TEST WITH CHANNEL 2

**Matlab Code:**

clear all

clc

SNR = 0:2:18;

k = 10^7; %

M = 2;

f = [0.227, 0.46, 0.688, 0.46, 0.227];

m = randi(2,[1,k]);

Es = 1;

inPhase = sqrt(Es)\*cos(2\*pi\*(m-1)/M);

quadrature = sqrt(Es)\*sin(2\*pi\*(m-1)/M);

signal = inPhase+(i\*(quadrature));

noisevar = 1/sqrt(2)\*(randn(1,k) + i\*randn(1,k));

r=conv(signal,f);

% Detection part

for n = 1:length(SNR)

rl = r(3:length(r)-2);

rl = rl + 10^(-SNR(n)/20)\*noisevar;

r1 = real(rl);

r2 = imag(rl);

% recieved vectors

s\_cap = r1;

s\_cap1 = r2;

for i = 1:k

if s\_cap(i) ~= 0 && s\_cap(i) > 1/(sqrt(2))

s\_cap(i) = 1;

else if s\_cap(i)~= 0 && s\_cap(i) < (-1/sqrt(2))

s\_cap(i) = -1;

else

s\_cap(i) = 0;

end

end

if s\_cap1(i) ~= 0 && s\_cap1(i) > 1/(sqrt(2))

s\_cap1(i) = 1;

else if s\_cap1(i)~= 0 && s\_cap1(i) < (-1/sqrt(2))

s\_cap1(i) = -1;

else

s\_cap1(i) = 0;

end

end

end

s\_cap2 = s\_cap + (i\*s\_cap1);

error(1,n) = size(find([signal-s\_cap2]),2);

end

SER = error/k;

close all;

hold all

semilogy(SNR,SER,'Linewidth',2);

xlabel('SNR (dB)')

ylabel('symbol error probability')

legend('SER')

grid on

axis tight;

title('Symbol error probability for given channel');

TEST WITH CHANNEL 1

clear all

clc

SNR = 0:2:18;

k = 10^7;

M = 2;

f = [0.407, 0.815, 0.407];

m = randi(2,[1,k]);

Es = 1;

inPhase = sqrt(Es)\*cos(2\*pi\*(m-1)/M);

quadrature = sqrt(Es)\*sin(2\*pi\*(m-1)/M);

signal = inPhase+(i\*(quadrature));

noisevar = 1/sqrt(2)\*(randn(1,k) + i\*randn(1,k));

r=conv(signal,f);

% Detection part

for n = 1:length(SNR)

rl = r(2:length(r)-1);

rl = rl + 10^(-SNR(n)/20)\*noisevar;

r1 = real(rl);

r2 = imag(rl);

% recieved vectors

%iphat = real(rl)>0;

s\_cap = r1;

s\_cap1 = r2;

for i = 1:k

if s\_cap(i) ~= 0 && s\_cap(i) > 1/(sqrt(2))

s\_cap(i) = 1;

else if s\_cap(i)~= 0 && s\_cap(i) < (-1/sqrt(2))

s\_cap(i) = -1;

else

s\_cap(i) = 0;

end

end

if s\_cap1(i) ~= 0 && s\_cap1(i) > 1/(sqrt(2))

s\_cap1(i) = 1;

else if s\_cap1(i)~= 0 && s\_cap1(i) < (-1/sqrt(2))

s\_cap1(i) = -1;

else

s\_cap1(i) = 0;

end

end

end

s\_cap2 = s\_cap + (i\*s\_cap1);

error(1,n) = size(find([signal-s\_cap2]),2);

%error1(1,n) = size(find([signal-iphat]),2);

end

SER = error/k;

%SER1 = error1/k;

close all;

hold all

semilogy(SNR,SER,'Linewidth',2);

%semilogy(SNR,SER1,'Linewidth',2);

xlabel('SNR (dB)')

ylabel('symbol error probability')

legend('SER' , 'SER1');

grid on

axis tight;

title('Symbol error probability for given channel');

