_freqs_16QAM_rep(i,k))<0 && real(rec_fft_freqs_16QAM_rep(i,k))>-
2*sqrt(Eb_rep(1,k)/2.5) && ...
    imag(rec_fft_freqs_16QAM_rep(i,k))>0&&
imag(rec_fft_freqs_16QAM_rep(i,k))<2*sqrt(Eb_rep(1,k)/2.5))
rec_bits_freqs_rep_qam(4*i-3,k)=0;rec_bits_freqs_rep_qam(4*i-
2,k)=1;rec_bits_freqs_rep_qam(4*i-1,k)=1;...
    rec_bits_freqs_rep_qam(4*i,k)=1 ;
elseif (real(rec_fft_freqs_16QAM_rep(i,k))<0 && real(rec_fft_freqs_16QAM_rep(i,k))>-
2*sqrt(Eb_rep(1,k)/2.5) && ...
    imag(rec_fft_freqs_16QAM_rep(i,k))>2*sqrt(Eb_rep(1,k)/2.5))
rec_bits_freqs_rep_qam(4*i-3,k)=0;rec_bits_freqs_rep_qam(4*i-
2,k)=1;rec_bits_freqs_rep_qam(4*i-1,k)=1;...
    rec_bits_freqs_rep_qam(4*i,k)=0 ;
elseif (real(rec_fft_freqs_16QAM_rep(i,k))<0 && real(rec_fft_freqs_16QAM_rep(i,k))>-
2*sqrt(Eb_rep(1,k)/2.5) && ...
    imag(rec_fft_freqs_16QAM_rep(i,k))<0&& imag(rec_fft_freqs_16QAM_rep(i,k))>-
2*sqrt(Eb_rep(1,k)/2.5))
rec_bits_freqs_rep_qam(4*i-3,k)=0;rec_bits_freqs_rep_qam(4*i-
2,k)=1;rec_bits_freqs_rep_qam(4*i-1,k)=0;...
    rec_bits_freqs_rep_qam(4*i,k)=1 ;
elseif (real(rec_fft_freqs_16QAM_rep(i,k))<0 && real(rec_fft_freqs_16QAM_rep(i,k))>-
2*sqrt(Eb_rep(1,k)/2.5) && ...
    imag(rec_fft_freqs_16QAM_rep(i,k))<-2*sqrt(Eb_rep(1,k)/2.5))
rec_bits_freqs_rep_qam(4*i-3,k)=0;rec_bits_freqs_rep_qam(4*i-
2,k)=1;rec_bits_freqs_rep_qam(4*i-1,k)=0;...
    rec_bits_freqs_rep_qam(4*i,k)=0 ;
```

