PRÁCTICA 5 RESUELTA + PLOTS: cfar

Radiolocalització (Universitat Politècnica de Catalunya)

PRÁCTICA 5

MAIN

```
clear all;
clear all figures;
%% 6.1 Samples at the output of a square law detector
%INPUTS: Number of samples
%OUTPUT: square law detector signal
%--> i n p u t s   a j u s t a b l e s
k=1.38064852e-23; %Bolzmann constant
To=300; %K
B=1e6; %MHz
N_samples=10000;

%--> e x e c u c i ó
noise=k*To*B;

noise_factor_inphase=randn(1,N_samples);
Pot_ni=sum((abs(noise_factor_inphase)).^2)/N_samples;
noise_inphase=sqrt(noise).*noise_factor_inphase./sqrt(Pot_ni);

noise_factor_quadrature=randn(1,N_samples);
Pot_nq=sum((abs(noise_factor_quadrature)).^2)/N_samples;
noise_quadrature=sqrt(noise).*noise_factor_quadrature./sqrt(Pot_nq);

figure(1);
histogram(noise_inphase,'Normalization','pdf');
xlabel('Inphase noise');
ylabel('PDF of inphase noise');

figure(2);
histogram(noise_quadrature,'Normalization','pdf');
xlabel('Quadrature noise');
ylabel('PDF of quadrature noise');

y=noise_inphase.^2+noise_quadrature.^2;
figure(3);
histogram(y,'Normalization','pdf');
xlabel('Square law dectector signal');
ylabel('PDF of square law dectector signal');

%% 6.2 Scaling factor alpha for a given Pfa
%---------INPUTS-----------
%   Number of samples
%   Prob of False Alarm
%   Training cells

%---------OUTPUT-------------
%   Threshold vector
%   Number of Pfa
%   False alarm probability=Number of Pfa/Number of samples

k=1.38064852e-23; %Bolzmann constant

%--> i n p u t s   a j u s t a b l e s
P_fa=0.01; %Prob of False Alarm
M=40; %Training cells
To=300; %[K]
B=1e6; %[MHz]
N_samples=10000; %Number of samples
```

