DSSS

clc;

close all;

clear all;

b=round(rand(1,20));

pattern=[];

for k=1:20

if b(1,k)==0

sig=zeros(1,6);

else

sig=ones(1,6);

end

pattern=[pattern sig];

end

subplot(3,1,1);

plot(pattern);

axis([-1 130 -.5 1.5]);

title('\bf\it Original Bit Sequence');

% Generating the pseudo random bit pattern for spreading

spread\_sig=round(rand(1,120));

subplot(3,1,2);

plot(spread\_sig);

axis([-1 130 -.5 1.5]);

title('\bf\it Pseudorandom Bit Sequence');

% XORing the pattern with the spread signal

hopped\_sig=xor(pattern,spread\_sig);

% Modulating the hopped signal

dsss\_sig=[];

t=[0:100];

fc=50

c1=cos(2\*pi\*fc\*t);

c2=cos(2\*pi\*fc\*t+pi);

for k=1:120

if hopped\_sig(1,k)==0

dsss\_sig=[dsss\_sig c1];

else

dsss\_sig=[dsss\_sig c2];

end

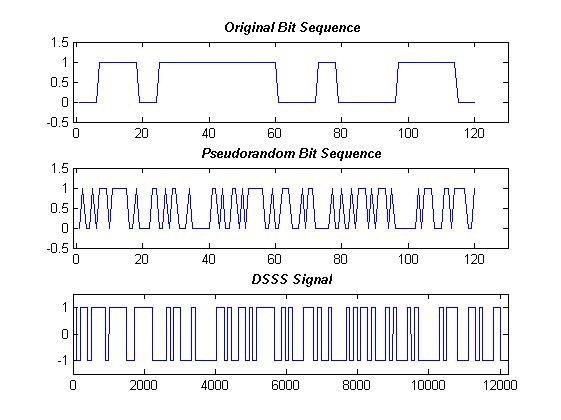
end

subplot(3,1,3);

plot([1:12120],dsss\_sig);

axis([-1 12220 -1.5 1.5]);

title('\bf\it DSSS Signal');



Clustering

clear all;

clc;

i=input('enter value of i:');

j=input('enter value of j:');

R=input('enter value of radius of cell');

cityarea=input('enter city area');

N=i^2+i\*j+j^2;

areacell= 2.5981\*R^2;

totalcells=round(cityarea/areacell)

ch\_bw=60\*10^3; % channel bandwidth given

spectr=40\*10^6; % spectrumsize given

totch=round(spectr/ch\_bw); % Total channels

k=round(totch/N)