```
elseif (real(rec_fft_freqs_16QAM_rep(i,k))<-2*sqrt(Eb_rep(1,k)/2.5) &&
imag(rec_fft_freqs_16QAM_rep(i,k))>0 && ...
    imag(rec_fft_freqs_16QAM_rep(i,k))<2*sqrt(Eb_rep(1,k)/2.5))
rec_bits_freqs_rep_qam(4*i-3,k)=0;rec_bits_freqs_rep_qam(4*i-
2,k)=0;rec_bits_freqs_rep_qam(4*i-1,k)=1;...
    rec_bits_freqs_rep_qam(4*i,k)=1 ;
elseif (real(rec_fft_freqs_16QAM_rep(i,k))<-2*sqrt(Eb_rep(1,k)/2.5) && ...
    imag(rec_fft_freqs_16QAM_rep(i,k))>2*sqrt(Eb_rep(1,k)/2.5))
rec_bits_freqs_rep_qam(4*i-3,k)=0;rec_bits_freqs_rep_qam(4*i-
2,k)=0;rec_bits_freqs_rep_qam(4*i-1,k)=1;...
    rec_bits_freqs_rep_qam(4*i,k)=0 ;
elseif (real(rec_fft_freqs_16QAM_rep(i,k))<-2*sqrt(Eb_rep(1,k)/2.5) &&
imag(rec_fft_freqs_16QAM_rep(i,k))<0 && ...
    imag(rec_fft_freqs_16QAM_rep(i,k))>-2*sqrt(Eb_rep(1,k)/2.5))
rec_bits_freqs_rep_qam(4*i-3,k)=0;rec_bits_freqs_rep_qam(4*i-
2,k)=0;rec_bits_freqs_rep_qam(4*i-1,k)=0;...
    rec_bits_freqs_rep_qam(4*i,k)=1 ;
elseif (real(rec_fft_freqs_16QAM_rep(i,k))<-2*sqrt(Eb_rep(1,k)/2.5)&& ...
    imag(rec_fft_freqs_16QAM_rep(i,k))<-2*sqrt(Eb_rep(1,k)/2.5))
rec_bits_freqs_rep_qam(4*i-3,k)=0;rec_bits_freqs_rep_qam(4*i-
2,k)=0;rec_bits_freqs_rep_qam(4*i-1,k)=0;...
    rec_bits_freqs_rep_qam(4*i,k)=0 ;
end
    end
end
%%Deinterleaver%%
for k=1:30
    for i=1:100
%QPSK nocoding AWGN deinterleaver %
        deintleav_QPSK_awgn_nocode=(reshape(rec_bits_awgn_nocoding_qpsk(((i-
1)*128)+1:(i*128),k),[8,16]))';
        out_deintleav_QPSK_awgn_nocode(((i-
1)*128)+1:(i*128),k)=reshape(deintleav_QPSK_awgn_nocode,128,1) ;
%QPSK repetition coding AWGN deinterleaver %
        deintleav_QPSK_awgn_rep =(reshape(rec_bits_awgn_rep_qpsk(((i-
1)*128)+1:(i*128),k),[8,16]))' ;
        out_deintleav_QPSK_awgn_rep(((i-
1)*128)+1:(i*128),k)=reshape(deintleav_QPSK_awgn_rep,128,1) ;
        y_awgn_repQPSK(((i-1)*126)+1:(i*126),k) =out_deintleav_QPSK_awgn_rep(((i-
1)*128)+3:(i*128),k) ;
%QPSK nocoding Freqs deinterleaver %
        deintleav_QPSK_Freqs_nocode=(reshape(rec_bits_freqs_nocoding_qpsk(((i-
1)*128)+1:(i*128),k),[8,16]))';
        out_deintleav_QPSK_Freqs_nocode(((i-
1)*128)+1:(i*128),k)=reshape(deintleav_QPSK_Freqs_nocode,128,1) ;
%QPSK repetition coding Freqs deinterleaver %
        deintleav_QPSK_Freqs_rep =(reshape(rec_bits_freqs_rep_qpsk(((i-
1)*128)+1:(i*128),k),[8,16]))' ;
        out_deintleav_QPSK_Freqs_rep(((i-
1)*128)+1:(i*128),k)=reshape(deintleav_QPSK_Freqs_rep,128,1) ;
        y_Freqs_repQPSK(((i-1)*126)+1:(i*126),k) =out_deintleav_QPSK_Freqs_rep(((i-
1)*128)+3:(i*128),k) ;
%16QAM nocoding AWGN deinterleaver %
        deintleav_16QAM_awgn_nocode=(reshape(rec_bits_awgn_nocode_qam(((i-
1)*256)+1:(i*256),k),[16,16]))';
        out_deintleav_16QAM_awgn_nocode(((i-
1)*256)+1:(i*256),k)=reshape(deintleav_16QAM_awgn_nocode,256,1);
%16QAM repetition coding AWGN deinterleaver %
        deintleav_16QAM_awgn_rep=(reshape(rec_bits_awgn_rep_qam(((i-
1)*256)+1:(i*256),k),[16,16]))';
```

