EXP 1   FADING done

```
clc, clear all; close all;

fs = 9000; f1 = 3000; f2 = 600;

f3=750;f4=950;
ts = 0:(1/fs):1-(1/fs);
signal1 = sin(2*pi*f1.*ts)+sin (2*pi*f2.*ts);
signal2 = sin(2*pi*f3.*ts)+sin (2*pi*f4.*ts);
message = signal1+signal2;
delayed = [zeros(1,99) message(1:length(message)-99)];
%FREQUENCY RESPONSE
freqres = fft(message); figure;
magnitudeplot = abs(freqres);
plot(1:fs,magnitudeplot);
title({'Input Signal';'Frequency Response'});
xlabel('Frequency(Hz)'); ylabel('Amplitude(V)');
figure; freqres2=fft(delayed);
plot(1:fs,abs(freqres2));
title({'Delayed Input signal';'Frequency Response'});
xlabel('Frequency(Hz)');
ylabel('Amplitude(V)');
%Slow and Flat Fading
h = randn + (1i*randn);
trans1 = message.*h;
freqres = fft(trans1);
magnitudeplot = abs(fft(trans1));
figure;
plot(1:fs,magnitudeplot(1:fs));
title('Slow and Flat Fading');
xlabel('Frequency(Hz)'); ylabel('Amplitude(V)');
%Slow and Frequency Selective Fading
h1 = randn + (1i*randn);
h2 = randn + (1i*randn);
trans2 = (h1.*message) + (h2.*delayed);
magnitudeplot = abs(fft(trans2));
figure; plot(1:fs,magnitudeplot);
title('Slow and Frequency Selective Fading');
xlabel('Frequency(Hz)'); ylabel('Amplitude(V)');
%Fast and Flat Fading
fd = 10;%doppler frequency
hs = randi([0 1],1,length(ts)) +(1i*randi([0 1],1,length(ts)));
[b,a] = butter(12,((2*fd)/1000));
lpf = filter(b,a,hs);
trans3 = lpf.*message;
magnitudeplot = abs(fft(trans3));
figure; plot(1:fs,magnitudeplot);
title('Fast and Flat Fading');
xlabel('Frequency(Hz)'); ylabel('Amplitude(V)');
% Fast and Frequency Selective Fading
ht = randi([0 1],1,length(ts)) +1i*(randi([0 1],1,length(ts)));
lpf1 = filter(b,a,ht); trans4 = (lpf1.*message) +(lpf1.*delayed);
magnitudeplot = abs(fft(trans4));
figure;
```

```
plot(1:fs,magnitudeplot);
title('Fast and Frequency Selective Fading');
xlabel('Frequency(Hz)'); ylabel('Amplitude(V)');
```

EXP 2 ADAPTIVE MOD

```
clc; clear; close all;

snr= randi([10,30],1,15);
snr=sort(snr);
err=[];

for i=1:length(snr)

if (snr(i)>=0 && snr(i)<13)
    a=randi([0,1],1,1000);b=pskmod(a,2);
end
if (snr(i)>=13 && snr(i)<20)
    a=randi([0,3],1,1000); b=pskmod(a,4);
end
if (snr(i)>=20 && snr(i)<26)
    a=randi([0,15],1,1000); b=qammod(a,16);
end
if (snr(i)>=26 && snr(i)<=30)
    a=randi([0,63],1,1000);b=qammod(a,64);
end

N0=1/10^(snr(i)/10);
g=awgn(b,snr(i));
n=sqrt(N0/2)*(randn(1,length(a))+ 1i*randn(1,length(a)));
f= g+n;

if(snr(i)>=10 && snr(i)<13)
    d=pskdemod(f,2);
end
if(snr(i)>=13 && snr(i)<20)
    d=pskdemod(f,4);
end
if(snr(i)>=20 && snr(i)<26)
    d=qamdemod(f,16);
end
if(snr(i)>=26 && snr(i)<=30)
    d=qamdemod(f,64);
end

[n,r]=biterr(a,d);
if(r==0)
    r=1e-7;
end
err = [err r];

end
```

```
subplot(2,1,1); xlim([10 30]);
semilogy(snr,err,'linewidth',1);
xlabel('SNR'); ylabel('BER'); title({'AMC';'SNR Vs BER'});

thruput=[];
for i=1:length(snr)
if(snr(i)>=10 && snr(i)<13)
thruput=[thruput 1]; end
if(snr(i)>=13 && snr(i)<20)
thruput=[thruput 2]; end
if(snr(i)>=20 && snr(i)<=25)
thruput=[thruput 4]; end
if(snr(i)>=26 && snr(i)<=30)
thruput=[thruput 6];
end
end

subplot(2,1,2);xlim([10 30]);ylim([0 10]);
plot(snr,thruput);
xlabel('SNR'); ylabel('Throughput'); title({'AMC';'SNR Vs Throughput'});
```

