EC5.406 Signal Detection & Estimation Theory
Assignment 4

Niharika Vadlamudi
2018122008

December 8th 2021

1 Problem I

Design a Newman Pearson detector to detect a signal $s[n] = A\cos(2\pi f_o n)$ in the presence of WGN noise of variance σ^2. Provide a MATLAB code for RoC plots verifying the analytical performance with simulations for various values of $\mathrm{SNR} = \frac{A^2}{\sigma^2}$.

This is a problem of Binary Hypothesis testing where we solve it using NP detector to maximise the probability of detection . To begin with , we start off with few assumptions :

- f is set to 1 Hz , $A = 1$ & N = 10000 points .
- σ^2 on dB scale value is fixed to -50db , SNR will be ranging from -20 to 20 db scale - [-20,-10,10,20] , A is computed accordingly .
- Here P_{fa} is increased from 0.0 to 1.0 , in steps of 0.1.

We must perform the Likelihood Ratio Test (LRT) to compute $L(x)$, and use the information of P_{fa} to compute γ. Let's state the hypothesis - H_0 and H_1 first :

$$H_0\text{: } x(n)\text{=}w(n)$$
$$H_1\text{: } x(n)\text{=}s(n) + w(n)$$

From prior derived results we know that , for cases like these , the LRT gives :

$$T(\mathrm{x}) = \sum_{n=0}^{N-1} x[n]s[n] > \gamma'$$

$$T \sim \begin{cases} \mathcal{N}\left(0, \sigma^2\mathcal{E}\right) & \text{under } \mathcal{H}_0 \\ \mathcal{N}\left(\mathcal{E}, \sigma^2\mathcal{E}\right) & \text{under } \mathcal{H}_1 \end{cases} \quad \text{where } \mathcal{E} \text{ is the energy of the signal } s(n).$$

Now , on computing the value of P_{fa} and P_D theoretically we get :

$$\begin{aligned} P_{FA} &= \Pr\left\{T > \gamma'; \mathcal{H}_0\right\} \\ &= Q\left(\frac{\gamma'}{\sqrt{\sigma^2\mathcal{E}}}\right) \\ P_D &= \Pr\left\{T > \gamma'; \mathcal{H}_1\right\} \\ &= Q\left(\frac{\gamma' - \mathcal{E}}{\sqrt{\sigma^2\mathcal{E}}}\right) \end{aligned}$$

where $Q(x)$ is the complimentary CDF , of $\phi(x)$, i.e the unit normal distribution .

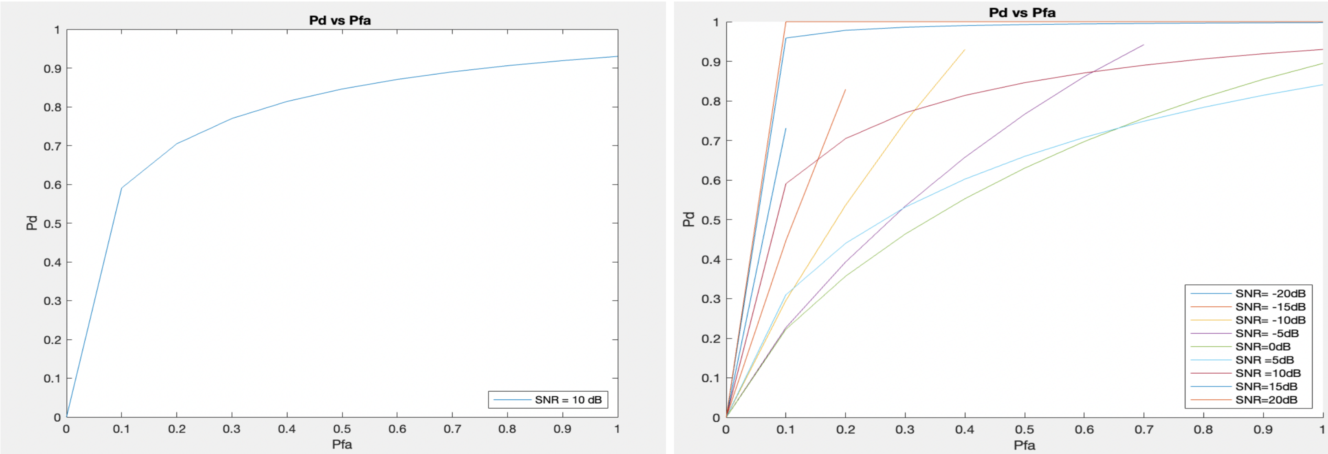
1.1 Source Code

A brief description about the code lines :

