MATLAB ASSIGNMENT

Submitted on February 16, 2023

Submitted to

Prof. ELIA Petros

Submitted by

Anni Ranta-Lassila
Logesh Dhanraj
Sai Shri Ram Saiveeraaswaamy
Godswill Onche

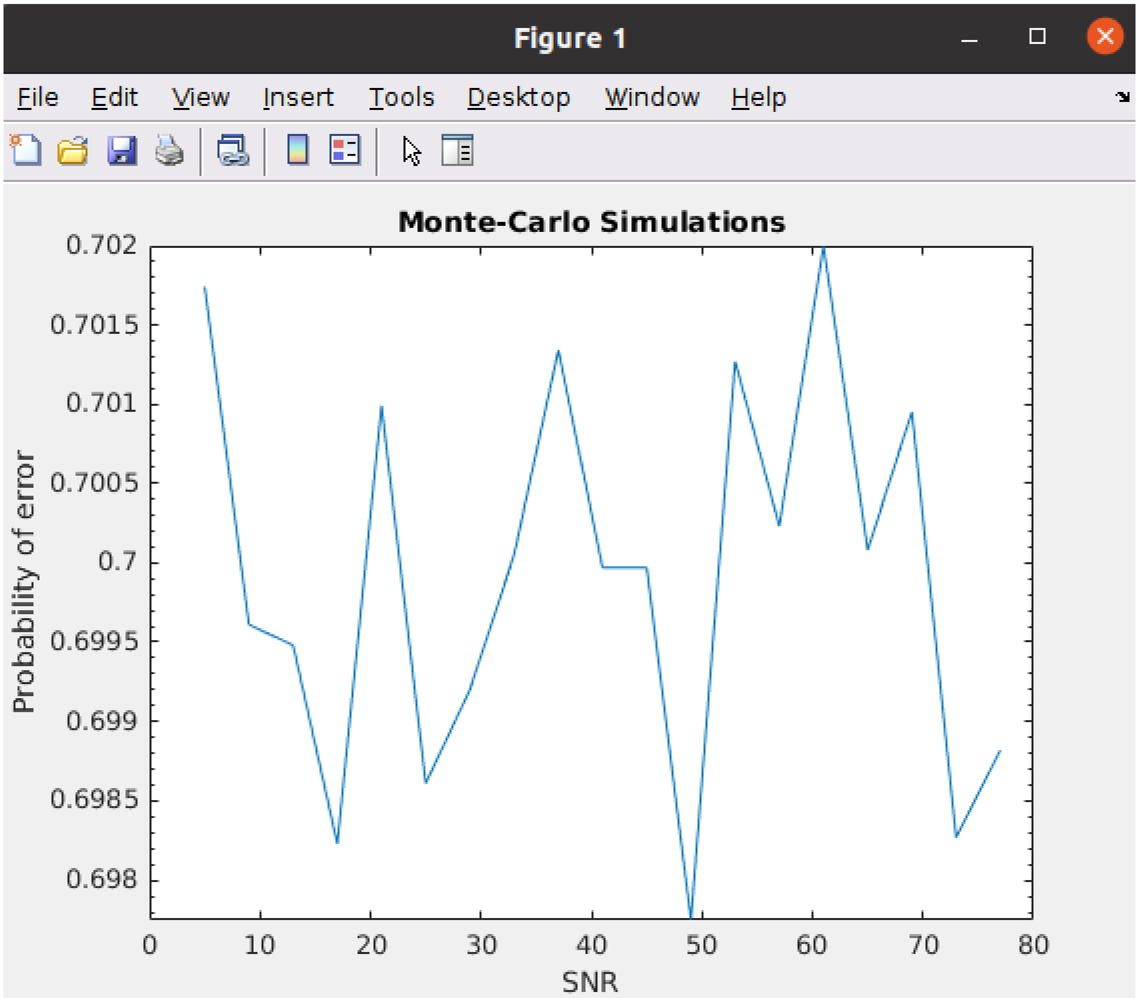
(Github repo: https://github.com/dhanrajlogesh/MATLAB_ASSIGNMENT_ATW)

PROBLEM 1a:

MATLAB CODE:

```
PAM = [-15,-13,-11,-9,-7,-5,-3,-1,1,3,5,7,9,11,13,15];
Total_error = 0;
samples = 100000;
counter = 0;
mu = 0;
sigma = 1;
for s = (5:4:80)
Total_error = 0;
for i = (0:samples)
x1 = PAM(randi(numel(PAM)));
x2 = PAM(randi(numel(PAM)));
x3 = PAM(randi(numel(PAM)));
h1 = (sigma.*randn(1)+mu)+ 1i*(sigma.*randn(1)+mu);
h2 = (sigma.*randn(1)+mu)+ 1i*(sigma.*randn(1)+mu);
h3 = (sigma.*randn(1)+mu)+ 1i*(sigma.*randn(1)+mu);
w1 = (sigma.*randn(1)+mu)+ 1i*(sigma.*randn(1)+mu);
w2 = (sigma.*randn(1)+mu)+ 1i*(sigma.*randn(1)+mu);
w3 = (sigma.*randn(1)+mu)+ 1i*(sigma.*randn(1)+mu);
x = [x1,0,0;0,x2,0;0,0,x3];
h = [h1,h2,h3];
w = [w1,w2,w3];
theta = sqrt(1/5 * db2mag(s));
y = theta*h*x + w;
minerr = intmax;
detectPAM = 0;
for j = (1:16)
dist = norm(y - theta*h*PAM(j));
if dist < minerr
minerr = dist;
detectedPAM = j;
end
end
if x ~= PAM(detectedPAM)
Total_error = Total_error +1;
end
end
counter = counter +1;
Perror(counter) = Total_error / samples;
end
semilogy(5:4:80, Perror);
title('Monte-Carlo Simulations')
xlabel('SNR')
ylabel('Probability of error')
```

