Carleton University

Course: ELEC 4607 Wireless Communication

Final Report

SOLUTION: MIMO + DIVERSITY

1 Introduction

This report will discuss a simulated Matlab communication system using multiple antennas for transmission and receiving. The channel will consist of Additive-White Gaussian Noise(AWGN) and will model a Multiple-Input Multiple-Output (MIMO) which has multiple channels between different transmitters and receivers. Figure 1 depicts a MIMO setup where each of the transmitters can transmitter a signal to each of the receiver antennas. The 'h' term refers to the static gain of the channel, however in a more realistic system, the channel will suffer fading effects over time as well. Additionally, the design will take advantage of diversity among using multiple transmitters and receivers to achieve a better BER.

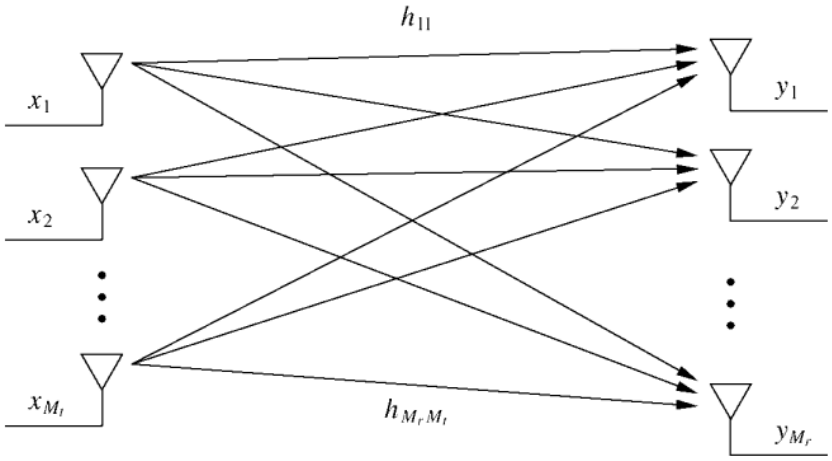

Figure 1 – MIMO [1]

2 Parameters and Design

The target for this design is to achieve a BER less than 10^{-3} at an SNR of 9dB. The system will consist of the parameters listed in Table 1. Notable parameters are packet length, which is 500 and the carrier frequency at 200kHz, which are limits per the final report instructions. Every other parameter was chosen based on previous term work, which was proven to achieve the required BER for this system. Section 3 shows the code work done for this project and how these parameters were used to achieve the target BER.

In addition to the chosen parameters below, a diversity combining technique was used to

decrease the BER of the system. This is because the MIMO alone, using 2 by 2 H matrix could not achieve the desired BER when I conducted the MIMO Matlab simulations. Therefore, incorporating Maximal Ratio Combining (MRC) into the MIMO setup allow the system to achieve and surpass the specification desired from the final report instructions. Choosing MRC technique over Selective Gain or Equal Gain Combining was based on performance results obtained from previous lab within the course. Goldsmith as confirms that a "signal transmitted or received over the multiple antennas coherently combined to maximize the channel SNR, as in MRC" [1, p.327]. The overall system design is shown in Figure 2 and brief descriptions of each block in Table 2.

Table 1 - Parameters and Specifications of Communication System

Parameters	Symbol	Values
Tested SNR Range	SNR	0 to 14 [dB]
Frames	Nf	1000
Bits per message(packet length)	Na	500
Samples per Symbol	eta	64
Symbol Duration	T	0.01 [s]
Number of transmitter/receivers	Nr	2
Carrier Frequency	Fc	200 000 [Hz]
Sampling Period	Ts	0.00015625 [samples/sec]
Modulation		16QAM
Pulse Shape		Rectangular (NRZ) with amplitude of $\frac{1}{\sqrt{T}}$