- Lines 1-10 Definition of necessary global variables & signal definition.
- Lines 14-31 Plotting of Pd vs Pfe for multiple values of SNR.
- Lines 34-39 Signal & Noise vector function definition .
- Lines 41 - 51 Function for computing P_d given noise variance & signal energy (e).

```
1 N=10000;
2 A=1;
3 f = 1/sqrt(2);
4 SNR_db_list=-20:5:20;
5 PFE_list=0:0.1:1;
6 
7 % Signal formation
8 time=1:1:N;
9 sig=A*cos(2*pi*f*time);
10 es=(1/N)*mtimes(sig,transpose(sig));
11 
12 %% Plotting- Computing for multiple values of SNR figure
13 % plot(SNR_list,Pe_list)
14 title('Pd vs Pfa ')
15 hold on
16 legendList=[];
17 i=1;
18 for snr_val=SNR_db_list
19     Pd=detectionProbability(es,snr_val,PFE_list);
20     plot(PFE_list,Pd)
21     i=i+1;
22 end
23 disp(legendList)
24 hold off
25 xlabel('Pfa')
26 ylabel('Pd')
27 xlim([0 1])
28 ylim([0 1])
29 %lgn=legend(legendList);
30 lgn=legend('SNR= -20dB','SNR= -15dB','SNR= -10dB','SNR= -5dB','SNR=0dB','SNR =5dB','SNR =10dB','
    SNR=15dB','SNR=20dB');
31 lgn.Location='southeast';
32 
33 %% Functions Defination
34 function s=signal(A,f,n)
35     s=vpa(A*cos(2*pi*f*n));
36 end
37 function w = noise(sigma_val)
38     w=normrnd(0,sigma_val);
39 end
40 %% Pd and Pfe calculation
41 function Pd=detectionProbability(energy,snr_db,PFE_list)
42     sigma=vpa(1/db2pow(snr_db));
43     term = sqrt(energy*sigma);
44     Pd=[];
45     i=1;
46     for pfe=PFE_list
47         gamma=qfuncinv(term*pfe);
48         Pd(i)=qfunc(gamma-energy/term);
49         i=i+1;
50     end
51 end
```

1.2 Qualitative Results



(a) Single Fixed SNR = 20 dB

(b) Multiple SNR Values

Figure 1: Pd vs SNR

1.3 Conclusion

From the above plots , we can conclude that P_d and $P_f a$ are log related , rise in P_d gives increases $P_f a$ as well . The role of SNR is seen in the (b) plot , where increase in SNR gives in better detection probability , the SNR values above 0dB are almost flat at $P_d = 1.0$, irrespective of P_{fa} variation.

2 Problem II

Consider 8-array PSK signal

$$\mathbf{s}_k = [A\cos((2k-1)\pi/8) \quad A\sin((2k-1)\pi/8)]^T, \text{ for } k = 1, 2, \ldots, 8$$

is transmitted over the AWGN channel such that the received signal is

$$\mathbf{x} = \mathbf{s}_k + \mathbf{w}$$

where $\mathbf{w} \sim \mathcal{N}\left(\mathbf{0}, \sigma^2\mathbf{I}\right)$. Assuming equal prior probabilities, design a detector to detect the transmitted signal such that the probability of error P_e is minimum. Provide a Matlab code for P_e vs. SNR (in dB) plot verifying the analytical performance using simulations.

This is a Multiple Hypothesis problem, which can be solved via Newman Person approach. Here , the goal is to minimise P_e , and we assume that the prior probabilities are equal , i.e $P_{Hi} = 1/8$. Now, given multiple signals , we complete T_k for each kth signal .