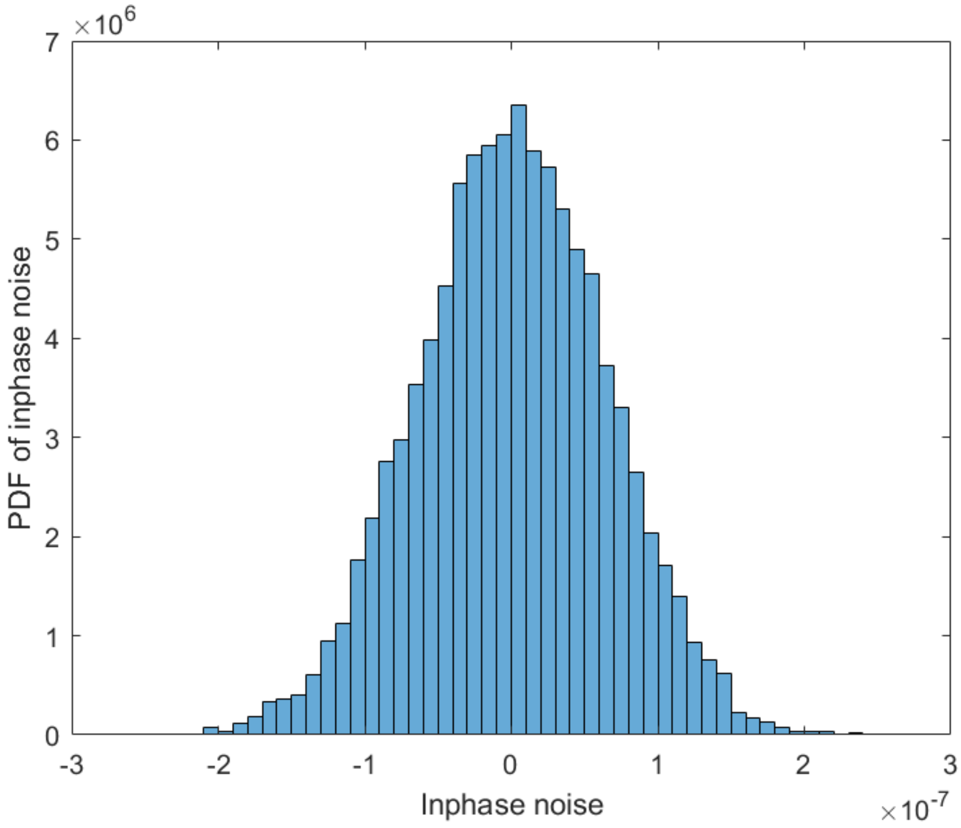
```
%--> e x e c u c i ó
noise=k*To*B;
% Knowing that Pfa=1/((1+alpha/M)^M --> isolating alpha
alpha=M*(1/(((P_fa)^(1/M)))-1);

%% 6.3 Simulation of the CA-CFAR for a single Pfa

%--> e x e c u c i ó
    %OUTPUT of SQUARE LAW DECTECTOR
n_i=randn(M+1,N_samples);
n_q=randn(M+1,N_samples);
for i=1:M+1
Pot_ni=sum(abs(n_i(i,:)).^2)/N_samples;
n_i_2=sqrt(noise).*n_i(i,:)/sqrt(Pot_ni);
Pot_nq=sum(abs(n_q(i,:)).^2)/N_samples;
n_q_2=sqrt(noise).*n_q(i,:)/sqrt(Pot_nq);
y(i,:)=n_q_2.^2+n_i_2.^2;
end

    %THRESHOLD COMPUTATION
for(i=1:N_samples)
anterior=sum(y(1:M/2,i));
posterior=sum(y(M/2+2:M,i));
suma_total(i)=sum([anterior posterior]);
llindar(i)=alpha/M*suma_total(i);
end

    %COMPARISON TO KNOW IF ITS A TARGET
Pfa_vector=zeros(1,N_samples);
Pfa_counter=0;
for i=1:N_samples
    if(llindar(i)<y(M/2+1,i))
        Pfa_vector(i)=1;
        Pfa_counter=Pfa_counter+1;
    else
        Pfa_vector(i)=0;
    end
end

    %PLOT CUT AND THRESHOLD
plot(20*log10(llindar));
hold on;
plot(20*log10(y(M/2+1,:)));
xlabel('Number of samples');
ylabel('Level of noise (dB)');
legend('Threshold (dB)','CUT (dB)');
hold off;
title('CUT and threshold coexistance');

    %False alarm probability=Number of Pfa/Number of samples
Pfa_obtained=Pfa_counter/N_samples;

%% 6.4 ROUTINE for different Pfa
%---------INPUTS-----------
%   Number of samples
%   Prob of False Alarm
%   Training cells

%---------OUTPUT-------------
%   Threshold vector
```

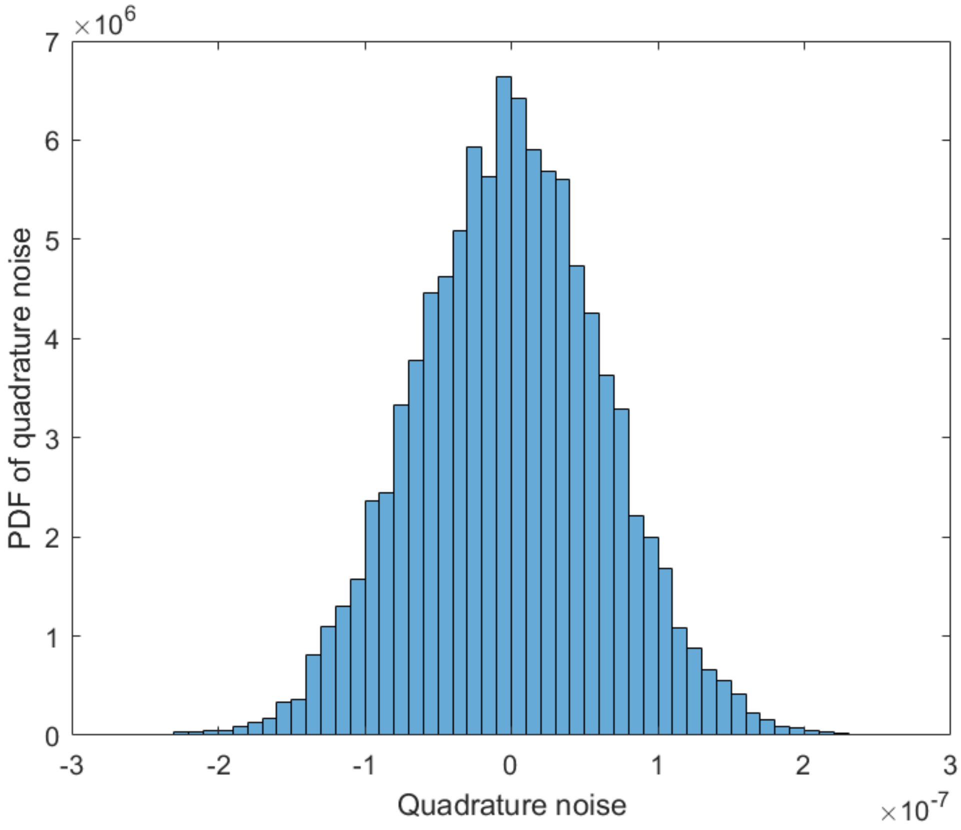
```
%   CUT vector
%   Number of Pfa
%   False alarm probability=Number of Pfa/Number of samples

M=40;
k=1.38064852e-23; %Bolzmann constant
To=300; %K
B=1e6; %MHz
N_samples=10000;
noise=k*To*B;

P_fa=[0.1 0.001 0.0001];

[Pfa_obtained, llindar, CUT, Pfa_counter]=CFAR(M,N_samples,P_fa,noise);

for i=1:length(P_fa)
    figure(i);
    plot(20*log10(llindar(i,:)));
    hold on;
    plot(20*log10(CUT));
    xlabel('Number of samples');
    ylabel('Level of noise (dB)');
    legend('Threshold (dB)','CUT (dB)');
    hold off;
    title(sprintf('CUT and threshold coexistance for probability of FA=
%g',P_fa(i)));
end
```