MATLAB OUTPUT:



PROBLEM 1b:

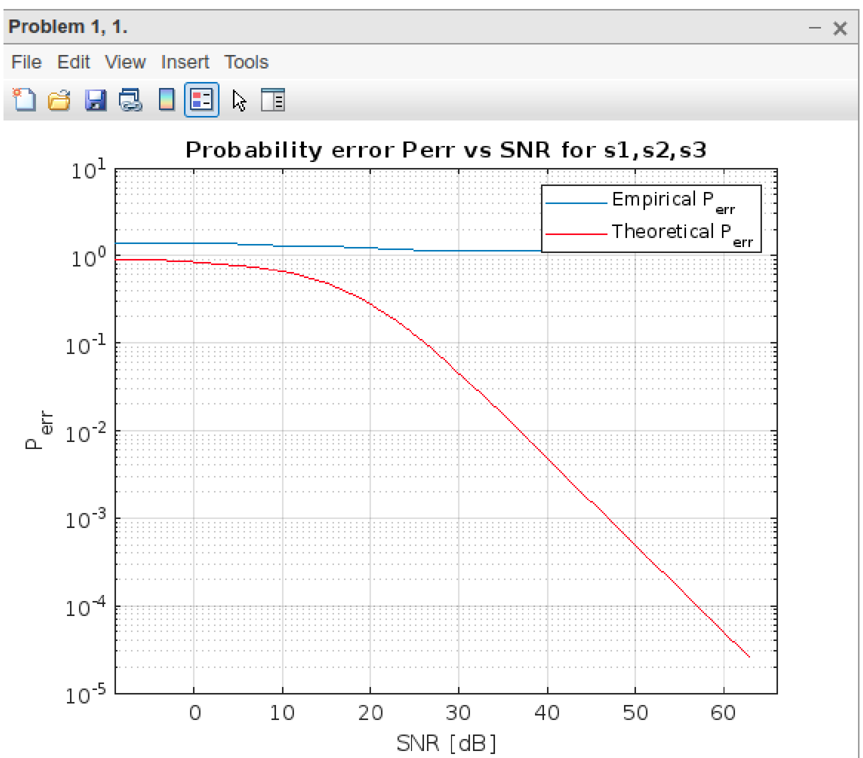
MATLAB CODE:

(s1,s2,s3 using QAM constellation)

```
sigma_w2 = 6;
Mont = 1e6;
SNR_max = 66;
SNR_start = -9;
SNR_step = 4;
Error_sum = zeros(1,length(SNR_start:SNR_step:SNR_max));
for i = 0:15
  const(i+1) = qammod(i,16);
end
for SNR = SNR_start:SNR_step:SNR_max
  P = 10^(SNR/10)*sigma_w2;
  % theta = 1/sqrt(10)*sqrt(P);
  theta = 1/sqrt((16^2-1)/3)*sqrt(P);
  for k = 1:Mont
      H = (randn(1,3)+1i*randn(1,3))/sqrt(2);
      s1 = const(randperm(16,1));
      s2 = const(randperm(16,1));
      s3 = const(randperm(16,1));
      xtr = theta*[s1,0,0;0,s2,0;0,0,s3];
      n = sqrt(sigma_w2)*((randn(1,3)+1i*randn(1,3))/sqrt(2));
      y = H*xtr+n;
      x1_hat = (y(1)/H(1,1))/theta;
      x2_hat = (y(2)/H(1,2))/theta;
      x3_hat = (y(3)/H(1,3))/theta;
      diff1 = abs(real(x1_hat-const(1)));
      diff2 = abs(real(x2_hat-const(1)));
      diff3 = abs(real(x3_hat-const(1)));
      idx1 = 1;
      idx2 = 1;
      idx3 = 1;
      for i = 2:length(const)
          if abs(real(x1_hat-const(i)))<diff1
              diff1 = abs(real(x1_hat-const(i)));
              idx1 = i;
          end
          if abs(real(x2_hat-const(i)))<diff2
              diff2 = abs(real(x2_hat-const(i)));
              idx2 = i;
          end
          if abs(real(x3_hat-const(i)))<diff3
              diff3 = abs(real(x3_hat-const(i)));
              idx3 = i;
```

```
            end
        end
        x1_hat = const(idx1);
        x2_hat = const(idx2);
        x3_hat = const(idx3);
        Error_sum((SNR-SNR_start)/SNR_step+1) = 
Error_sum((SNR-SNR_start)/SNR_step+1) + ((x1_hat~=s1) + (x2_hat~=s2) + 
(x3_hat~=s3));
  end
end
P_err = Error_sum./(2*Mont);
figure('NumberTitle','off','Name','Problem 1, 1.')
semilogy(SNR_start:SNR_step:SNR_max,P_err)
hold on
[ber,ser] = 
berfading((SNR_start-10*log(2):SNR_step:SNR_max-10*log(2)),'pam',16,1);
semilogy(SNR_start:SNR_step:SNR_max,ser,'r')
legend('Empirical P_{err}','Theoretical P_{err}')
xlabel('SNR [dB]')
ylabel('P_{err}')
title('Probability error Perr vs SNR for s1,s2,s3');
xlim([SNR_start SNR_max])
grid on
```