exp 3 ofdm done

```
clc; close all;
% input sequences
x=randi([0 1],1,6400); snr=0:30; biterror=[];
%modulation
y=qammod(x,16);
%parallel
p=reshape(y,80,80);
%ifft
q=ifft(p,80);
%series
s=reshape(q,1,6400);
%cyclic prefix
a=[s(6001:6400) s]
be=[];
for j=0:1:30
% h=1/sqrt(rand(1,1)+i*sqrt(rand(1,1)));
% r=h*s;
n=awgn(a,j,'measured');
% m=inv(h)*n;
m=n(401:end);

p11=reshape(m,80,80);
q11=fft(p11,80);
s11=reshape(q11,1,6400);

y11=qamdemod(s11,16);

[num1,e1]=symerr(y11,x);
be=[be e1];
```

```
end
biterror=be;

semilogy(snr, biterror,'*-k','linewidth',2);hold on;

xlabel('SNR(dB)');ylabel('BER');title('SNR VS BER PLOT');
legend('16qam');
```

EXP 4 RC CIPHER TEXT:

```
clc;
clear all;
close all;
% RC4 ENCRYPTION ALGORITHM
% INITIALISE A STATE VECTOR
S = zeros(1,8);
% GENERATE A KEY VECTOR
%KEY_STRING = 'WCN LAB'
KEY = [5 6 8 9];
K = zeros(1,8);
% GENERATE A PLAIN TEXT OF MESSAGE
%MSG_STRING = 'I NEED TO TRANSMIT'
MSG = [3 1 3 1];
KEY_LENGTH = length(KEY);
% PERFORM KEY SCHEDULING ALGORITHM
for p = 1:8
S(p) = p-1;
TEMP = mod(p-1,KEY_LENGTH)+1;
K(p) = KEY(TEMP);
% K(p)

end
% GENERATE CIPHER TEXT
q = 0;
for p = 1:8
q = mod((q+S(p)+K(p)),8)+1;
% q
S([p q]) = S([q p]);
% S([p q])
q=q-1;
end
cipher_text = [];
cipher_text_store = [];
key_stream = [];
% PSEUDO RANDOM GENERATION ALGORITHM (PRGA)
q = 0;
for p = 1:length(MSG)
q = mod((q+S(p+1)),8)+1;
S([p+1 q]) = S([q p+1]);
% S([p+1 q])
```

```
TEMP = mod((S(p+1)+S(q)),8);
% TEMP
current_key = S(TEMP+1);
% current_key
key_stream = [key_stream,current_key];
cipher_text(p) = bitxor(MSG(p),current_key)
cipher_text_store = [cipher_text_store,cipher_text];
q=q-1;
end
%cipher_text_string = cipher_text_store
% RC4 DECRYPTION
decrypted_message = [];
decrypted_msg = [];
for p = 1:length(cipher_text)
current_message = bitxor(cipher_text(p),key_stream(p));
decrypted_message = [decrypted_message,current_message];
end
decrypted_output = decrypted_message
```

EX 5 STBC ALAMOUTI (ONE METHOD)

```
clc
clear all
close all
N = 10^6; % number of bits or symbols
SNR = [0:25];
for ii = 1:length(SNR)
ip = rand(1,N)>0.5;
s = 2*ip-1;
sCode = zeros(2,N);
sCode(:,1:2:end) = (1/sqrt(2))*reshape(s,2,N/2);
sCode(:,2:2:end) = (1/sqrt(2))*(kron(ones(1,N/2),[-
1;1]).*flipud(reshape(conj(s),2,N/2)));
h = 1/sqrt(2)*(randn(1,N) + 1i*randn(1,N));
hMod = kron(reshape(h,2,N/2),ones(1,2));
n = 1/sqrt(2)*(randn(1,N) + 1i*randn(1,N));
y = sum(hMod.*sCode,1) + 10^(-SNR(ii)/20)*n;
yMod = kron(reshape(y,2,N/2),ones(1,2));
yMod(2,:) = conj(yMod(2,:));
hEq = zeros(2,N);
hEq(:,[1:2:end]) = reshape(h,2,N/2);
hEq(:,[2:2:end]) = kron(ones(1,N/2),[1;-1]).*flipud(reshape(h,2,N/2));
hEq(1,:) = conj(hEq(1,:));
hEqPower = sum(hEq.*conj(hEq),1);
yHat = sum(hEq.*yMod,1)./hEqPower;
yHat(2:2:end) = conj(yHat(2:2:end));
% receiver - hard decision decoding
ipHat = real(yHat)>0;
% counting the errors
nErr(ii) = size(find([ip- ipHat]),2);
end
simBer = nErr/N; % simulated ber
```

```
SNRLin = 10.^(SNR/10);
pAlamouti = 1/2 - 1/2*(1+2./SNRLin).^(-1/2);
berAlamouti = pAlamouti.^2.*(1+2*(1-pAlamouti));
close all
figure
semilogy(SNR,berAlamouti,'c+-','LineWidth',2);
hold on
axis([0 25 10^-5 0.5])
grid on
legend('STBC');
xlabel('SNR, dB');
ylabel('Bit Error Rate');
title('Alamouti STBC');
```