TEST SCENARIO 2

clc;

clear all;

k = 10^7; %no of symbols

SNR = 0:2:18;

M = 2;

f = [0.407, 0.815, 0.407]; % channel taps

no\_taps = 3;

% Transmitter

for n = 1:length(SNR)

symbols = randi(2,[1,k]); % random symbol generation.

inphase = cos(2\*pi\*(symbols-1)/M);

quadrature = sin(2\*pi\*(symbols-1)/M);

signal = inphase + 1i\*quadrature;

r = conv(signal,f);

noise = 1/sqrt(2)\*(randn(1,k+length(f)-1) + 1i\*randn(1,k+length(f)-1));

y = r + 10^(-SNR(n)/20)\*(noise); % adding noise with 0dB

l = length(f);

for taps = 1:no\_taps

fm = toeplitz([f([2:end]) zeros(1,10\*taps+1-l+1)], [ f([2:-1:1]) zeros(1,10\*taps+1-l+1) ]);

d = zeros(1,10\*taps+1);

d(taps+1) = 1;

%c = [inv(fm)\*d.'].';

c = [fm\d.'].';

% mathched filter

yfilt = conv(y,c);

yfilt = yfilt(taps+2:end);

yfilt = conv(yfilt,ones(1,1)); % convolution

filter\_sampled = yfilt(1:1:k); % sampling at time T

% reciever

s\_cap = real(filter\_sampled);

s\_cap1 = imag(filter\_sampled);

for i = 1:k

if s\_cap(i) ~= 0 && s\_cap(i) > 1/(sqrt(2))

s\_cap(i) = 1;

else if s\_cap(i)~= 0 && s\_cap(i) < (-1/sqrt(2))

s\_cap(i) = -1;

else

s\_cap(i) = 0;

end

end

if s\_cap1(i) ~= 0 && s\_cap1(i) > 1/(sqrt(2))

s\_cap1(i) = 1;

else if s\_cap1(i)~= 0 && s\_cap1(i) < (-1/sqrt(2))

s\_cap1(i) = -1;

else

s\_cap1(i) = 0;

end

end

end

%receiver - hard decision decoding

ipHat = real(filter\_sampled)>0;

% counting the errors

error1(taps,n) = size(find([signal - ipHat]),2);

s\_cap2 = s\_cap + (i\*s\_cap1);

error(taps,n) = size(find([signal-s\_cap2]),2);

end

end

SER = error/k;

SER1 = error1/k;

%Simulated Error Probability

SER\_theoretical = 0.5\*qfunc(sqrt(10.^(SNR/10)));

%theoretical\_SER

% plot

close all

figure (1)

subplot(3,1,1);

semilogy(SNR,SER\_theoretical,'Linewidth',2);

legend('Theoretical');

title('Symbol error probability for BPSK using ZeroForcing Equalizer');

grid on

axis tight;

subplot(3,1,2);

hold all

semilogy(SNR,SER(1,:),'Linewidth',2);

semilogy(SNR,SER(2,:),'Linewidth',2);

semilogy(SNR,SER(3,:),'Linewidth',2);

legend('11-Taps','21-Taps','31-Taps');

title('Symbol error probability for BPSK using ZeroForcing Equalizer');

grid on

axis tight;

subplot(3,1,3);

hold all

semilogy(SNR,SER1(1,:),'Linewidth',2);

semilogy(SNR,SER1(2,:),'Linewidth',2);

semilogy(SNR,SER1(3,:),'Linewidth',2);

legend('11-Taps','21-Taps','31-Taps');

title('Symbol error probability for BPSK using ZeroForcing Equalizer hat');

grid on

axis tight;

TEST 3 LMMSE

clc;

clear all;

close all;

k = 10^7; %no. of symbols

no\_taps = 3;

SNR = 0:2:18;