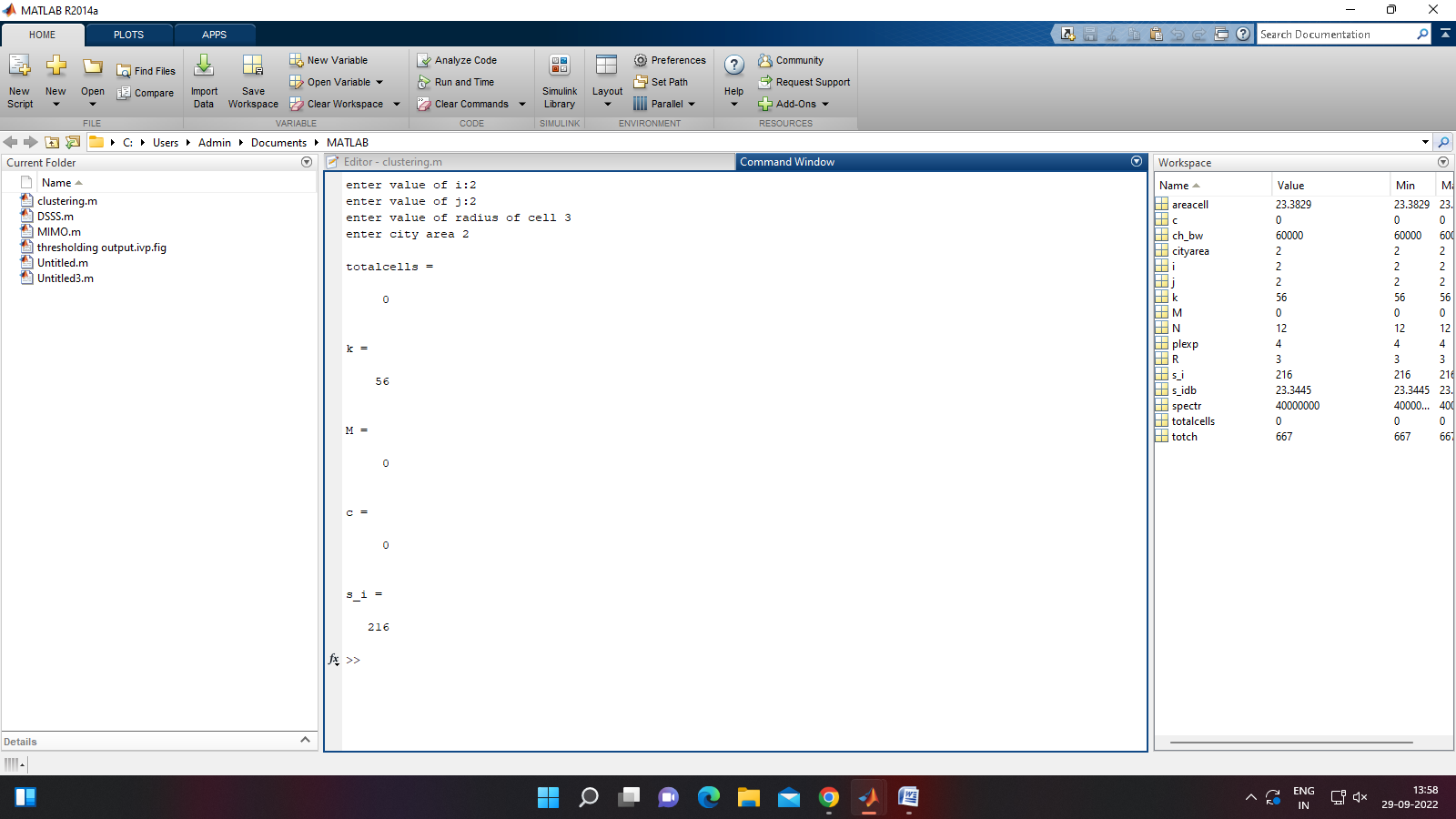
M=round(totalcells/N)

c=M\*k\*N

plexp=4; %path loss exponent

s\_i=(((3\*N)^0.5)^plexp)/6 % i0 =6 total number of co-channels

s\_idb=10\*log10(s\_i);



MIMO

clear all

clc

%Shannon capacity

snr=0;

for i = 1:10

snr = snr +2;

c=(log(1+10^(snr/10)))/log(2);

x(i)=snr;

y(i)=c;

end

figure

plot(x,y,'-','LineWidth',1.5)

hold on

% capacity of MIMO Link with NR=2, NT=2

NR=2;

rand('state',456321)

snr=0;

for i=1:10;

snr=snr+2;

for j=1:10000;

c(j)=(NR\*log(1+(10^(snr/10))\*abs(normrnd(0,1)))/log(2));

end

yy(i)=mean(c);

xx(i)=snr;

end

plot(xx,yy,':','LineWidth',1.5)

% capacity of MIMO Link with NR=3, NT=3

NR=3;

rand('state',456321)

snr=0;

for i=1:10;

snr=snr+2;

for j=1:10000;

c(j)=(NR\*log(1+(10^(snr/10))\*abs(normrnd(0,1)))/log(2));

end

yy(i)=mean(c);

xx(i)=snr;

end

plot(xx,yy,'-.','LineWidth',1.5)