MATLAB OUTPUT:

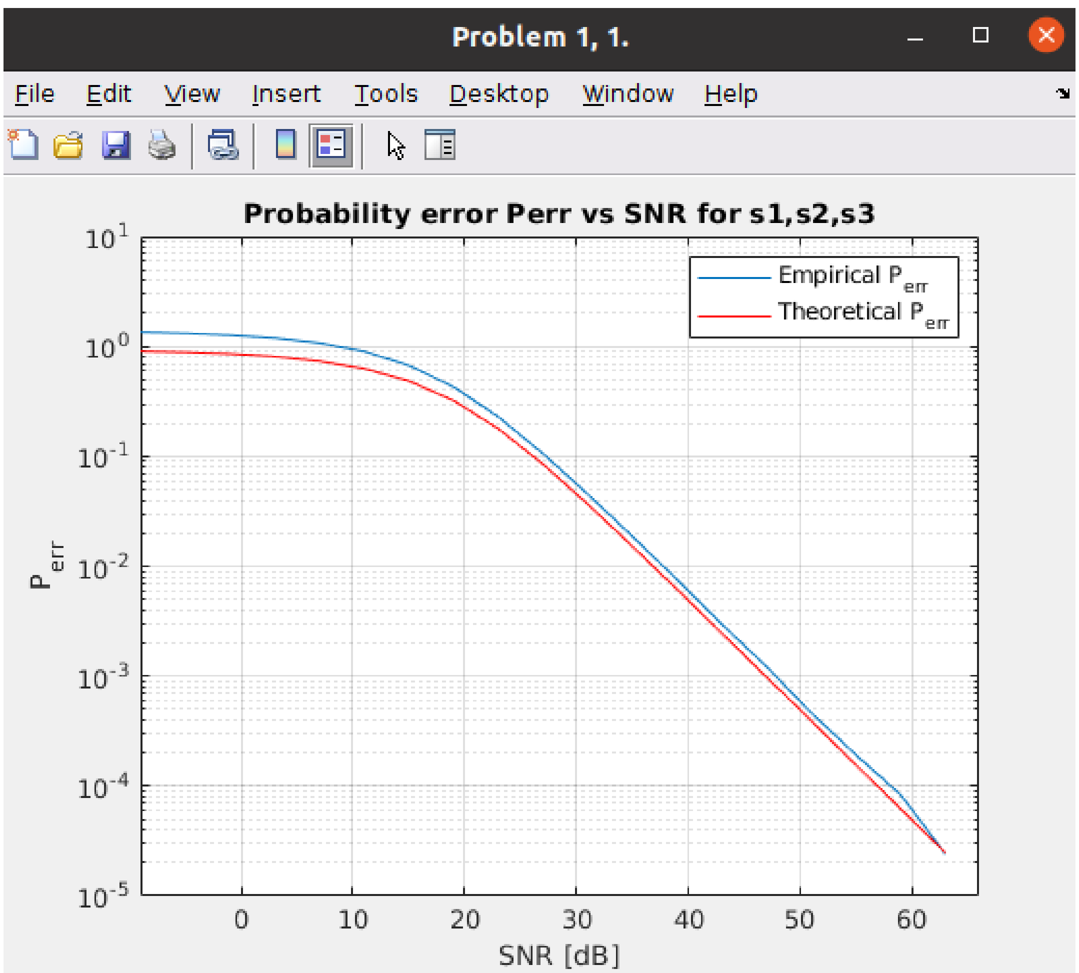


(s1,s2,s3 using PAM constellation)

```
sigma_w2 = 6;
Mont = 1e6;
SNR_max = 66;
SNR_start = -9;
SNR_step = 4;
Error_sum = zeros(1,length(SNR_start:SNR_step:SNR_max));
for i = 0:15
   const(i+1) = pammod(i,16);
end
for SNR = SNR_start:SNR_step:SNR_max
   P = 10^(SNR/10)*sigma_w2;
   theta = 1/sqrt((16^2-1)/3)*sqrt(P);
   for k = 1:Mont
       H = (randn(1,3)+1i*randn(1,3))/sqrt(2);
       %  16-PAM
       s1 = const(randperm(16,1));
       s2 = const(randperm(16,1));
       s3 = const(randperm(16,1));
       xtr = theta*[s1,0,0;0,s2,0;0,0,s3];
       n = sqrt(sigma_w2)*((randn(1,3)+1i*randn(1,3))/sqrt(2));
       y = H*xtr+n;
       x1_hat = (y(1)/H(1,1))/theta;
       x2_hat = (y(2)/H(1,2))/theta;
       x3_hat = (y(3)/H(1,3))/theta;
       diff1 = abs(real(x1_hat-const(1)));
       diff2 = abs(real(x2_hat-const(1)));
       diff3 = abs(real(x3_hat-const(1)));
       idx1 = 1;
       idx2 = 1;
       idx3 = 1;
       for i = 2:length(const)
           if abs(real(x1_hat-const(i)))<diff1
               diff1 = abs(real(x1_hat-const(i)));
               idx1 = i;
           end
           if abs(real(x2_hat-const(i)))<diff2
               diff2 = abs(real(x2_hat-const(i)));
               idx2 = i;
           end
           if abs(real(x3_hat-const(i)))<diff3