M=2;

%constellation

for m=1:M

H(m,1) = cos(2\*pi\*(m-1)/M);

H(m,2) = sin(2\*pi\*(m-1)/M);

end

%Transmitter

for n = 1:length(SNR)

sym = randi(2,[1,k]); % generating N random symbols

inphase = cos(2\*pi\*(sym-1)/M);

quadrature = sin(2\*pi\*(sym-1)/M);

signal = inphase+(1i\*(quadrature));

% multiple channel taps

f = [0.407,0.815,0.407];

sig\_with\_sym1 = conv(inphase,f); % convolution of random integers & taps of channel 2

sig\_with\_sym2 = conv(quadrature,f);

sig\_with\_sym=sig\_with\_sym1+(1i\*sig\_with\_sym2);

% Noise with 0db variance

noise = 1/sqrt(2)\*(randn(1,k+length(f)-1) + 1i\*randn(1,k+length(f)-1));

% Addition of nosyme to channel

y = sig\_with\_sym + 10^(-SNR(n)/10)\*(noise);

L = length(f);

% mmse equalization

for taps = 1:no\_taps

tap = 10\*taps+1;

hautocorr = conv(f,fliplr(f));

hM = toeplitz([hautocorr([3:end]) zeros(1,2\*tap+1-L)], [ hautocorr([3:end]) zeros(1,2\*tap+1-L) ]);

hM = hM + 1/2\*10^(-SNR(n)/10)\*eye(2\*tap+1);

d = zeros(1,2\*tap+1);

d([-1:1]+tap+1) = fliplr(f);

c\_mmse = [inv(hM)\*d.'].';

% matched filter

yfilt\_mmse = conv(y,c\_mmse);

yfilt\_mmse = yfilt\_mmse(tap+2:end);

yfilt\_mmse = conv(yfilt\_mmse,ones(1,1)); % convolution

filter\_sampled\_mmse = yfilt\_mmse(1:1:k); % sampling at time T

s\_cap=real(filter\_sampled\_mmse);

s\_cap1=imag(filter\_sampled\_mmse);

for i = 1:k

if s\_cap(i)~= 0 && s\_cap(i) > (1/sqrt(2))

s\_cap(i) = 1;

else if s\_cap(i)~= 0 && s\_cap(i) < (-1\*(1/sqrt(2)))

s\_cap(i) = -1;

else

s\_cap(i) = 0;

end

end

if s\_cap1(i)~= 0 && s\_cap1(i) > (1/sqrt(2))

s\_cap(i) = 1;

else if s\_cap1(i)~= 0 && s\_cap1(i) < (-1\*(1/sqrt(2)))

s\_cap1(i) = -1;

else

s\_cap1(i) = 0;

end

end

end

s\_cap2=s\_cap+(1i\*(s\_cap1)) ;

error(taps,n) = size(find([signal-s\_cap2]),2);

end

end

SER = error/k; %Simulated Error Probability

SER\_theoretical = 0.5\*erfc(sqrt(10.^(SNR/10))); % theoretical Symbol Error Rate

% plot

close all;

figure (1)

subplot(2,1,1);

semilogy(SNR,SER\_theoretical,'Linewidth',2);

grid on

legend('Theoritical');

xlabel('SNR, dB');

ylabel('Symbol Error Rate');

title('Symbol error probability for BPSK using MMSE Equalizer Channel 3');

axis tight;

2

subplot(2,1,2);

hold all

semilogy(SNR,SER(1,:),'Linewidth',2);

semilogy(SNR,SER(2,:),'Linewidth',2);

semilogy(SNR,SER(3,:),'Linewidth',2);

legend('11-Taps','21-Taps','31-Taps');

title('Symbol error probability for BPSK using ZeroForcing Equalizer');

grid on

axis tight;

TEST SCENARIO 3\_2

clc;

clear all;

close all;

k = 10^7; %no. of symbols

no\_taps = 3;