```
        out_deintleav_16QAM_awgn_rep(((i-
1)*256)+1:(i*256),k)=reshape(deintleav_16QAM_awgn_rep,256,1);
        y_awgn_rep16QAM(((i-1)*255)+1:(i*255),k) = out_deintleav_16QAM_awgn_rep(((i-
1)*256)+2:(i*256),k) ;
%16QAM nocoding Freqs deinterleaver %
        deintleav_16QAM_Freqs_nocode=(reshape(rec_bits_freqs_nocode_qam(((i-
1)*256)+1:(i*256),k),[16,16]))';
        out_deintleav_16QAM_Freqs_nocode(((i-
1)*256)+1:(i*256),k)=reshape(deintleav_16QAM_Freqs_nocode,256,1);
%16QAM repetition coding Freqs deinterleaver %
        deintleav_16QAM_Freqs_rep=(reshape(rec_bits_freqs_rep_qam(((i-
1)*256)+1:(i*256),k),[16,16]))';
        out_deintleav_16QAM_Freqs_rep(((i-
1)*256)+1:(i*256),k)=reshape(deintleav_16QAM_Freqs_rep,256,1);
        y_Freqs_rep16QAM(((i-1)*255)+1:(i*255),k) = out_deintleav_16QAM_Freqs_rep(((i-
1)*256)+2:(i*256),k) ;
    end
end
y_awgn_nocodeQPSK=out_deintleav_QPSK_awgn_nocode ;
y_Freqs_nocodeQPSK=out_deintleav_QPSK_Freqs_nocode ;
y_awgn_nocode16QAM=out_deintleav_16QAM_awgn_nocode ;
y_Freqs_nocode16QAM=out_deintleav_16QAM_Freqs_nocode ;
for k=1:30
    for i=1:3:12600
        y_awgn_norepQPSK((i+2)/3,k) = mode(y_awgn_repQPSK(i:i+2,k)) ;
        y_Freqs_norepQPSK((i+2)/3,k)=mode(y_Freqs_repQPSK(i:i+2,k)) ;
    end
    for z=1:3:25500
        y_awgn_norep16QAM((z+2)/3,k)=mode(y_awgn_rep16QAM(z:z+2,k)) ;
        y_Freqs_norep16QAM((z+2)/3,k)=mode(y_Freqs_rep16QAM(z:z+2,k)) ;
    end
end
%%%%% BER %%%%%%
errors_awgn_nocodeqpsk=zeros(1,30) ;
errors_Freqs_nocodeqpsk=zeros(1,30) ;
errors_awgn_norepqpsk=zeros(1,30) ;
errors_Freqs_norepqpsk=zeros(1,30) ;
errors_awgn_nocodeqam=zeros(1,30) ;
errors_Freqs_nocodeqam=zeros(1,30) ;
errors_awgn_norepqam=zeros(1,30) ;
errors_Freqs_norepqam=zeros(1,30) ;
%%%%% No Coding Qpsk %%%%%
for i=1:12800
    for k=1:30
    if y_awgn_nocodeQPSK(i,k)~=x_nocodeQPSK(i,1)
       errors_awgn_nocodeqpsk(1,k)=errors_awgn_nocodeqpsk(1,k)+1 ;
    end
    if y_Freqs_nocodeQPSK(i,k)~=x_nocodeQPSK(i,1)
       errors_Freqs_nocodeqpsk(1,k)=errors_Freqs_nocodeqpsk(1,k)+1 ;
    end
    end
end
%%Removed Repetiton Coding QPSK%%
for i=1:4200
    for k=1:30
    if y_awgn_norepQPSK(i,k)~=x_torepQPSK(i,1)
       errors_awgn_norepqpsk(1,k)=errors_awgn_norepqpsk(1,k)+1 ;
    end
    if y_Freqs_norepQPSK(i,k)~=x_torepQPSK(i,1)
       errors_Freqs_norepqpsk(1,k)=errors_Freqs_norepqpsk(1,k)+1 ;
    end
    end
```

```
end
%%%%% No Coding 16QAM %%%%%
for i=1:25600
    for k=1:30
    if y_awgn_nocode16QAM(i,k)~=x_nocode16QAM(i,1)
       errors_awgn_nocodeqam(1,k)=errors_awgn_nocodeqam(1,k)+1 ;
    end
    if y_Freqs_nocode16QAM(i,k)~=x_nocode16QAM(i,1)
       errors_Freqs_nocodeqam(1,k)=errors_Freqs_nocodeqam(1,k)+1 ;
    end
    end
end
%%Removed Repetiton Coding 16QAM%%
for i=1:8500
    for k=1:30
    if y_awgn_norep16QAM(i,k)~=x_torep16QAM(i,1)
       errors_awgn_norepqam(1,k)=errors_awgn_norepqam(1,k)+1 ;
    end
    if y_Freqs_norep16QAM(i,k)~=x_torep16QAM(i,1)
       errors_Freqs_norepqam(1,k)=errors_Freqs_norepqam(1,k)+1 ;
    end
    end
end