% capacity of MIMO Link with NR=4, NT=4

NR=4;

rand('state',456321)

snr=0;

for i=1:10;

snr=snr+2;

for j=1:10000;

c(j)=(NR\*log(1+(10^(snr/10))\*abs(normrnd(0,1)))/log(2));

end

yy(i)=mean(c);

xx(i)=snr;

end

plot(xx,yy,'--','LineWidth',1.5)

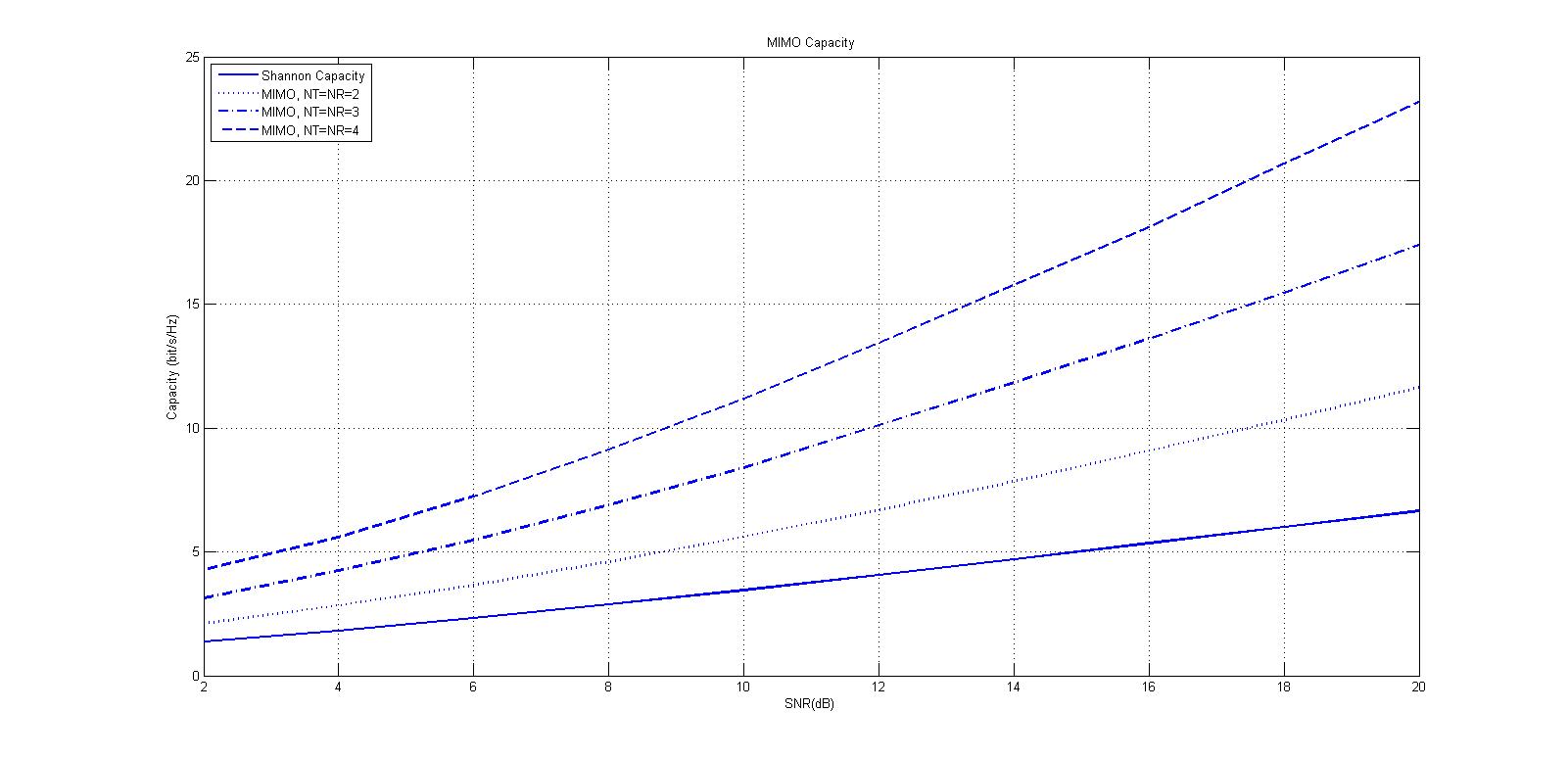
xlabel('SNR(dB)')

ylabel('Capacity (bit/s/Hz)')

grid on

legend('Shannon Capacity','MIMO, NT=NR=2','MIMO, NT=NR=3','MIMO, NT=NR=4',2)

title('MIMO Capacity')