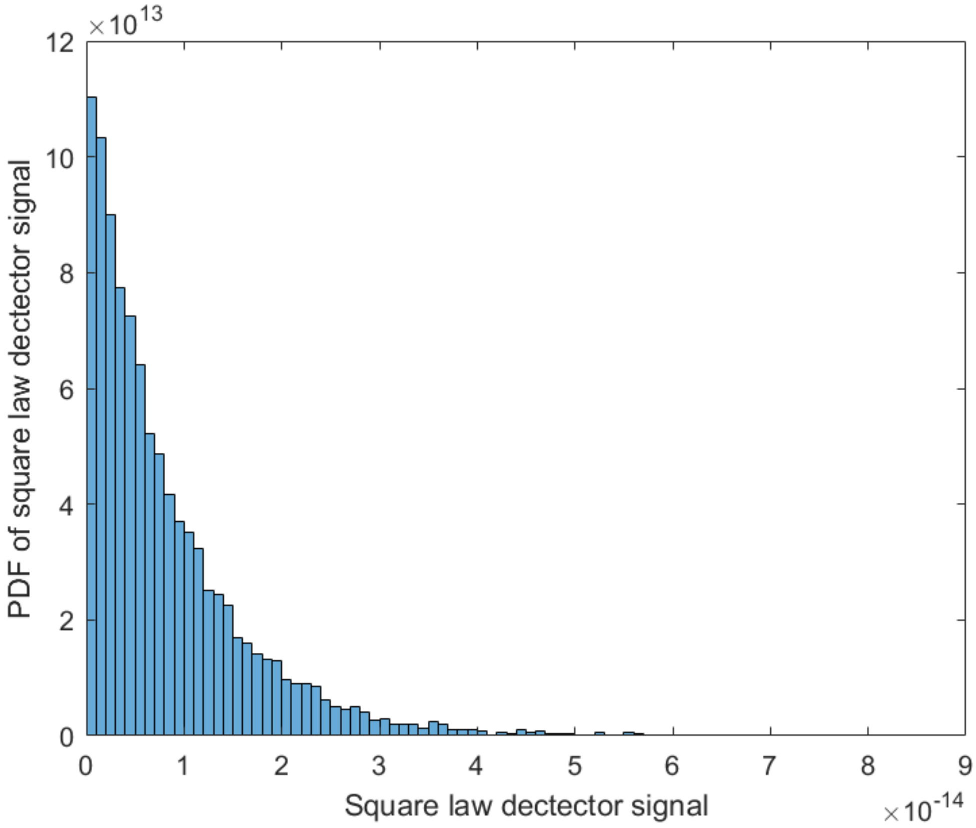
FUNCTION: CFAR

```
function [Pfa_obtained, llindar, CUT,
Pfa_counter]=CFAR(M,N_samples,P_fa,noise)
alpha=M*(1./(((P_fa).^(1/M)))-1);
n_i=randn(M+1,N_samples);
n_q=randn(M+1,N_samples);

for i=1:M+1
Pot_ni=sum(abs(n_i(i,:)).^2)/N_samples;
n_i_2=sqrt(noise).*n_i(i,:)/sqrt(Pot_ni);
Pot_nq=sum(abs(n_q(i,:)).^2)/N_samples;
n_q_2=sqrt(noise).*n_q(i,:)/sqrt(Pot_nq);
y(i,:)=n_q_2.^2+n_i_2.^2;
end

for i=1:N_samples
anterior=sum(y(1:M/2,i));
posterior=sum(y(M/2+2:M,i));
suma_total(i)=sum([anterior posterior]);
llindar(:,i)=alpha./M*suma_total(i);
end

 for j=1:length(alpha)
     Pfa_counter(j)=0;
    for i=1:N_samples
        if(llindar(j,i)<y(M/2+1,i))
            Pfa_vector(j,i)=1;
            Pfa_counter(j)=Pfa_counter(j)+1;
        else
            Pfa_vector(j,i)=0;
        end
    end
```

