**ECE 254 Homework 4**

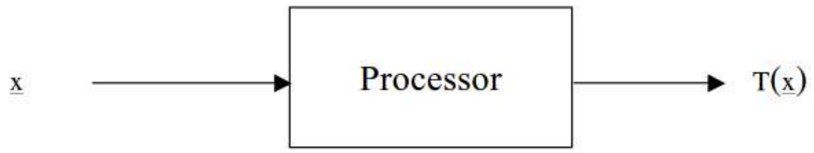
**Rayleigh Fading Signal**

**Name: Mingxuan Wang**

**Date: 2015/11/6**

- Title: Rayleigh Fading Signal
- Objective:

Consider the following processor structure:




Our goal is to decide presence or absence of a signal buried in uncorrelated Gaussian noise

where: $H_1$: $x(n) = s(n) + w(n)$ , $n = 0,1,\ldots, N\text{-}1$
$H_0$: $x(n) = w(n)$ , $n = 0,1,\ldots, N\text{-}1$
$w(n)$ is an uncorrelated Gaussian noise sequence $\sim N(0,\sigma^2)$

1. Consider three different classes of signals:

A. $s(n) = A \sin(2\pi f_c n + \phi)$ , $f_c = 1/16$

A known and $\phi$ uniformly distributed.

B. $s(n) = A \sin(2\pi f_c n + \phi)$ , $f_c = 1/16$

A Rayleigh distributed and $\phi$ uniformly distributed.

C. $s(n) = w_s(n)$

Uncorrelated Gaussian signal $\sim N(0,\sigma_s^2)$

2. Summarize briefly the analytical derivation of the test statistic and performance for the following optimum detection receivers:

A. SKEP (N = 128)

B. Rayleigh fading sinusoid (N = 128)

C. Energy detector (N = 128 and N = 16).

Express $P_D$ in terms of $P_F$ for the SKEP and Rayleigh fading sinusoid processors.

3. Plot the performance of the processors in II above as:

A. $P_D$ vs. $P_F$ on normal probability paper for 10 log (ENR) = 10 dB.

B. $P_D$ (linear) vs. ENR (dB) for $P_F = 10^{-1}$, $10^{-2}$, and $10^{-3}$ and ENR from 0 to 30 dB.

ENR is the expected energy-to-noise ratio.

- Approach:

  See handwriting.

- Results(including plots):