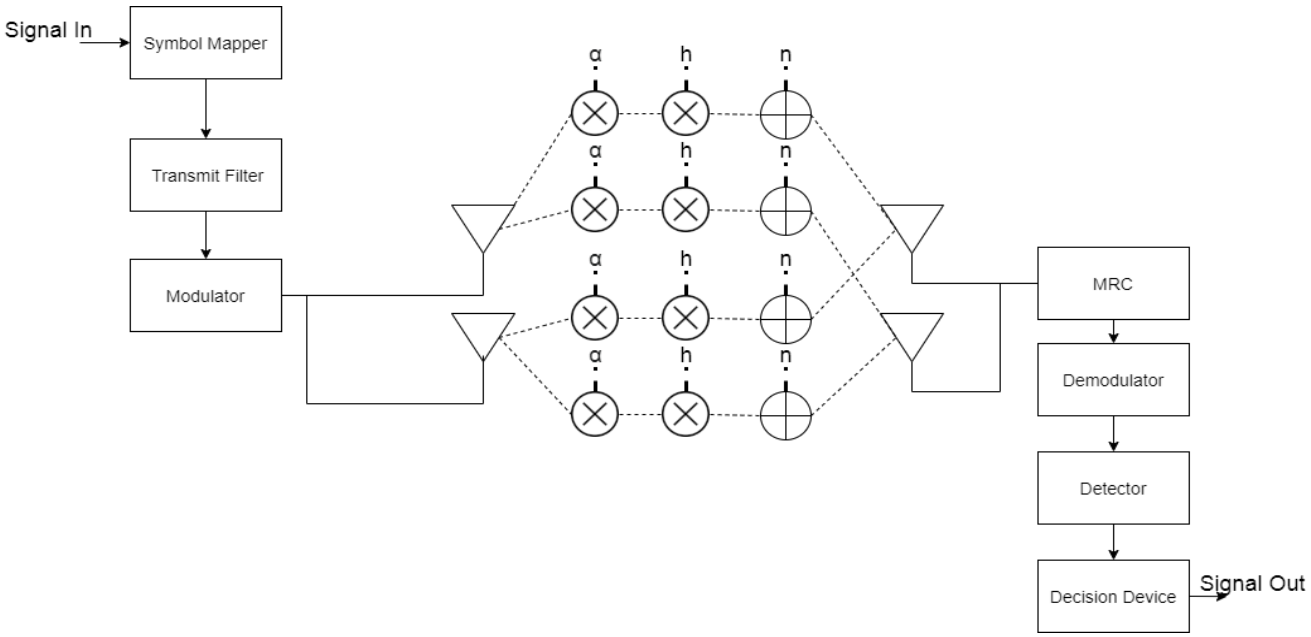

Figure 2 - Block Diagram of Design

Table 2- Descriptions of Blocks

Block	Brief Description	Shown at line
Symbol Mapper	This block converts bits into amplitudes for the transmit filter	44
Transmit Filter	Takes the bits and generates an analog signal	45
Modulator	Up-converts the baseband signal to higher frequencies	47
Alpha(α)	Fading gain and phase shifts gets multiplied by the signal's values	54-62
Static Channel Gain/Attenuation(h)	Static channel gain/attenuation and phase shift that also multiplies with the signal's values (same all the time)	54-62
Noise(n)	Standard Additive White Gaussian Noise, that adds to the signal values	54-62
MRC	Maximal Ratio Combining uses this formula: $$\mathrm{zn} = \frac{\sum_l^L \alpha_{n,l} \cdot \mathrm{e}^{-\mathrm{j}\phi\mathrm{n}} \cdot \mathrm{r}_{n,l}}{\sum_l^L \alpha^2}$$ where α is real portion of the fading and ϕ is the phase shift of the fading. 'r' represents the received signal and L the number of received sources. This technique effectively weights the signals and gives priority to the signal values with the highest SNR.	63 to 68
Demodulator	Opposite of Modulator; used to down-convert the baseband signal to lower frequencies	70
Detector	Takes an estimation of the amplitude sent bit	72
Decision Device	Makes a guess of what should be the actual bit sent	73 to 76