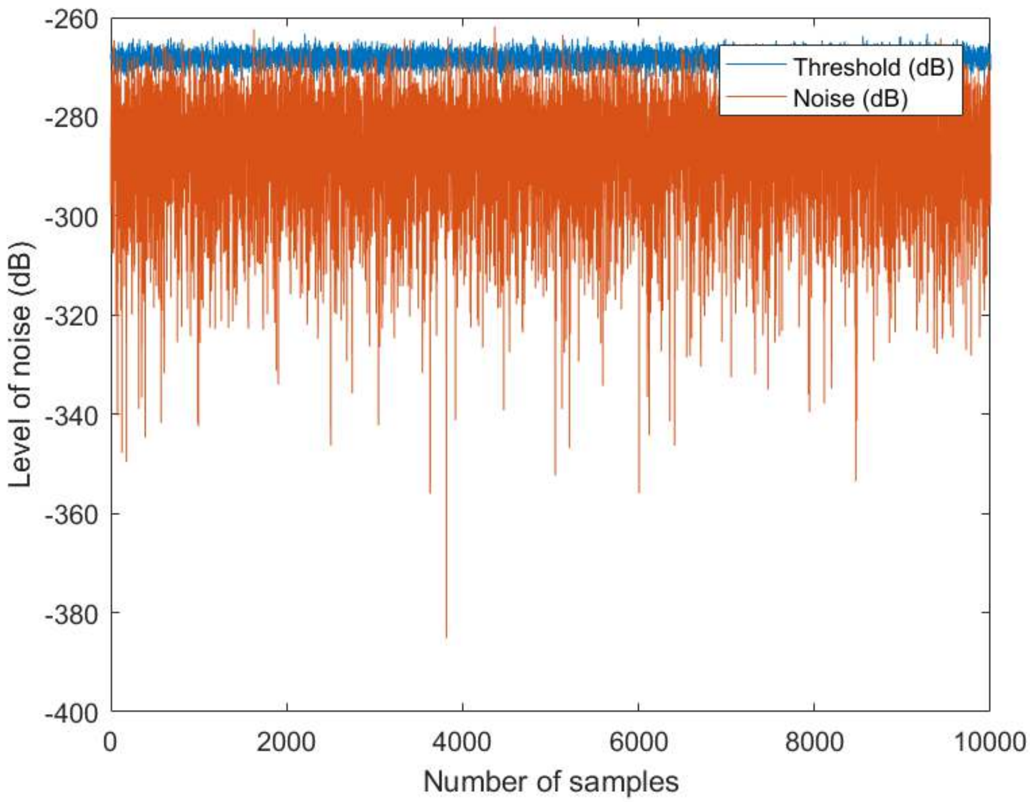
```
end
CUT=y(M/2+1,:);
Pfa_obtained=Pfa_counter/N_samples;
```