%----Rayleigh\_PDF-----------------------------------------

%----------Input Section----------------

N=1000000; %Number of samples to generate

variance = 0.2; % Variance of underlying Gaussian random variables

%---------------------------------------

%Independent Gaussian random variables with zero mean and unit variance

x = randn(1, N);

y = randn(1, N);

%Rayleigh fading envelope with the desired variance

r = sqrt(variance\*(x.^2 + y.^2));

%Define bin steps and range for histogram plotting

step = 0.1; range = 0:step:3;

%Get histogram values and approximate it to get the pdf curve

h = hist(r, range);

approxPDF = h/(step\*sum(h)); %Simulated PDF from the x and y samples

%Theoritical PDF from the Rayleigh Fading equation

theoretical = (range/variance).\*exp(-range.^2/(2\*variance));

plot(range, approxPDF,'b\*', range, theoretical,'r');

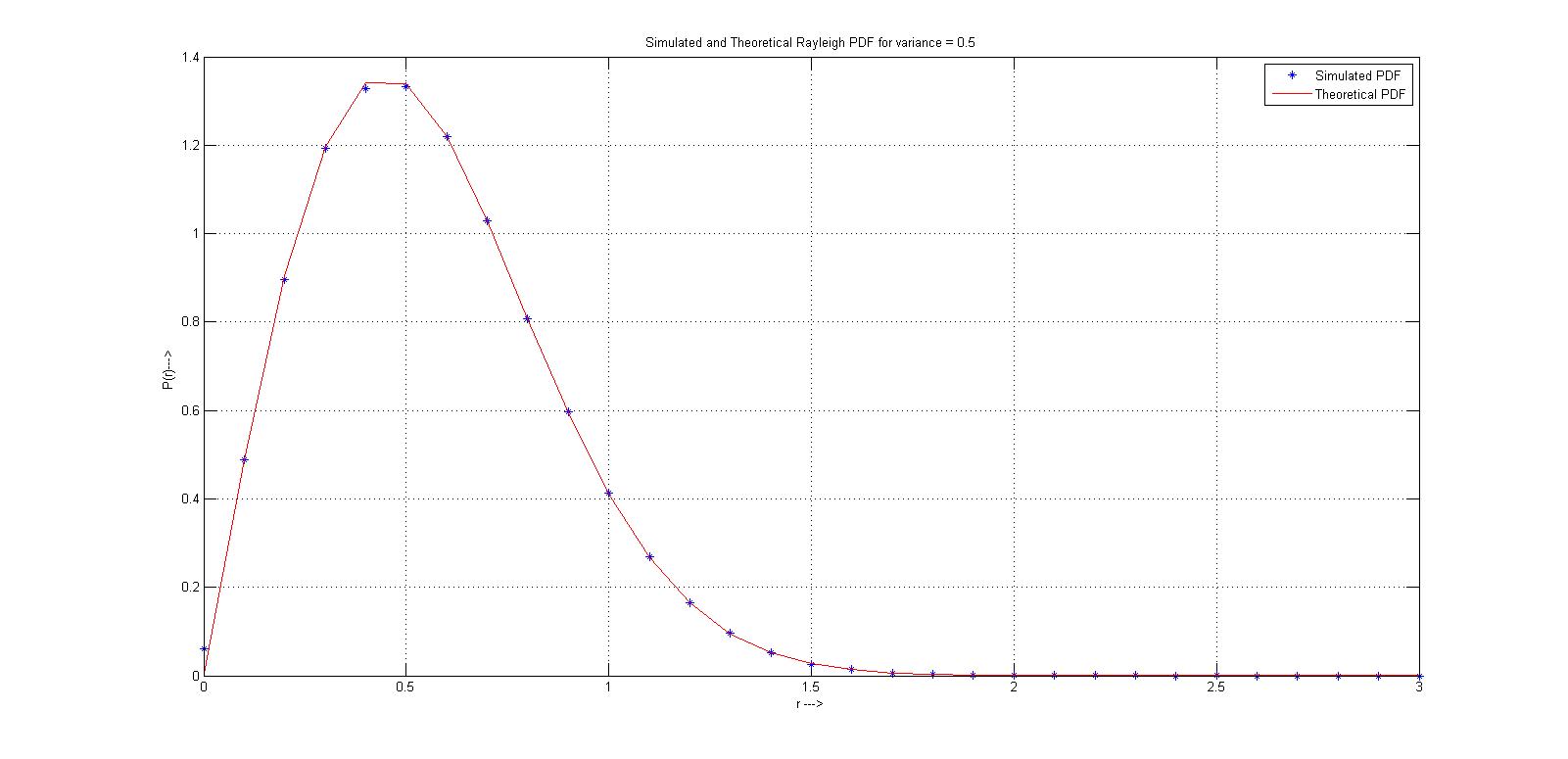
title('Simulated and Theoretical Rayleigh PDF for variance = 0.5')