3 Code

Below in Table 3, shows the code used to generate a single run using MIMO and MRC setup. Most of the code is used from previous labs except for the channel code and MRC code where it had to be manipulated to fit into a MIMO system. The MIMO system is implemented by repeating the same signal on each transmitting antenna which then gets received by each of the receiving antennas. This creates an H matrix of the different paths the signal has taken to reach each of the receivers and is shown at line 60 where two repeating signals 'vct' is multiplied with the H matrix. Additionally, fading was added into the simulation shown by line 58 which also get multiplied by the signal as a more realistic approach. These attenuation and phase shift coefficients combined create losses and errors on the signal, which occurs at line 61 in the code.

The signal is then summed and manipulated according to the MRC equation presented in Table 2. The MRC then weights the signals according to the highest SNR to obtain an improvement in the overall BER of the system.

Table 3 - Code used for Designed System

Line #	Code used to generate BER vs SNR, and PSD vs Frequency
1	`%16QAM`
2	`%works with 2020b matlab`
3	`clc`
4	`clear`
5	`SNRdB = 0:14; % SNR (in dB)`
6	`Nf = 1000; % number of frames to process`
7	`Na = 500; % number of message bits`
8	`eta = 64; % number of samples per symbol`
9	`T = 0.01; % symbol duration (seconds/symbol)`
10	`fc = 200E3; % carrier frequency (Hz)`
11	`Ts = T / eta; % sampling period (samples/second)`
12	`ant=2;%number of attennas/recievers`
13	`% Symbol Map (16-QAM)`
14	`%1000 1001 1011 1010`
15	`%1100 1101 1111 1110`
16	`%0100 0101 0111 0110`
17	`%0000 0001 0011 0010`
18	`SM = [-3-3i -1-3i 3-3i 1-3i ...`
19	`  -3-1i -1-1i 3-1i 1-1i ...`
20	`  -3+3i -1+3i 3+3i 1+3i ...`
21	`  -3+1i -1+1i 3+1i 1+1i];`
22	`M = length(SM); % Number of symbols in the constellation`
23	`Nm = log2(M); % Number of bits per symbol`
24	`Es = SM*SM' / M; % Energy per symbol`
25	`Eb = Es / Nm; % Energy per bit`
26	`% Pulse Shape`
27	`hT = 1/sqrt(T) * ones(1, eta); % rectangular (NRZ)`
28	`% Matched Filter`
29	`hR = fliplr(hT); % receive filter (samples)`
30	`Nv = Na/Nm ; % number of symbols per frame`
31	`Ns = Nv * eta; % number of samples per frame`
32	`t = (0:Ns-1)*Ts; % time`
33	
34	`%H will change as time progresses`
35	`H = 1/sqrt(2) * (randn(ant,ant) + j * randn(ant,ant));%Generate random channel gain matrix`
36	
37	`for ni = 1:length(SNRdB) % Loop over each SNR`
38	`     nErrs = 0;`
39	`     No = Eb * 10.^(-SNRdB(ni)/10);`
40	`         for i=1:Nf % Loop over each message word`
41	`         % Transmitter`
42	`         a = randi([0 1], Nm, Nv); % Source`