               diff3 = abs(real(x3_hat-const(i)));
               idx3 = i;
           end
       end
       x1_hat = const(idx1);
       x2_hat = const(idx2);
       x3_hat = const(idx3);
```

```
        Error_sum((SNR-SNR_start)/SNR_step+1) =
Error_sum((SNR-SNR_start)/SNR_step+1) + ((x1_hat~=s1) + (x2_hat~=s2) +
(x3_hat~=s3));
   end
end
P_err = Error_sum./(2*Mont);
figure('NumberTitle','off','Name','Problem 1, 1.')
semilogy(SNR_start:SNR_step:SNR_max,P_err)
hold on
[ber,ser] =
berfading((SNR_start-10*log(2):SNR_step:SNR_max-10*log(2)),'pam',16,1);
semilogy(SNR_start:SNR_step:SNR_max,ser,'r')
legend('Empirical P_{err}','Theoretical P_{err}')
xlabel('SNR [dB]')
ylabel('P_{err}')
title('Probability error Perr vs SNR for s1,s2,s3');
xlim([SNR_start SNR_max])
grid on
```

MATLAB OUTPUT:



PROBLEM 2:

We want to simulate the 1 × 4 SIMO model y = hx + w (where h = [h1 h2 h3 h4]) to establish the probability of deep fade:

P(||h||2 < SNR^-1)

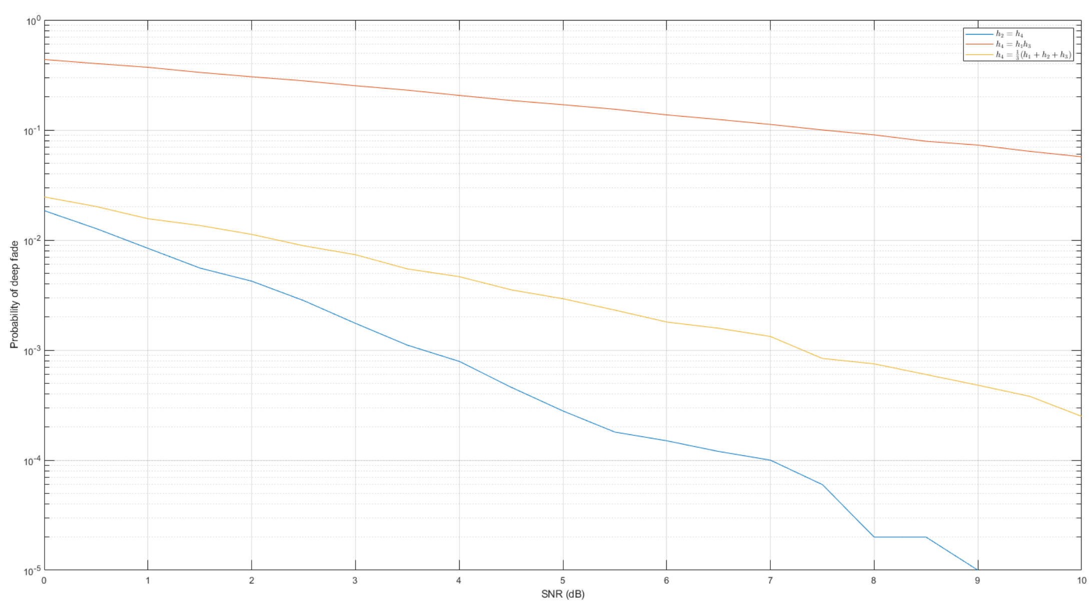
We generated h = [h 1 h 2] using the MATLAB function rand 100’000 times and we compared the resulted ||h|| with the SNR from 0 to 10 dB. We have done this when h i ~ CN (0, 1) i.i.d and we repeated the procedure in the case in which h2=h4 ;h4= h1xh3 where h 1 , h 3 ~ CN (0, 1) i.i.d and when h 4 = ⅓ (h 1 + h2+h 3) where h 1 , h 2, h3 ~ CN (0, 1).