If we now choose to transmit one of M signals $\{s_0[n], s_1[n], \ldots, s_{M-1}[n]\}$ with equal prior probabilities, then as before we should choose $\mathcal{H}_i$ for which $p(\mathbf{x} \mid \mathcal{H}_i)$ is maximum. The optimal receiver is again a minimum distance receiver and thus we choose $\mathcal{H}_k$ if

$$T_k(\mathrm{x}) = \sum_{n=0}^{N-1} x[n]s_k[n] - \frac{1}{2}\mathcal{E}_k$$

is the maximum statistic of $\{T_0(\mathrm{x}), T_1(\mathrm{x}), \ldots, T_{M-1}(\mathrm{x})\}$, where $\mathcal{E}_k$ is the energy of the kth signal . Now, we can observe that $\mathcal{E}_i = \mathcal{E}_j \ \forall \ i{\neq}j$. Now , to compute the P_e , we compute via P_c , which is given by :

$$\begin{aligned} P_c &= \sum_{i=0}^{M} P\left(\mathcal{H}_i \mid \mathcal{H}_i\right) P\left(\mathcal{H}_i\right) \\ &= \frac{1}{M} \sum_{i=0}^{M} P\left(\mathcal{H}_i \mid \mathcal{H}_i\right) \end{aligned}$$

Now , we simply apply the following relation :

$$P_e = 1 - P_c$$

Now , to solve the question via simulations , we make the following assumptions :

- A is set to 1 & f = 1 as well .
- σ^2 on computed from corresponding SNR value . SNR will range from -20 to 20 db scale [-20.0..20] .

2.1 Detector Source Code

A brief description about the code lines :

- Lines 1 - 16 Defining various global variables & testing signal x.
- Lines 17 - 42 Main detector function code - includes s_k definitions , and computing test statistics for every kth signal.
- Lines 44 - 58 Function defination for Test Statistics $T_k(x)$, signal- s_k and noise - w.

```
1 %% Assignment 2 Multiple Hypothesis Testing.
2 A=1;
3 f=1;
4 sigma_val=0.0001;
5
6 % Testing signal , close to signal k=2 + added noise .
7 sm=[cos(2*(2*4-1)*pi/8) sin(2*(2*4-1)*pi/8)];
8 L=size(sm);
9 w=normrnd(0,0.5,L);
10 x_test=sm+w;
11
12 %Result
13 res=detector(x_test,sigma_val);
14 fprintf("Input Signal(x) matches Hypothes H_%d ; i.e S_%d\n",res,res)
15
16 %% Function Definations
17 function index = detector(x,sigma_val)
18     A=1;
19     s1=h_s(1,1,sigma_val);
20     s2=h_s(1,2,sigma_val);
21     s3=h_s(1,3,sigma_val);
22     s4=h_s(1,4,sigma_val);
23     s5=h_s(1,5,sigma_val);
24     s6=h_s(1,6,sigma_val);
25     s7=h_s(1,7,sigma_val);
26     s8=h_s(1,8,sigma_val);
27
28     % Check if the si signal's size is equal to x .
29     if(length(s1)~=length(x))
30         return;
31     end
32     s=[s1;s2;s3;s4;s5;s6;s7;s8];
33     L=length(s);
34     scores=[];
35     for i=1:1:8
36         s_i=s(i,:);
37         scores(i)=vpa(test_statistics(x,s_i));
38     end
39     % Finding maximum case .
40     [score,ind]=max(scores);
41     index=ind;
42 end
43
44 function h=h_s(A,k,sigma_val)
45     h=signal(A,k)+noise(sigma_val);
46 end
47
48 function s=signal(A,k)
49     s=[A*cos(2*(2*k-1)*pi/8) A*sin(2*(2*k-1)*pi/8)];
50 end
51
```

```
function w = noise(sigma_val)
    w=[normrnd(0,sigma_val) normrnd(0,sigma_val)];
end

function test=test_statistics(x,s)
    test=mtimes(x,transpose(s));
end
```

2.2 Simulation Pe - Plot Source Code

A brief description about the code :