43	`         d = 2.^(Nm-1:-1:0) * a; % Group bits into integers`
44	`         v = SM(d+1); % Symbol Mapper`
45	`         vt = conv(upsample(v, eta), hT); % Pulse Shaping Filter`
46	`         vt = vt(1:Ns); % Truncate`
47	`         vct = real(vt .* sqrt(2) .* exp(j*2*pi*fc*t)); % Modulator`
48	`         vcct=zeros(ant,length(vct));`
49	
50	`         %zero arrays`

```
         bottom=zeros([1 length(vct)]);
         top=zeros([1 length(vct)]);

         for k=1:ant
            for n=1:ant
                % (Channel gain matrix multiplied with the signal vector plus AWGN) in cell array denoted by '{}'
                awgn=sqrt(1/Ts*No/2)*randn(1, length(vct));% AWGN
                hn = (sqrt(1/2) * [1 j] * randn(2, length(vcct))); %fading process
                Hm= hn*H(k,n);
                rct(k,n) = {(vct.*Hm) + awgn};
                %implementing MRC (maximal ratio combining)
                %takes a weighted sum of all channels and heavily weights the best SNR of tested channels
                top=top+(exp(-j .*angle(Hm)).*rct{k,n} .*abs(Hm)) ;
                bottom=bottom+(abs(Hm).^2);
            end
         end
         % Receiver
         rcct= top./bottom;
         %Recieved signal is chosen by best gain channel and now is demodulated
         rot = rcct .* sqrt(2) .* exp(-j*2*pi*fc*t); % Demodulator
         rt = conv(rot, hR) * Ts; % Matched Filter
         r = rt((1:Nv)*eta); % Downsample
         test1=abs(r.' - SM);
         [X,I] = min(test1, [], 2);
         ah = de2bi(I-1, Nm, 'left-msb').';
         nErrs = nErrs + sum(sum(bitxor(a, ah))); % Count Errors
         end
     disp(sprintf('SNR: %g dB', SNRdB(ni)));
     disp(sprintf('Number of errors: %d', nErrs));
     disp(sprintf('BER: %g', nErrs/(Na*Nf)));
     totErrs(ni) = nErrs;
end
Pb_sim = totErrs / (Nf*Na);
%PSD calc
vt = vt(1:Ns); % Truncate to first Ns samples
Vf = fftshift(fft(vt)); % Calculate FFT
PSD = Vf.*conj(Vf) * Ts / Ns; % Calculate PSD

figure(1)
Pb_theory = 4/log2(M) * 0.5*erfc(sqrt(3*10.^(SNRdB/10)*log2(M)/(M-1)/2));
%a better approximation
Pb_theory = 4/log2(M) * (1-1/sqrt(M))* 0.5*erfc(sqrt(3*10.^(SNRdB/10)*log2(M)/(M-1)/2));
semilogy(SNRdB, Pb_sim, 'b',SNRdB, Pb_theory, 'k');
xlabel('E_b/N_0 (dB)');
xlabel('BER');
legend('Simulated', 'Theoretical');
grid on

%PSD graph
figure(2)
faxis = (-0.5:1/length(vcct):0.5-1/length(vcct))*1/T*eta;
plot(faxis/1000,10*log10(PSD)) %experimental
filttime = zeros(1,length(Vf));
filttime(1:eta) = 1/sqrt(T);
Vf= fftshift(fft(filttime));
PSD1 = Vf .* conj(Vf)*T/length(filttime);
hold on
%plot(faxis/1000,10*log10(PSD1),'r'); % theoretical
legend('Experimental')
xlabel('Frequency (kHz)')
ylabel('PSD')
```