SNR = 0:2:18;

M=2;

%constallation

for m=1:M

H(m,1)=cos(2\*pi\*(m-1)/M);

H(m,2)=sin(2\*pi\*(m-1)/M);

end

%Transmitter

for n = 1:length(SNR)

sym = randi(2,[1,k]); % generating N random symbols

inphase = cos(2\*pi\*(sym-1)/M);

quadrature = sin(2\*pi\*(sym-1)/M);

signal = inphase+(1i\*(quadrature));

% multiple channel taps

f=[0.227, 0.46, 0.688, 0.46, 0.227];

sig\_with\_sym1 = conv(inphase,f); % convolution of real part of transmitted symbol & taps of channel

sig\_with\_sym2 = conv(quadrature,f); % convolution of real part of transmitted symbol & taps of channel

sig\_with\_sym = sig\_with\_sym1+(1i\*sig\_with\_sym2);

% Nosyme with 0db variance

noise = 1/sqrt(2)\*(randn(1,k+length(f)-1) + 1i\*randn(1,k+length(f)-1));

% Addition of nosyme to channel

y = sig\_with\_sym + 10^(-SNR(n)/20)\*(noise);

L = length(f);

% mmse equalization

for taps = 1:no\_taps

tap = 10\*taps+1;

hautocorr = conv(f,fliplr(f));

hM = toeplitz([hautocorr([5:end]) zeros(1,2\*tap+1-L)], [ hautocorr([5:end]) zeros(1,2\*tap+1-L) ]);

hM = hM + 1/2\*10^(-SNR(n)/10)\*eye(2\*tap+1);

d = zeros(1,2\*tap+1);

d([-2:2]+tap+1) = fliplr(f);

c\_mmse = [inv(hM)\*d.'].';

% matched filter

yfilt\_mmse = conv(y,c\_mmse);

yfilt\_mmse = yfilt\_mmse(tap+2:end);

yfilt\_mmse = conv(yfilt\_mmse,ones(1,1)); % convolution

filter\_sampled\_mmse = yfilt\_mmse(1:1:k); % sampling at time T

s\_cap=real(filter\_sampled\_mmse);

s\_cap1=imag(filter\_sampled\_mmse);

for i = 1:k

if s\_cap(i)~= 0 && s\_cap(i) > (1/sqrt(2))

s\_cap(i) = 1;

else if s\_cap(i)~= 0 && s\_cap(i) < (-1\*(1/sqrt(2)))

s\_cap(i) = -1;

else

s\_cap(i) = 0;

end

end

if s\_cap1(i)~= 0 && s\_cap1(i) > (1/sqrt(2))

s\_cap(i) = 1;

else if s\_cap1(i)~= 0 && s\_cap1(i) < (-1\*(1/sqrt(2)))

s\_cap1(i) = -1;

else

s\_cap1(i) = 0;

end

end

end

s\_cap2=s\_cap+(1i\*(s\_cap1)) ;

error(taps,n) = size(find([signal - s\_cap2]),2);

end

end

SER = error/k; % Symbol Error Rate

SER\_theoretical = 0.5\*erfc(sqrt(10.^(SNR/10))); % Theoretical Error Rate

% plot

close all;

figure (1)

subplot(2,1,1);

semilogy(SNR,SER\_theoretical,'Linewidth',2);

grid on

legend('Theoritical');

xlabel('SNR, dB');

ylabel('Symbol Error Rate');

title('Symbol error probability for QPSK using MMSE Equalizer Channel 3');

axis tight;

subplot(2,1,2);

hold all

semilogy(SNR,SER(1,:),'Linewidth',2);

semilogy(SNR,SER(2,:),'Linewidth',2);

semilogy(SNR,SER(3,:),'Linewidth',2);

legend('11-Taps','21-Taps','31-Taps');

title('Symbol error probability for BPSK using ZeroForcing Equalizer');

grid on

axis tight;