- Lines 1 - 16 Defining all variables , here amplitude A = 1 , SNR is varied from -20 dB to +20 dB , in step size -of 2 dB.
- Lines 16 - 30 For every SNR value , we compute Pe value , using $p_{e}rror()$ function and perform plotting .
- Lines 32 - 48 Probability of error counting while performing multiple trails .
- Lines 61 - 104 All helper functions pertaining to detector , signal noise vectors .

```
1 %% Assignment 2 Multiple Hypothesis Testing.
2 A=1;
3 f=1;
4 M=8;
5 SNR_db=(-20:2:5);
6 SNR_list=vpa(db2pow(SNR_db));
7 sigma_list=reciprocal(SNR_list);
8 k_list=1:1:M;
9
10 % Iterating through sigma values .
11 xaxis=[];
12 yaxis=[];
13 i=1;
14 N=1000;
15 for sig=SNR_db
16     sig_val=1./db2pow(sig);
17     xaxis(i)=sig;
18     yaxis(i)=peCalculation(N,sig_val);
19     i=i+1;
20 end
21
22 %% Plotting
23 figure
24 semilogy(xaxis,yaxis)
25 % plot(xaxis,yaxis)
26 title('Pe vs SNR ')
27 xlabel('ENR - 10log(1/sigma^2))')
28 ylabel('Pe')
29
30 %% Function Definations
31 function pe = peCalculation(N,sigma_val)
32     ntimes=1:1:N;
33     countError=0;
34     countTrue=0;
35     for n=ntimes
36         [gd_n,pred_n]=randGeneratorSignal(1,sigma_val);
37         if(gd_n~=pred_n)
38             countError=countError+1;
39         else
40             countTrue=countTrue+1;
41         end
42     end
43     accuracy=countTrue/N;
44     pe=1-accuracy;
45     fprintf('Error is : %f \n',pe)
46
47 end
48
49 function [gd,pred]=randGeneratorSignal(A,sigma_val)
50     M=8;
```

```
    r = randi([1,M],1);
    s_r=[A*cos(2*(2*r-1)*pi/M) A*sin(2*(2*r-1)*pi/M)];
    L=size(s_r);
    w=normrnd(0,1,L);
    x_r=s_r+w;
    % Assignment of values
    gd=r;
    pred=detector(x_r,sigma_val);
end
function index = detector(x,sigma_val)
    A=1;
    s1=h_s(1,1,sigma_val);
    s2=h_s(1,2,sigma_val);
    s3=h_s(1,3,sigma_val);
    s4=h_s(1,4,sigma_val);
    s5=h_s(1,5,sigma_val);
    s6=h_s(1,6,sigma_val);
    s7=h_s(1,7,sigma_val);
    s8=h_s(1,8,sigma_val);

    % Check if the si signal's size is equal to x .
    if(length(s1)~=length(x))
        return;
    end
    s=[s1;s2;s3;s4;s5;s6;s7;s8];
    L=length(s);
    scores=[];
    for i=1:1:8
        s_i=s(i,:);
        scores(i)=vpa(test_statistics(x,s_i));
    end
    % Finding maximum case .
    [score,ind]=max(scores);
    index=ind;
end
function h_s=h_s(A,k,sigma_val)
    h_s=signal(A,k)+noise(sigma_val);
end

function s=signal(A,k)
    s=[A*cos(2*(2*k-1)*pi/8) A*sin(2*(2*k-1)*pi/8)];
end
function w = noise(sigma_val)
    w=[normrnd(0,sigma_val) normrnd(0,sigma_val)];
end

function test=test_statistics(x,s)
    A=1;
    test=mtimes(x,transpose(s))-0.5^(A*A);
end
function rec=reciprocal(t)
    rec=vpa(1./t);
end
```

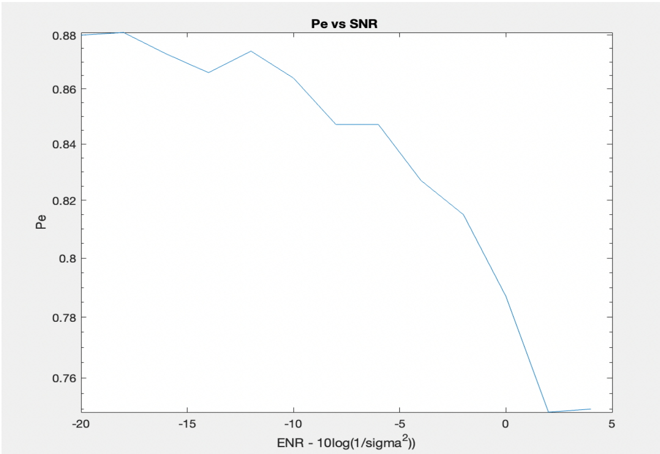
3 Theoretical Ps - Plot Source Code

A brief description about the code :