Below is an example of H gain/attenuation matrix values that show a magnitude greater than 0.2 as required by the final manual.

Table 4- Code table showing generation of H matrix

Code used to generate H matrix	Output
`H = 1/sqrt(2) * (randn(ant,ant) + j * randn(ant,ant));%Generate random channel gain matrix`	H = -1.0659 - 1.0557i 0.1484 + 0.2728i 0.7864 + 0.2423i -0.5397 - 0.4943i

4 Result

Figure 3 shows the result of the code from table 3, which indicates that using MRC and MIMO setup will achieve a BER lower than 10^{-3} at SNR 9dB. Indicating the design of the communication system is successful in meeting the requirements set in section 2.

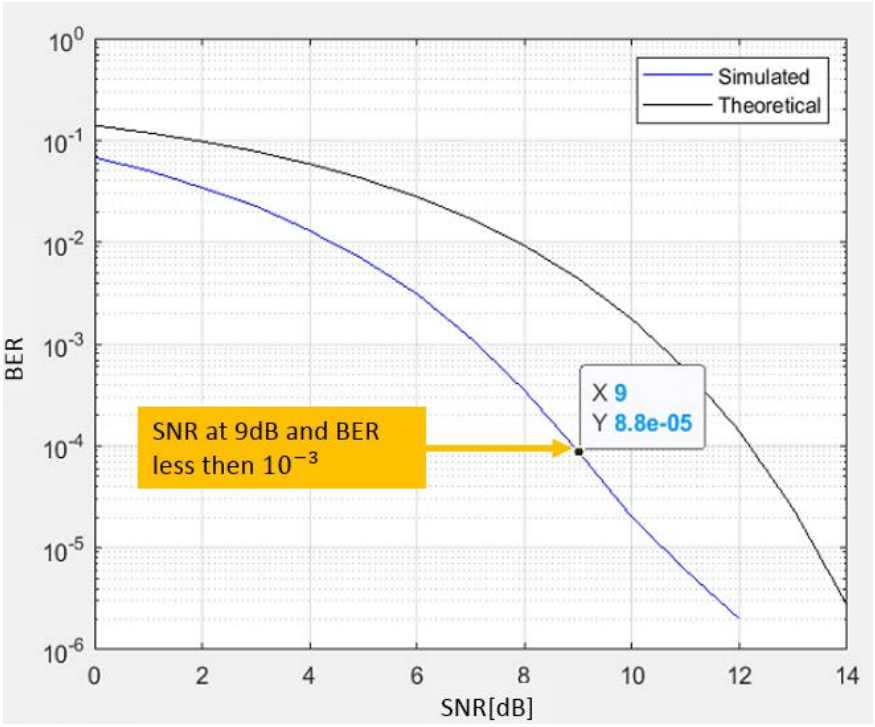

Figure 3 - BER vs SNR of a MIMO + MRC single simulation run

Furthermore, Figure 4 shows 10 simulations using the code presented in the appendix to prove that the system can repeatedly achieve a BER less than 10^{-3} at an SNR of 9dB.

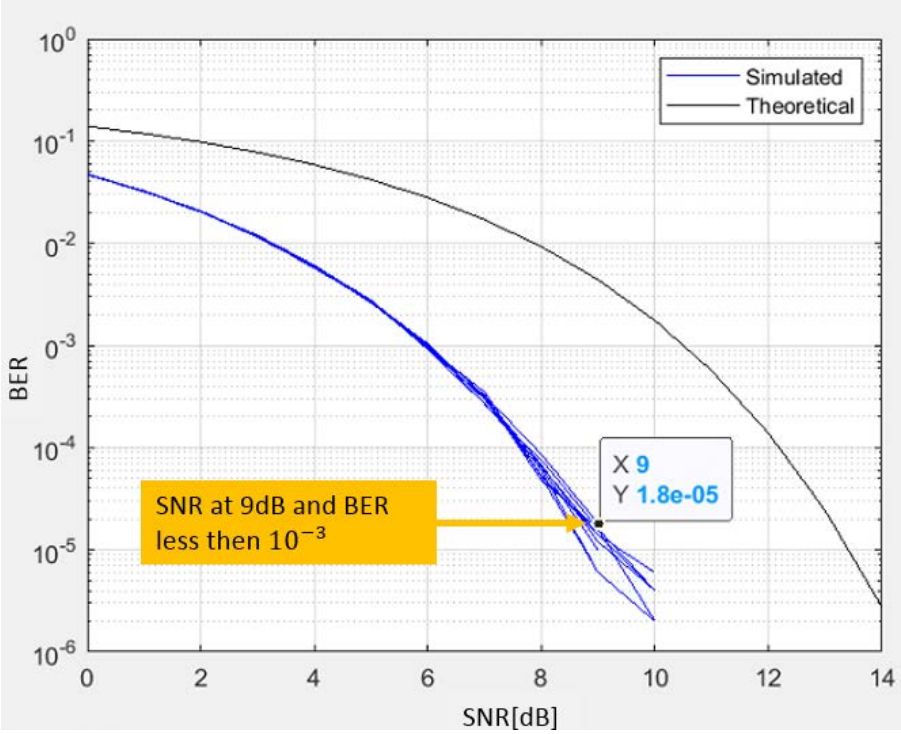

Figure 4 – BER vs SNR of a MIMO + MRC 10 simulation runs

Figure 5 shows the Power Spectral Density of the vcct signal.