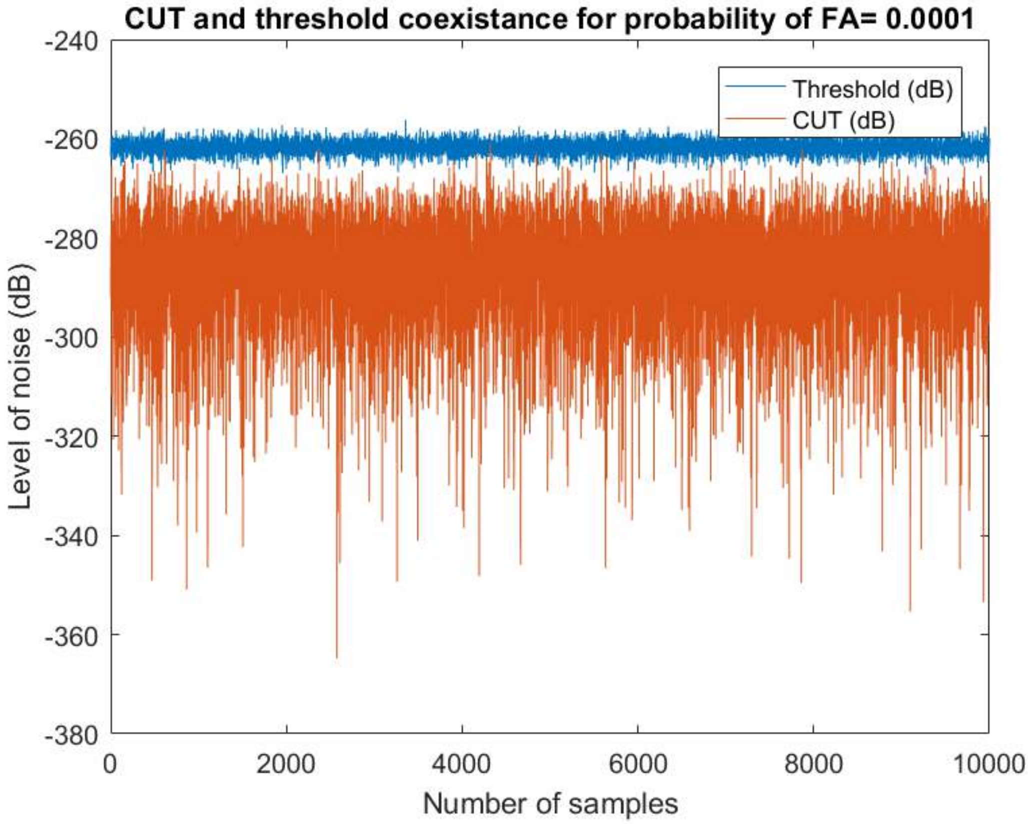


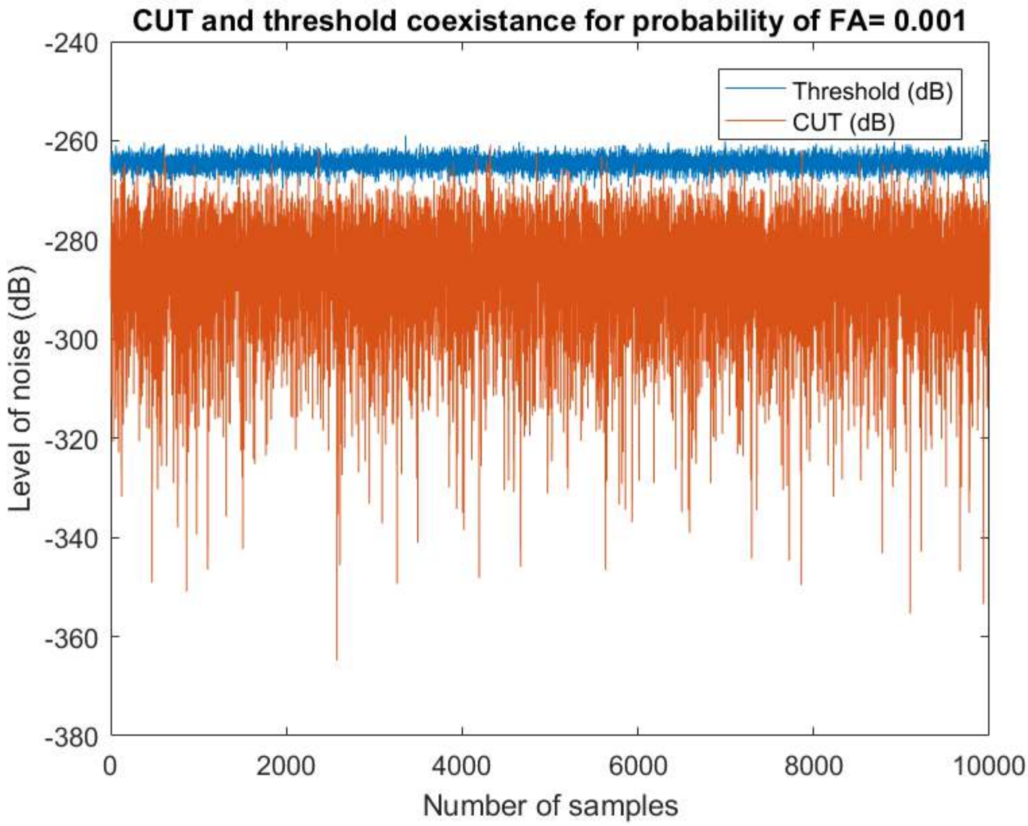


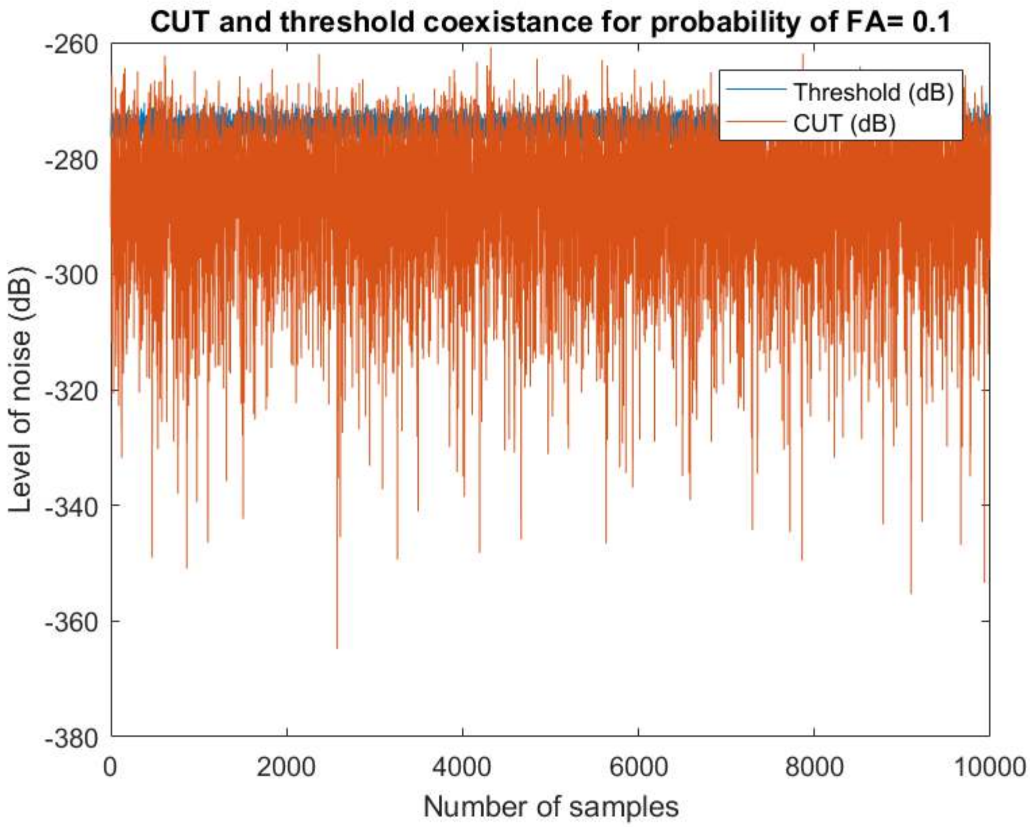
PDF of square law dectector signal
Square law dectector signal
×10^13
×10^-14



Threshold (dB)
Noise (dB)
Level of noise (dB)
Number of samples


CUT and threshold coexistance for probability of FA= 0.0001
Threshold (dB)
CUT (dB)
Level of noise (dB)
-240
-260
-280
-300
-320
-340
-360
-380
0
2000
4000
6000
8000
10000
Number of samples


CUT and threshold coexistance for probability of FA= 0.001
Threshold (dB)
CUT (dB)
Level of noise (dB)
-240
-260
-280
-300
-320
-340
-360
-380
0
2000
4000
6000
8000
10000
Number of samples


CUT and threshold coexistance for probability of FA= 0.1
Threshold (dB)
CUT (dB)
Level of noise (dB)
Number of samples
-260
-280
-300
-320
-340
-360
-380
0
2000
4000
6000
8000
10000