legend('Simulated PDF','Theoretical PDF')

xlabel('r --->');

ylabel('P(r)---> ');

grid;



CDMA

clc

clear all

Ad=input('Enter A data : ');

Bd=input('Enter B data : ');

Ak=input('Enter key for A : ');

Bk=input('Enter key for B : ');

if (Ad == 0)

Ad=-1;

end

if (Bd == 0)

Bd=-1;

end

m=length(Ak);

n=length(Bk);

for i=1:m

if (Ak(i) == 0)

Ak(i)=-1;

end

end

for i=1:n

if (Bk(i) == 0)

Bk(i)=-1;

end

end

As=Ad\*Ak;

Bs=Bd\*Bk;

Cs=As+Bs

a=Cs.\*Ak

b=Cs.\*Bk

m=length(a);

n=length(b);

k=0;

for i=1:m

k=k+a(i);

end

k

c=0;

for i=1:n

c=c+b(i);

end

c

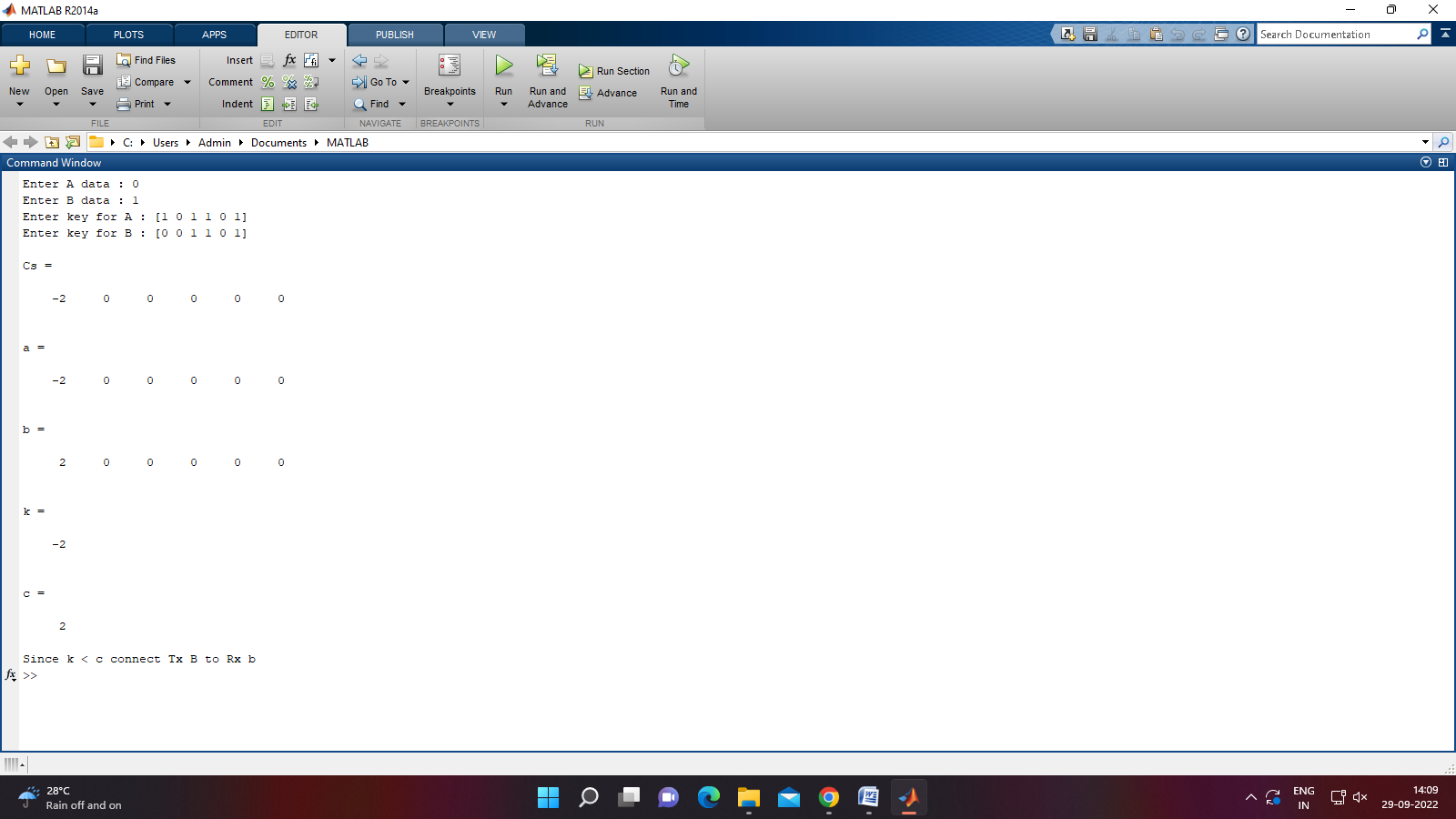
if (k > c)

disp('Since k > c connect Tx A to Rx a')

else

disp('Since k < c connect Tx B to Rx b')

end



COGNITIVE RADIO

clc;

clear all;

close all;

u = 1;

snr\_avgdB =0;