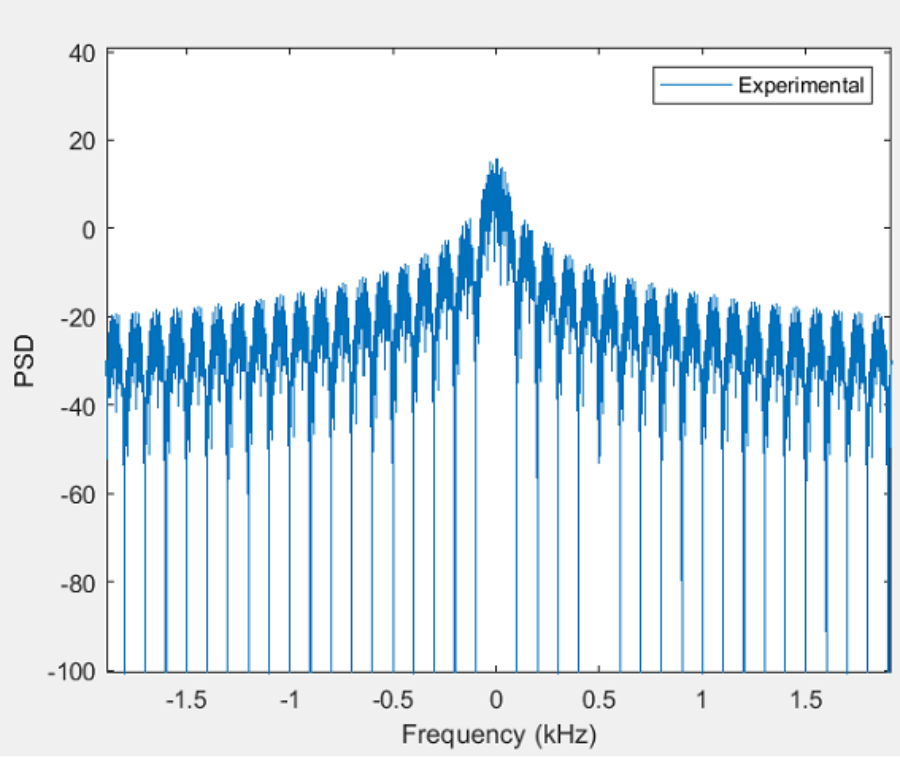

Figure 5 - PSD of vcct

5 Reference

[1] Goldsmith, A. (2005). Wireless Communications. Cambridge: Cambridge University Press. doi:10.1017/CBO9780511841224