BER_awgn_nocodeqpsk=errors_awgn_nocodeqpsk/12800 ;
BER_awgn_nocodeqpsk=BER_awgn_nocodeqpsk.' ;
BER_Freqs_nocodeqpsk=errors_Freqs_nocodeqpsk/12800 ;
BER_Freqs_nocodeqpsk=BER_Freqs_nocodeqpsk.' ;
BER_awgn_norepqpsk=errors_awgn_norepqpsk/4200 ; BER_awgn_norepqpsk=BER_awgn_norepqpsk.'
;
BER_Freqs_norepqpsk=errors_Freqs_norepqpsk/4200 ;
BER_Freqs_norepqpsk=BER_Freqs_norepqpsk.' ;
BER_awgn_nocodeqam=errors_awgn_nocodeqam/25600 ;
BER_awgn_nocodeqam=BER_awgn_nocodeqam.' ;
BER_Freqs_nocodeqam=errors_Freqs_nocodeqam/25600 ;
BER_Freqs_nocodeqam=BER_Freqs_nocodeqam.' ;
BER_awgn_norepqam=errors_awgn_norepqam/8500 ; BER_awgn_norepqam=BER_awgn_norepqam.' ;
BER_Freqs_norepqam=errors_Freqs_norepqam/8500 ; BER_Freqs_norepqam=BER_Freqs_norepqam.'
;
%%%%% Plotting %%%%%
Eb_No_nocode=10*log10(Eb_nocode/No); Eb_No_nocode=(Eb_No_nocode).' ;
Eb_No_rep=10*log10(Eb_rep/No); Eb_No_rep=(Eb_No_rep).' ;
figure(1)
slg=semilogy(Eb_No_nocode,BER_awgn_nocodeqpsk,'k',Eb_No_nocode,BER_awgn_norepqpsk,'r')
;
slg(1).LineWidth=2;slg(2).LineWidth=2;
xlabel('Eb/No(dB)');
ylabel('BER') ;
legend('No code','Repetition') ;
title('QPSK AWGN') ;
hold on
figure(2)
slg2=semilogy(Eb_No_nocode,BER_Freqs_nocodeqpsk,'k') ;
slg2(1).LineWidth=2;
xlabel('Eb/No(dB)');
ylabel('BER') ;
legend('No code') ;
title('QPSK Frequency selective') ;
hold on

figure(3)
slg3=semilogy(Eb_No_nocode,BER_Freqs_norepqpsk,'r') ;
```

```
slg3(1).LineWidth=2;
xlabel('Eb/No(dB)');
ylabel('BER') ;
legend('Repetition') ;
title('QPSK Frequency selective') ;
hold on

figure(4)
slg4=semilogy(Eb_No_nocode,BER_awgn_nocodeqam,'k',Eb_No_nocode,BER_awgn_norepqam,'r') ;
slg4(1).LineWidth=2;slg4(2).LineWidth=2;
xlabel('Eb/No(dB)');
ylabel('BER') ;
legend('No code','Repetition') ;
title('16QAM AWGN') ;
hold on
figure(5)
slg5=semilogy(Eb_No_nocode,BER_Freqs_nocodeqam,'k') ;
slg5(1).LineWidth=2;
xlabel('Eb/No(dB)');
ylabel('BER') ;
legend('No code') ;
title('16QAM Frequency selective') ;
hold on

figure(6)
slg6=semilogy(Eb_No_nocode,BER_Freqs_norepqam,'r') ;
slg6(1).LineWidth=2;
xlabel('Eb/No(dB)');
ylabel('BER') ;
legend('Repetition') ;
title('16QAM Frequency selective') ;

hold off
```

Water-filling

```
f=0:15 ;
h=[0.4 ;0 ;0.26 ;0 ;0 ;0.4 ;0 ;0.6 ;0 ;0.5];
H_freq_mag=abs(fft(h,16)) ;
H_freq_mag_squared=(H_freq_mag).^2 ;
scaled_noise_power=2./H_freq_mag_squared ;
figure(1)
bar(f,scaled_noise_power,1) ;
xlabel('Frequency');
ylabel('scaled noise power') ;
title('scaled noise power before waterfilling') ;
hold on
p=[20.47 ; 18.01 ; 18.43 ; 0 ; 2.65 ; 5.675 ; 15.71 ; 20.26 ; 18.06 ; ...
    20.26 ; 15.71 ; 5.675 ; 2.65 ; 0 ; 18.43 ; 18.01] ;
water_filled_power=[scaled_noise_power p] ;
figure(2)
bar(f,water_filled_power,1,'stacked') ;
xlabel('Frequency');
ylabel('scaled noise power') ;
title('scaled noise power after waterfilling') ;
hold off
```