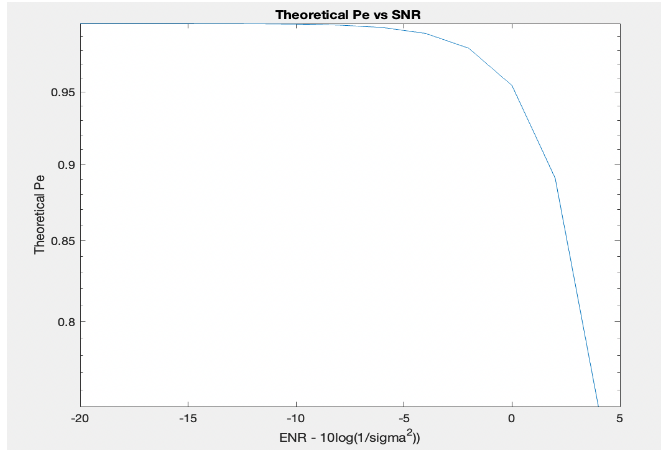
- Lines 1 - 12 Defining all variables , here amplitude A = 1 , SNR is varied from -20 dB to +20 dB .
- Lines 12 - 28 For every SNR value , we compute Pe value , using $pcIntegral()$ function and plotting SNR vs Pe , we use $10log(E/sigma^2)$ for SNR .
- Lines 30-47 We define the theoretical Pe function and bounds for integral fro every given k .
- Other helper functions such as reciprocal , signal , noise and test statistics .

```
%% Assignment 2 Multiple Hypothesis Testing.
%% Note : Single Observation
A=1;
f=1;
M=8;

SNR_db=(-20:2:5);
% Iterating through sigma values .
xaxis=[];
yaxis=[];
i=1;
N=1000;
for sig=SNR_db
    k=1;
    sig_val=1./db2pow(sig);
    xaxis(i)=sig;
    yaxis(i)=1-vpa(pcIntegral(sig_val,1));
    i=i+1;
end

%% Plotting
figure
semilogy(xaxis,yaxis)
% plot(xaxis,yaxis)
title('Theoretical Pe vs SNR ')
xlabel('ENR - 10log(1/sigma^2))')
ylabel('Theoretical Pe')

%% Function Definations
function pc=pcIntegral(sigma_val,k)
    hi=hypothesis(k,sigma_val);
    [xl0,xh0,yl0,yh0]=bounds(k);
    pc=abs(vpa(integral2(hi,xl0,xh0,yl0,yh0)));
end

function [xl,xh,yl,yh]=bounds(k)
    A=1;
    thetha=2*(2*k-1)*pi/8;
    thetha_l=thetha-pi/8;
    thetha_h=thetha+pi/8;
    % Cartesian Coordinate Bounds
    xl=A*cos(thetha_l);
    xh=A*cos(thetha_h);
    yl=A*sin(thetha_l);
    yh=A*sin(thetha_h);

end
function hi=hypothesis(k,sigma_val)
    A=1;
    s=signal(A,k);
    hi=@(x,y) normpdf(x,s(1),sigma_val).*normpdf(y,s(2),sigma_val);
end

function s=signal(A,k)
    s=[A*cos(2*(2*k-1)*pi/8);A*sin(2*(2*k-1)*pi/8)];
end
function w = noise(sigma_val)
    w=[normrnd(0,sigma_val) normrnd(0,sigma_val)];
end

function test=test_statistics(x,s)
    test=mtimes(x,transpose(s));
end
function rec=reciprocal(t)
    rec=1./t;
end
```

3.1 Qualitative Results



(a) Simulated Pe vs SNR



(b) Theoretical Pe vs SNR

Figure 2: Comparision between Theoretical vs Simulation Pe

In this experiment , we can observe the experimental P_e is following similar trend of theoretical P_e , with increase in SNR the detection process becomes more accurate . It also testifies that the test statistics chosen for this scenario is optimal in nature.

4 Refrences

- `StephenKay-DetectionTheoryTextbookVolumeII`
- `https://www.unilim.fr/pages_perso/vahid/notes/ber_awgn.pdf`
- `https://www.rfwireless-world.com/Terminology/8-PSK.html`
- `http://www.eecs.umich.edu/courses/eecs455/eecs455_F_2004/lect8.pdf`