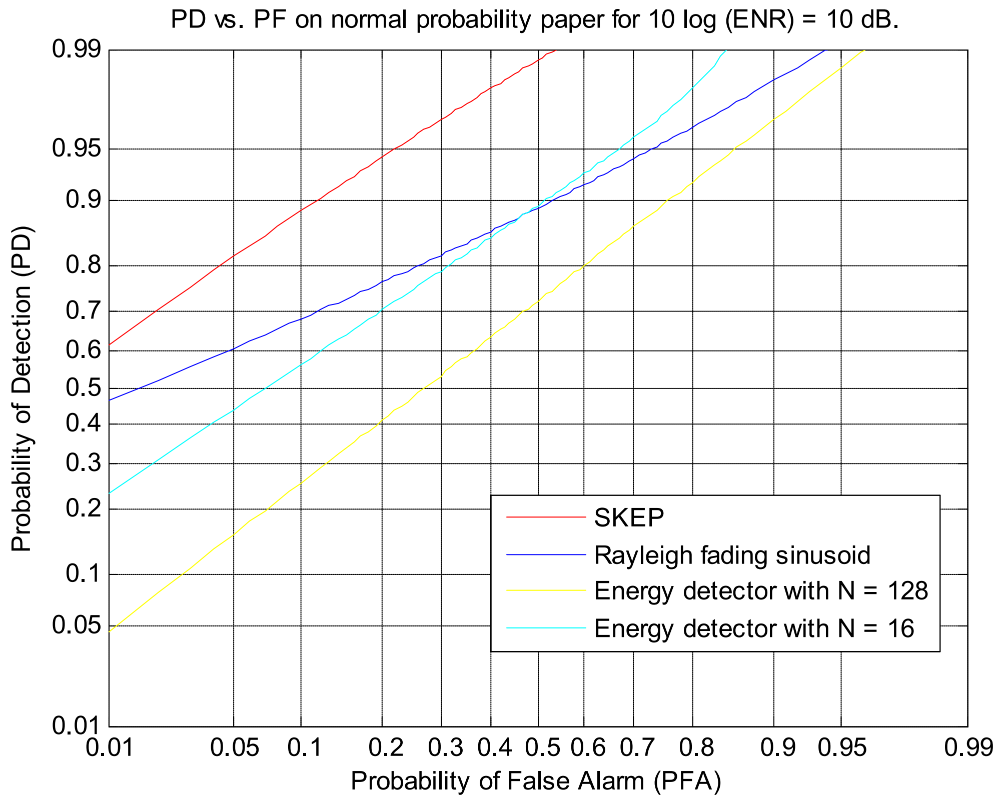



Figure 1 PD vs PF on normal probability paper for ENR = 10

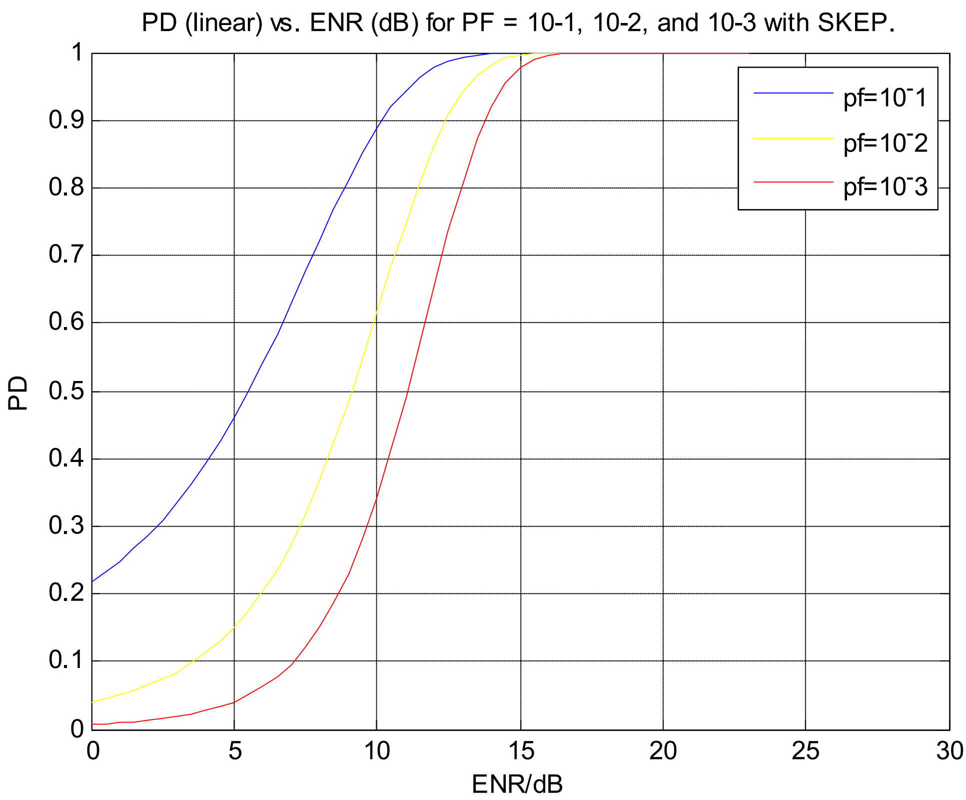



Figure 2 PD vs ENR for PF = 10^-1, 10^-2, 10^-3 with SKEP

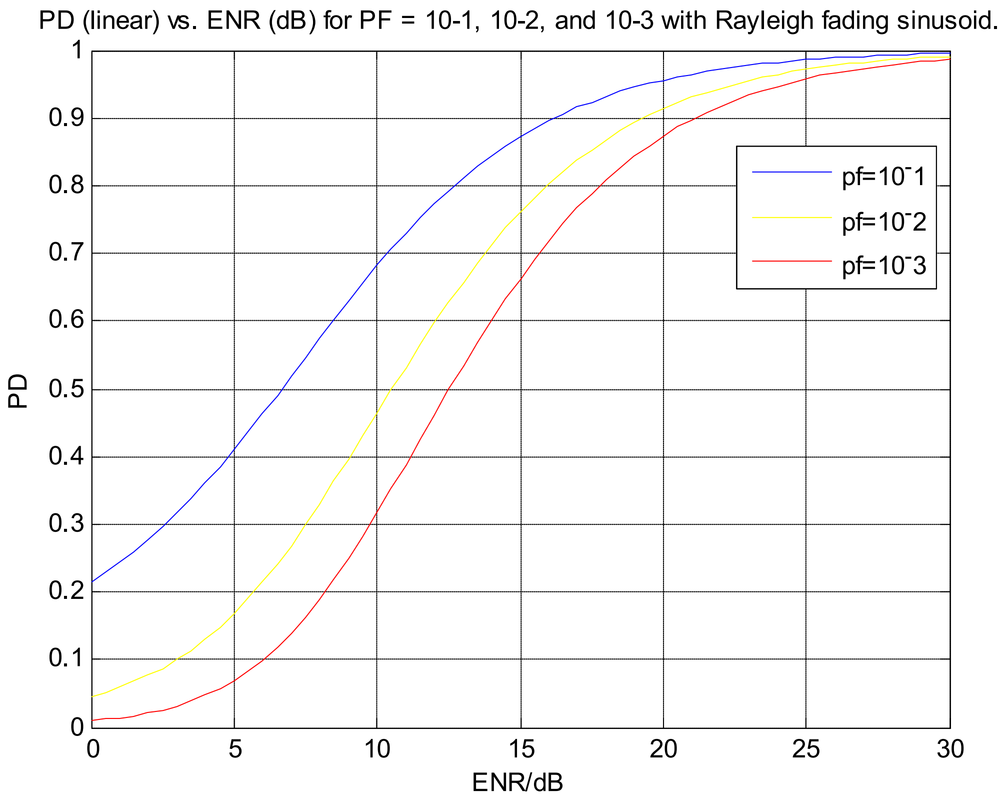



Figure 3 PD vs ENR for PF = 10^-1, 10^-2, 10^-3 with Rayleigh fading sinusoid

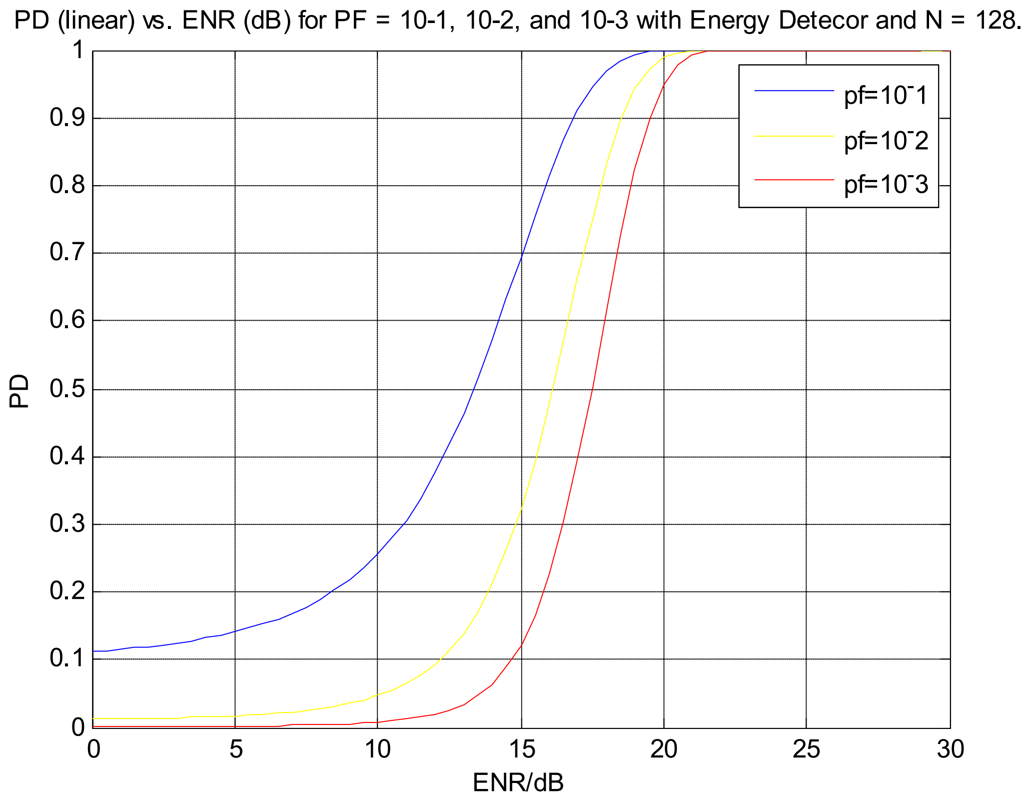



Figure 4 PD vs ENR for PF = 10^-1, 10^-2, 10^-3 with Energy Detector and N = 128

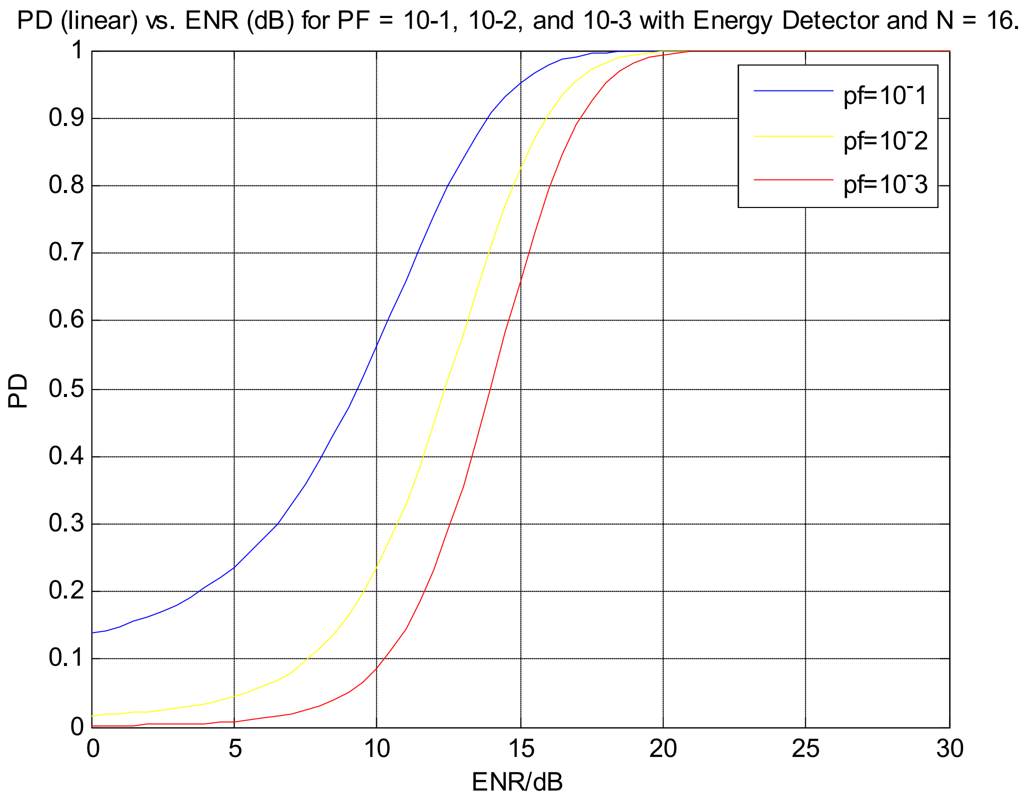



Figure 5 PD vs ENR for PF = 10^-1, 10^-2, 10^-3 with Energy Detector and N = 16

**Discussion:**

1. For figure 1, we can see that:
   a) SKEP performs the best for all time.
   b) Before around pf = 0.46, Rayleigh fading sinusoid performs the second and Energy Detector with N = 16 is the third.
   c) After around pf = 0.46, Energy Detector with N = 16 performs the second and Rayleigh fading sinusoid is the third.
   d) Energy Detector with N = 128 perform the worst.
   e) According to the formula derived in Approach part, we can see that for Energy Detector, with the increasing the N, the SNR is becoming less and less. This is what we see in figure 2 – N=16 condition is better than N = 128.
2. For figure2 – figure4:
   a) With the decrease of pf, the curve (pd vs ENR) turn to right a little bit each time.
   b) With the decrease of pf, the slope of the curve becomes steeper.
   c) With the increase of ENR, the performance becomes better.
   d) The slope of the curve in Rayleigh fading sinusoid is smaller than others. To obtains a certain pd, the ENR of Rayleigh fading sinusoid is bigger than others.
   e) We can still find that with the increasing of N, the performance of Energy Detector is worse.

- Appendix:

Hw4.m

```
%% SKEP with fixed ENR
clear;clc;
ENR = 10;
lambda = ENR;
pf1 = 0.01:0.01:1;
x1 = -2*log(pf1);
pd1 = Qchipr2(2, lambda, x1, 1e-5);
figure(1)
probpaper(pf1,pd1, 'r');
%% Rayleigh fading sinusoid with fixed ENR
pf2=0.01:0.01:1;
pd2=pf2.^(1/(1+ENR/2));
figure(1)
hold on
probpaper(pf2,pd2, 'b')
%% Energy detector with fixed ENR
% N = 128
N1=128;
pf3=0.01:0.01:1;
pd3=zeros(1,100);
for i=1:100
    r1=getgama(pf3(i),N1);
    R1=2*r1;
    gama1=R1/(ENR/N1+1);
    pd3(i)=Qchipr2(N1,0,gama1,1e-5);
end
figure(1)
hold on
probpaper(pf3,pd3, 'y')
% N = 16
N2=16;
pf4=0.01:0.01:1;
grid
r2=getgama(pf4,N2);
R2=2*r2;
gama2=R2/(ENR/N2+1);
pd4=Qchipr2(N2,0,gama2,1e-5);
figure(1)
probpaper(pf4,pd4, 'c')
legend('SKEP', 'Rayleigh fading sinusoid', 'Energy detector with N = 128', 'Energy
detector with N = 16');
```

```
%% 3B SKEP
pf1 = 10^-1;
pf2 = 10^-2;
pf3 = 10^-3;
ENR=0:0.5:30;
lambda=10.^(ENR/10);
x1=2*log(1/pf1);
x2=2*log(1/pf2);
x3=2*log(1/pf3);
pd1=zeros(1,61);
pd2=zeros(1,61);
pd3=zeros(1,61);
for i=1:61
        pd1(i)=Qchipr2(2,lambda(i),x1,1e-5);
        pd2(i)=Qchipr2(2,lambda(i),x2,1e-5);
        pd3(i)=Qchipr2(2,lambda(i),x3,1e-5);
end
figure(2)
plot(ENR,pd1,'b')
hold on
plot(ENR,pd2,'y')
plot(ENR,pd3,'r')
grid;
title('PD (linear) vs. ENR (dB) for PF = 10-1, 10-2, and 10-3 with SKEP.');
legend('pf=10^-1','pf=10^-2','pf=10^-3');
%% Rayleigh
x=10.^(ENR/10);
y=1./(x/2+1);
pd4=pf1.^y;
pd5=pf2.^y;
pd6=pf3.^y;
figure(3)
plot(ENR,pd4,'b')
hold on
plot(ENR,pd5,'y')
plot(ENR,pd6,'r')
grid;
title('PD (linear) vs. ENR (dB) for PF = 10-1, 10-2, and 10-3 with Rayleigh fading
sinusoid.');
legend('pf=10^-1','pf=10^-2','pf=10^-3');
%% Energy Detector
%N=128
N1=128;
pd7=zeros(1,61);
```

```
pd8=zeros(1,61);
pd9=zeros(1,61);
r7=getgama(pf1,N1);
R7=2*r7;
r8=getgama(pf2,N1);
R8=2*r8;
r9=getgama(pf3,N1);
R9=2*r9;
for i=1:61
        gama7=R7/(x(i)/N1+1);
        pd7(i)=Qchipr2(N1,0,gama7,1e-5);
        gama8=R8/(x(i)/N1+1);
        pd8(i)=Qchipr2(N1,0,gama8,1e-5);
        gama9=R9/(x(i)/N1+1);
        pd9(i)=Qchipr2(N1,0,gama9,1e-5);
end
figure(4)
plot(ENR,pd7,'b')
hold on
plot(ENR,pd8,'y')
plot(ENR,pd9,'r')
axis([0 30 0 1])
grid;
title('PD (linear) vs. ENR (dB) for PF = 10-1, 10-2, and 10-3 with Energy Detecor and N
= 128.');
legend('pf=10^-1','pf=10^-2','pf=10^-3');

% N= 16
N2=16;
pd10=zeros(1,61);
pd11=zeros(1,61);
pd12=zeros(1,61);
r10=getgama(pf1,N2);
R10=2*r10;
r11=getgama(pf2,N2);
R11=2*r11;
r12=getgama(pf3,N2);
R12=2*r12;
for i=1:61
        gama10=R10/(x(i)/N2+1);
        pd10(i)=Qchipr2(N2,0,gama10,1e-5);
        gama11=R11/(x(i)/N2+1);
        pd11(i)=Qchipr2(N2,0,gama11,1e-5);
        gama12=R12/(x(i)/N2+1);
```

```
    pd12(i)=Qchipr2(N2,0,gama12,1e-5);
end
figure(5)
plot(ENR,pd10,'b')
hold on
plot(ENR,pd11,'y')
plot(ENR,pd12,'r')
grid;
title('PD (linear) vs. ENR (dB) for PF = 10-1, 10-2, and 10-3 with Energy Detector and
N = 16.');
legend('pf=10^-1','pf=10^-2','pf=10^-3');
```

getgama.m

```
function y=getgama(PFA,N)
    x1=1;
    x2=0;
    while(abs(x1-x2)>1e-9)
        x2=x1;
        x1=iteration(PFA,x1,N);
    end
    y=x1;
end
```

```
function y=iteration(PFA,r,N)
    s=0;
    for i=1:(N/2)-1
        s=s+(r.^i)/factorial(i);
    end
    y=-log(PFA)+log(1+s);
end
```