6 Appendix

Code used to generate H matrix	Output

```
%16QAM
%works with 2020b matlab
clc
clear
SNRdB = 0:14; % SNR (in dB)
Nf = 1000; % number of frames to process
Na = 500; % number of message bits
eta = 64; % number of samples per symbol
T = 0.01; % symbol duration (seconds/symbol)
fc = 200E3; % carrier frequency (Hz)
Ts = T / eta; % sampling period (samples/second)
ant=2;%number of attennas/recievers
% Symbol Map (16-QAM)
%1000 1001 1011 1010
%1100 1101 1111 1110
%0100 0101 0111 0110
%0000 0001 0011 0010
SM = [-3-3i -1-3i 3-3i 1-3i ...
 -3-1i -1-1i 3-1i 1-1i ...
 -3+3i -1+3i 3+3i 1+3i ...
 -3+1i -1+1i 3+1i 1+1i];
M = length(SM); % Number of symbols in the constellation
Nm = log2(M); % Number of bits per symbol
Es = SM*SM' / M; % Energy per symbol
Eb = Es / Nm; % Energy per bit
% Pulse Shape
hT = 1/sqrt(T) * ones(1, eta); % rectangular (NRZ)
% Matched Filter
hR = fliplr(hT); % receive filter (samples)
Nv = Na/Nm ; % number of symbols per frame
Ns = Nv * eta; % number of samples per frame
t = (0:Ns-1)*Ts; % time

%H will change as time progresses
H = 1/sqrt(2) * (randn(ant,ant) + j * randn(ant,ant));%Generate random
channel gain matrix
for test =1:10
for ni = 1:length(SNRdB) % Loop over each SNR
     nErrs = 0;
     No = Eb * 10.^(-SNRdB(ni)/10);
         for i=1:Nf % Loop over each message word
         % Transmitter
         a = randi([0 1], Nm, Nv); % Source
         d = 2.^(Nm-1:-1:0) * a; % Group bits into integers
         v = SM(d+1); % Symbol Mapper
         vt = conv(upsample(v, eta), hT); % Pulse Shaping Filter
         vt = vt(1:Ns); % Truncate
         vct = real(vt .* sqrt(2) .* exp(j*2*pi*fc*t)); % Modulator
         vcct=zeros(ant,length(vct));

         %zero arrays
         bottom=zeros([1 length(vct)]);
```

Shown in section

```
         top=zeros([1 length(vct)]);

         for k=1:ant
            for n=1:ant
                 % (Channel gain matrix multiplied with the signal
vector plus AWGN) in cell array denoted by '{}'
                 awgn=sqrt(1/Ts*No/2)*randn(1, length(vct));% AWGN
                 hn = (sqrt(1/2) * [1 j] * randn(2, length(vcct)));
%fading process
                 Hm= hn*H(k,n);
                 rct(k,n) = {(vct.*Hm) + awgn};
                 %implementing MRC (maximal ratio combining)
                 %takes a weighted sum of all channels and heavily
weights the best SNR of tested channels
                 top=top+(exp(-j .*angle(Hm)).*rct{k,n} .*abs(Hm)) ;
                 bottom=bottom+(abs(Hm).^2);
            end
         end
         % Receiver
         rcct= top./bottom;
         %Recieved signal is chosen by best gain channel and now is
demodulated
         rot = rcct .* sqrt(2) .* exp(-j*2*pi*fc*t); % Demodulator
         rt = conv(rot, hR) * Ts; % Matched Filter
         r = rt((1:Nv)*eta); % Downsample
         test1=abs(r.' - SM);
         [X,I] = min(test1, [], 2);
         ah = de2bi(I-1, Nm, 'left-msb').';
         nErrs = nErrs + sum(sum(bitxor(a, ah))); % Count Errors
         end
     disp(sprintf('SNR: %g dB', SNRdB(ni)));
     disp(sprintf('Number of errors: %d', nErrs));
     disp(sprintf('BER: %g', nErrs/(Na*Nf)));
     totErrs(ni) = nErrs;
end
Pb_sim = totErrs / (Nf*Na);
%Pb_theory = 0.5 * erfc(sqrt(10.^(SNRdB/10))); % BPSK
%Goldsmith formula
Pb_theory = 4/log2(M) * 0.5*erfc(sqrt(3*10.^(SNRdB/10)*log2(M)/(M-
1)/2));
%a better approximation
Pb_theory = 4/log2(M) * (1-1/sqrt(M))*
0.5*erfc(sqrt(3*10.^(SNRdB/10)*log2(M)/(M-1)/2));
semilogy(SNRdB, Pb_sim, 'b',SNRdB, Pb_theory, 'k');
xlabel('E_b/N_0 (dB)');
xlabel('BER');
legend('Simulated', 'Theoretical');
grid on
hold on
end
```