MATLAB CODE:

```
close all
clear
samples = 1e5; %MonteCarlo
SNR = 0:0.5:10; %dB
P1=zeros(1, length(SNR));
P2=zeros(1, length(SNR));
P3=zeros(1, length(SNR));
for j=1:samples
   %case 1
   for SNRi = 1:length(SNR)
       h1=(randn(1,1)+randn(1,1)*sqrt(-1))/sqrt(2);
       h2=(randn(1,1)+randn(1,1)*sqrt(-1))/sqrt(2);
       h3=(randn(1,1)+randn(1,1)*sqrt(-1))/sqrt(2);
       h4=(randn(1,1)+randn(1,1)*sqrt(-1))/sqrt(2);

       h = [h1 h2 h3 h4];
       if (sum(abs(h).^2)*10^(SNR(SNRi)/10)<1)
           P1(SNRi)=P1(SNRi)+1;
       end
  end
   %case 2
   for SNRi = 1:length(SNR)
       h1=(randn(1,1)+randn(1,1)*sqrt(-1))/sqrt(2);
       h4=h1*(randn(1,1)+randn(1,1)*sqrt(-1))/sqrt(2);

       h = [h1 h4];
       if (sum(abs(h).^2)*10^(SNR(SNRi)/10)<1)
           P2(SNRi)=P2(SNRi)+1;
       end
   end
```

```
    %case 3
    for SNRi = 1:length(SNR)
        h1=(randn(1,1)+randn(1,1)*sqrt(-1))/sqrt(2)*sqrt(10);
        h4=(h1+(randn(1,1)+randn(1,1)+randn(1,1)*sqrt(-1))/sqrt(2)*sqrt(10))/3;

        h = [h1 h4];
        if (sum(abs(h).^2)*10^(SNR(SNRi)/10)<1)
            P3(SNRi)=P3(SNRi)+1;
        end
    end
end
P1=P1./samples;
P2=P2./samples;
P3=P3./samples;
figure('NumberTitle','off','Name','Problem 2, case 1,2,3');
semilogy(SNR, P1);
hold on
semilogy(SNR, P2);
hold on
semilogy(SNR, P3);
ylabel('Probability of deep fade')
xlabel('SNR (dB)')
grid on;
lgd=legend('$h_2=h_4$', '$h_4=h_1 h_3$', '$h_4 =\frac{1}{3}(h_1+h_2+h_3)$');
set(lgd,'Interpreter','latex');
```

MATLAB OUTPUT:

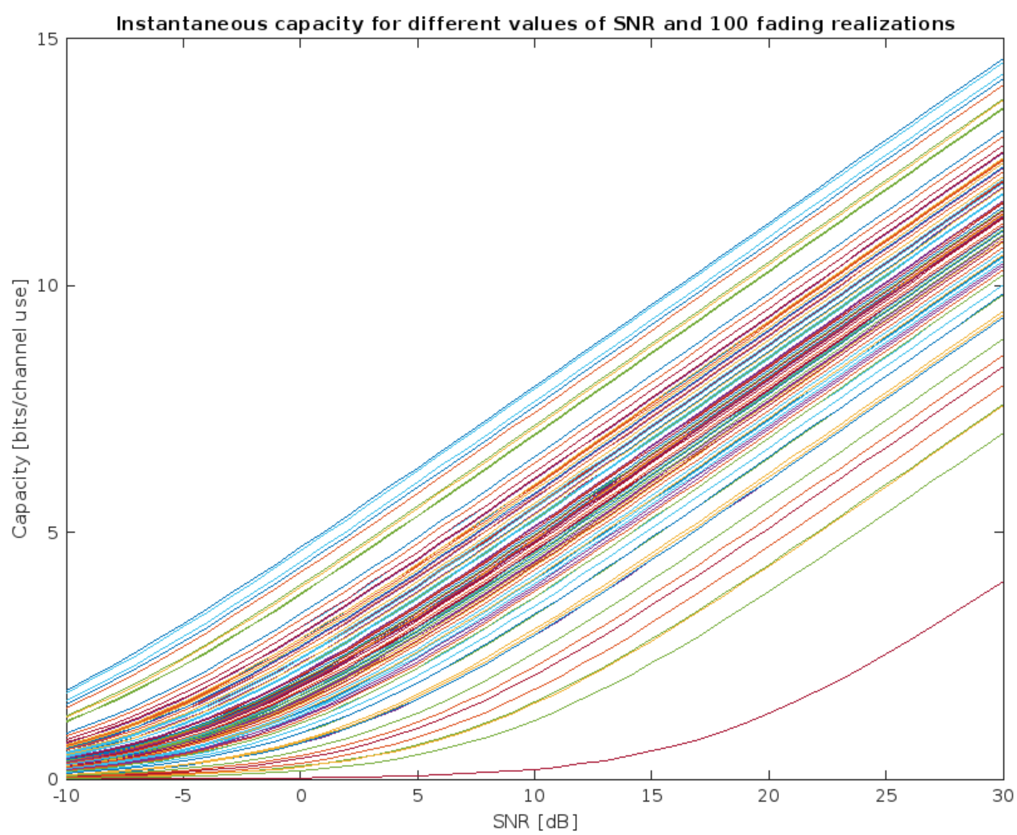


PROBLEM 3a:

MATLAB CODE:

```
clc
clear
nFading = 100;
% Define SNR in dB and in linear scale
snrdB = -10:1:30;
snr = 10.^(snrdB/10);
% Defining the fading coefficient with mean 0 and variance 1/2
h = (randn(nFading, 1) + 1i * randn(nFading, 1))*(sqrt(2));
% Calculating the instantaneous capacity from the formula
C = log2(1+abs(h).^2*snr);
% Plotting of C vs SNR
figure(1)
plot(snrdB, C);
title('Instantaneous capacity for different values of SNR and 100 fading
realizations');
xlabel('SNR [dB]');
ylabel('Capacity [bits/channel use]');
```

MATLAB OUTPUT:

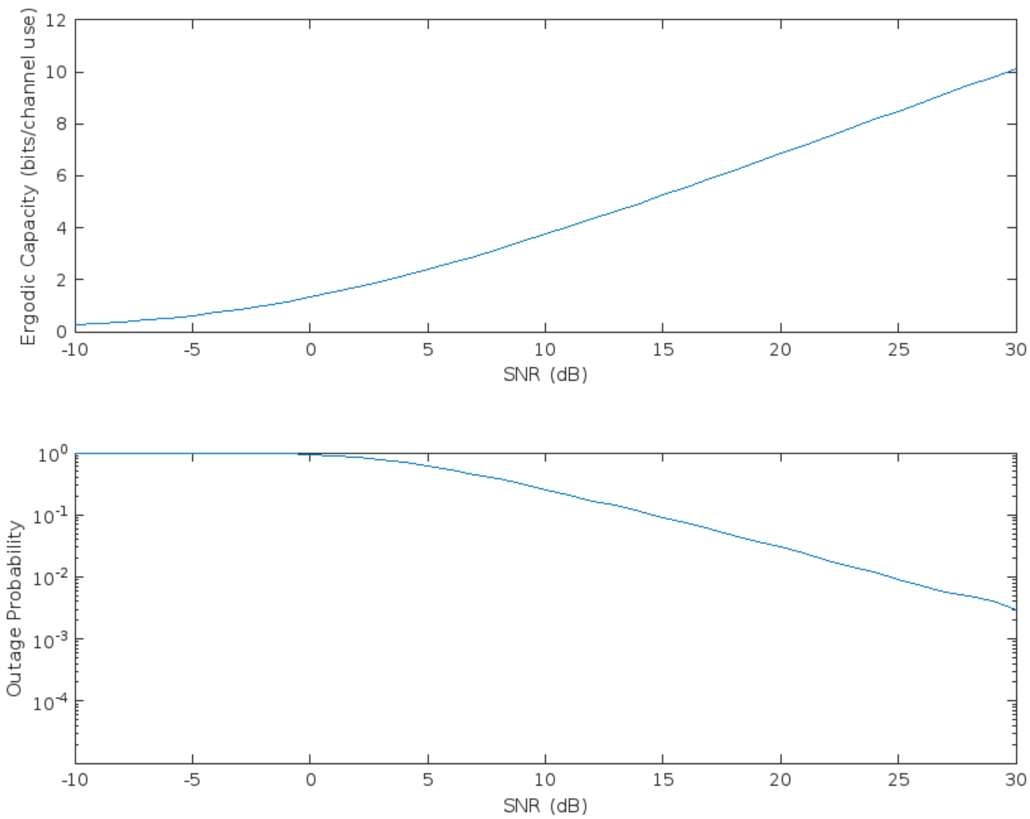


PROBLEM 3b:

MATLAB CODE:

```
num_fadings = 10000; % Number of fading realizations
SNR_dB_vec = -10:1:30; % Vector of SNR values in dB
SNR_vec = 10.^(SNR_dB_vec/10); % Vector of SNR values in linear scale
R = 2; % Target rate in bits/channel use
% Generate fading coefficients
h = sqrt(1/2)*(randn(num_fadings,1) + 1i*randn(num_fadings,1));
% Compute the ergodic capacity for each SNR value
C = zeros(length(SNR_vec),1);
for i = 1:length(SNR_vec)
   SNR = SNR_vec(i);
   C(i) = mean(log2(1 + SNR*R*abs(h).^2)); % Ergodic capacity
end
% Compute the outage probability for each SNR value
Pout = zeros(length(SNR_vec),1);
for i = 1:length(SNR_vec)
   SNR = SNR_vec(i);
   Pout(i) = mean(log2(1 + SNR*abs(h).^2) < R); % Outage probability
end
% Plot the results
figure;
subplot(2,1,1);
plot(SNR_dB_vec, C);
xlabel('SNR (dB)');
ylabel('Ergodic Capacity (bits/channel use)');
subplot(2,1,2);
semilogy(SNR_dB_vec, Pout);
xlabel('SNR (dB)');
ylabel('Outage Probability');
ylim([1e-5 1]);
```

MATLAB OUTPUT:



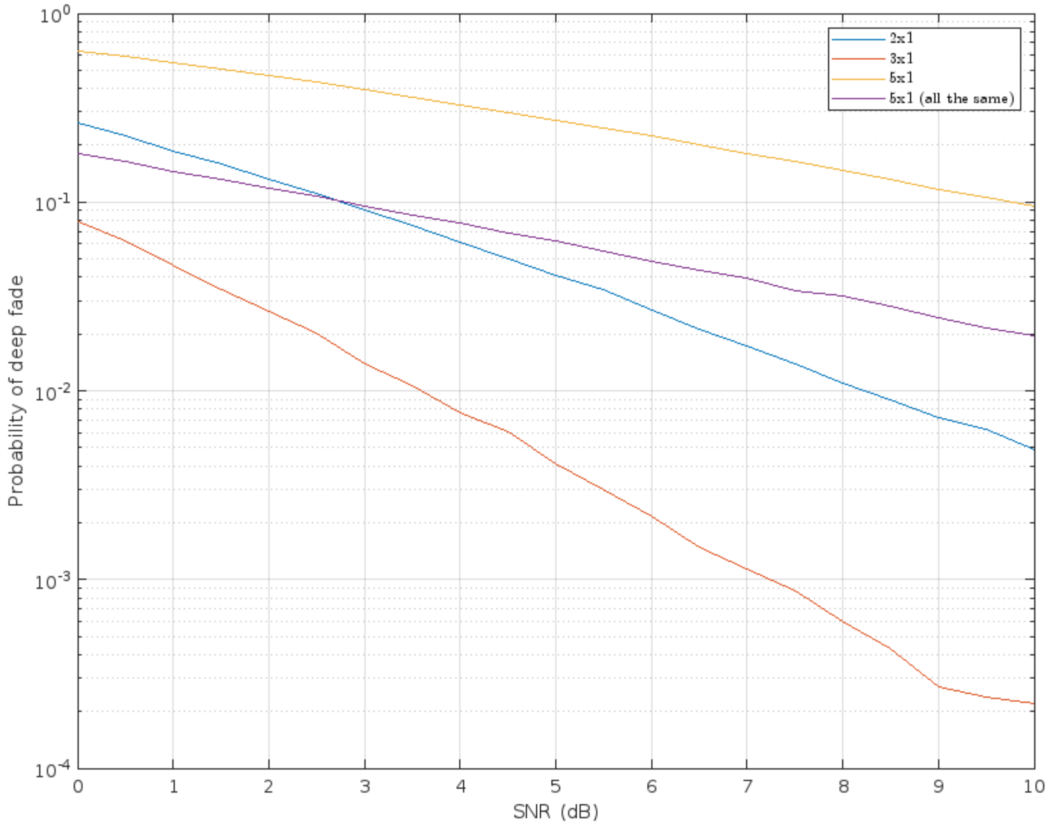
PROBLEM 4:

MATLAB CODE:

```
close all
clear
samples = 1e5; %MonteCarlo
SNR = 0:0.5:10; %dB
P1=zeros(1, length(SNR));
P2=zeros(1, length(SNR));
P3=zeros(1, length(SNR));
P4=zeros(1, length(SNR));
%P5=zeros(1, length(SNR));
for j=1:samples
   %case 1
   for SNRi = 1:length(SNR)
       h1=(randn(1,1)+randn(1,1)*sqrt(-1))/sqrt(2);
       h2=(randn(1,1)+randn(1,1)*sqrt(-1))/sqrt(2);

       h = [h1 h2].';
       if (sum(abs(h).^2)*10^(SNR(SNRi)/10)<1)
           P1(SNRi)=P1(SNRi)+1;
       end
  end
   %case 2
   for SNRi = 1:length(SNR)
       h1=(randn(1,1)+randn(1,1)*sqrt(-1))/sqrt(2);
       h2=(randn(1,1)+randn(1,1)*sqrt(-1))/sqrt(2);
       h3=(randn(1,1)+randn(1,1)*sqrt(-1))/sqrt(2);

       h = [h1 h2 h3].';
       if (sum(abs(h).^2)*10^(SNR(SNRi)/10)<1)
           P2(SNRi)=P2(SNRi)+1;
       end
   end
   %case 3
   for SNRi = 1:length(SNR)
       h = [];
       for loop=1:5
           h = (randn(1,1)+randn(1,1)*sqrt(-1))/sqrt(2);
       end
       if (sum(abs(h).^2)*10^(SNR(SNRi)/10)<1)
           P3(SNRi)=P3(SNRi)+1;
       end
   end
   %case 4
   for SNRi = 1:length(SNR)
       h1=(randn(1,1)+randn(1,1)*sqrt(-1))/sqrt(2);
       h = [];
```

```
        for loop=1:5
            h = [h h1];
        end
        if (sum(abs(h).^2)*10^(SNR(SNRi)/10)<1)
            P4(SNRi)=P4(SNRi)+1;
        end
    end
end
P1=P1./samples;
P2=P2./samples;
P3=P3./samples;
P4=P4./samples;
%P5=P5./samples;
figure();
semilogy(SNR, P1);
hold on
semilogy(SNR, P2);
hold on
semilogy(SNR, P3);
hold on
semilogy(SNR, P4);
hold on
ylabel('Probability of deep fade');
xlabel('SNR (dB)');
grid on;
lgd=legend('2x1', '3x1', '5x1', '5x1 (all the same)');
set(lgd,'Interpreter','latex');
```

MATLAB OUTPUT:



PROBLEM 5:

We created different experiments to check the validity of some assumptions.