EXP 5
STBC WITH AND WITHOUT (ANOTHER METHOD)

```
clc;
 clear all;
close all;
n = randi([0,1],1,16384);
a = reshape(n,length(n),1);
bpskmod = pskmod(a,2);
snr = 0:1:30;
h1 = 2+1j; h2 = 1-2j;
y = []; M= []; Op = []; OpwoutSTBC = [];
q = 1;
for l = 1:length(snr)
    for p = 1:2:length((n))-1
        c1 = (h1*bpskmod(p,1))+(h2*bpskmod(p+1,1));
        M(p,q) = awgn(c1,snr(l),'measured');
        c2 = (-h1*conj(bpskmod(p+1,1)))+(h2*conj(bpskmod(p,1)));
        M(p+1,q) = awgn(c2,snr(l),'measured');
    end
    for r = 1:2:(length(n) -1 )
        y(r,q) = (conj(h1)*M(r,1)) + h2*conj(M(r+1,1));
        y(r+1,q) = (conj(h2)*M(r,1)) - h1*conj(M(r+1,1));
    end
    t1 = pskdemod(y,2);
    Op(l,:) = reshape (t1,1,length(n));
    [number,ratio] = biterr(Op,n);
end
semilogy(snr,ratio,'pentagram--C','Color','black')
xlabel("SNR(in dB"); ylabel("BER"); hold on
%without STBC
for l = 1:length(snr)
    rec = awgn((h1*bpskmod),snr(l),"measured");
    demod = pskdemod(rec,2);
    OpwoutSTBC(l,:) = reshape(demod,1,length(n));
    [number1,ratio1] = biterr(OpwoutSTBC,n);
end
```

```
semilogy(snr,ratio1,'-k')
grid on
legend('BER with STBC','BER without STBC')
xlabel("SNR(in dB"); ylabel("BER"); title("SNR vs BER")
```

EXP 6 ZEROFORCING

```
clc; clear all; close all;
message = randi([0,1],1,1e7);
snr = 0:2:40; mod = 2;
modulated_bpsk_msg = pskmod(message,mod);
h1 = 1+2j;
for p = 1:length(snr)
y = awgn(h1*modulated_bpsk_msg,snr(p),'measured');
wzf=inv(h1)*y;
Demodulated_BPSK_msg_with_ZFE =pskdemod(wzf,mod);
Demodulated_BPSK_msg_without_ZFE = pskdemod(y,mod);
ber1(p) = biterr(message,Demodulated_BPSK_msg_with_ZFE)/1e7;
ber2(p)= biterr(message,Demodulated_BPSK_msg_without_ZFE)/1e7;
end
semilogy(snr,ber2)
hold on
semilogy(snr,ber1,'--')
legend('BER WITHOUT ZFE','BER WITH ZFE')
title('SNR VS BER FOR BPSK')
xlabel('SNR');ylabel('BER');
```

LEAST MEAN SQUARE

```
clc; clear all; close all;
N = 1000;
data = randi([0 1],1,N);
d =2*data-1;
order=2;
a=[1.2728 -0.81];b=1;
x=filter(b,a,d);

U = zeros(2,1);
W = zeros(2,1);
eta = 0.04;
s=[]
for n = 1 : N
U(2:end) = U(1:end-1);
U(1) = x(n);
y(n) = W(:,n)'*U;
e(n) =[ d(n) - y(n)];
s=[s e(n)]
W =[W W(:,n) + (eta .* e(n) .* U)] ;
```

```
end

figure(1)
plot(0:N,W)
xlabel('No of iteration')
ylabel('filter coefficients')
figure(2)
plot(0:N-1,(s.^2))

xlabel('No of iteration')
ylabel('mean square error')
```