snr\_avg = power(10,snr\_avgdB/10);

Base= 0.01:0.02:1;

Pf =Base.^2;

for i=1:length(Pf)

%Threshold

Th(i) = gaminv(1-Pf(i),u,1)\*2;

pd(i) = marcumq(sqrt(snr\_avg\*2\*u),sqrt(Th(i)),u);

end

plot(Pf,pd,'->r','Linewidth',2)

holdon

snr\_avgdB =3;

snr\_avg = power(10,snr\_avgdB/10);

for i=1:length(Pf)

%Threshold

Th1(i) = gaminv(1-Pf(i),u,1)\*2;

pd1(i) = marcumq(sqrt(snr\_avg\*2\*u),sqrt(Th1(i)),u);

end

plot(Pf,pd1,'-ob','Linewidth',2)

holdon

snr\_avgdB =8;

snr\_avg = power(10,snr\_avgdB/10);

for i=1:length(Pf)

%Threshold

Th2(i) = gaminv(1-Pf(i),u,1)\*2;

pd2(i) = marcumq(sqrt(snr\_avg\*2\*u),sqrt(Th2(i)),u);

end

plot(Pf,pd2,'-\*k','Linewidth',2)

holdon

xlabel('Probability of False alarm (Pf)');

ylabel('Probability of detection (Pd)');

legend ('SNR=0db','SNR=3db','SNR=8db');

grid on;