- The far tail for the Gaussian random variable hr~ N (0, 1) is approximated by the exponential $e^{-x^2/2}$

As we observe from the plots below, when x is big the complementary cumulative distribution function is very near to the exponential and the difference between them is negligible.

MATLAB CODE:

```
x=0:0.00001:10;
expApprox=exp(-(x.^2)./2);
figure();
plot(x, qfunc(x));
hold on;
plot(x, expApprox);
hold on;
plot(x, abs(expApprox-qfunc(x)), 'k--');
grid on;
lgd=legend('$Q(\alpha)$', '$e^{-\alpha^2/2}$', '$|Q(\alpha) -
e^{-\alpha^2/2}|$' );
set(lgd,'Interpreter','latex');
xlabel('x');
smallerThan = 0:0.001:2;
results=zeros(1, length(smallerThan));
rep=10000;
for i=1:rep
   htest=(randn(1,1)+randn(1,1)*sqrt(-1));
   for j=1:length(smallerThan)
       if((abs(htest)^2)<smallerThan(j))
           results(j)=results(j)+1;
       end
   end

end

figure, plot(smallerThan, results./rep);
hold on;
plot(smallerThan, smallerThan);
hold on;
plot(smallerThan, abs(results./rep-smallerThan), 'k--');
grid on;
lgd=legend('$P(||h||^2<\epsilon)$', '$\epsilon$', '$|P(||h||^2<\epsilon) -
\epsilon|$' );
xlabel('\epsilon')
ylabel('f(\epsilon)')
set(lgd,'Interpreter','latex');
daspect([1 1 2]);
```

```
%3rd question
smallerThan = 0:0.001:100;
results=zeros(1, length(smallerThan));
rep=50000;
for i=1:rep
   htest=(10*randn(1,1)+10*randn(1,1)*sqrt(-1));
   for j=1:length(smallerThan)
       if((abs(htest)^2)<smallerThan(j))
           results(j)=results(j)+1;
       end
   end

end

figure;
plot(smallerThan, results./rep);
hold on
plot(smallerThan, smallerThan/250)
hold on
plot(smallerThan, abs(results./rep-smallerThan/250), 'k--');
grid on;
lgd=legend('$P(||h||^2<\epsilon)$' ,'$\epsilon/100$','$|P(||h||^2<\epsilon) -
\epsilon|$');
xlabel('\epsilon')
ylabel('f(\epsilon)')
set(lgd,'Interpreter','latex');
%daspect([1 1 2]);
%xlim([0 0.5])
%4th question
k=1:3;
rep=50000;
smallerThan = 0:0.01:2;
results=zeros(length(k), length(smallerThan));
figure;
for t=1:length(k)
   for i=1:rep
       hSqtest = chi2rnd(2*k(t));
       for j=1:length(smallerThan)
           if(hSqtest<smallerThan(j))
               results(t,j)=results(t,j)+1;
           end
       end
   end
   plot(smallerThan, results(t,:)./rep);
   hold on;
end
legend('k=1', 'k=2', 'k=3');
figure, plot(smallerThan, gammainc(smallerThan./2,k(1),'lower'));
hold on;
```

```
plot(smallerThan, smallerThan.^k(1))
hold on;
plot(smallerThan, abs(smallerThan.^k(1)-gammainc(smallerThan./2,k(1),'lower')),
'--k')
grid on;
lgd=legend('$P(||h||^2<\epsilon)$', '$\epsilon$', '$|P(||h||^2<\epsilon) -
\epsilon|$');
xlabel('\epsilon')
ylabel('f(\epsilon)')
set(lgd,'Interpreter','latex');
daspect([1 1 2]);
figure, plot(smallerThan, gammainc(smallerThan./2,k(2),'lower'));
axis([-inf inf 0 0.7]);
hold on;
plot(smallerThan, smallerThan.^k(2))
hold on;
plot(smallerThan, abs(smallerThan.^k(2)-gammainc(smallerThan./2,k(2),'lower')),
'--k')
grid on;
lgd=legend('$P(||h||^2<\epsilon)$', '$\epsilon^2$', '$|P(||h||^2<\epsilon) -
\epsilon^2|$' );
xlabel('\epsilon')
ylabel('f(\epsilon)')
set(lgd,'Interpreter','latex');
figure, plot(smallerThan, gammainc(smallerThan./2,k(3),'lower'));
axis([-inf inf 0 0.7]);
hold on;
plot(smallerThan, smallerThan.^k(3))
hold on;
plot(smallerThan, abs(smallerThan.^k(3)-gammainc(smallerThan./2,k(3),'lower')),
'--k')
grid on;
lgd=legend('$P(||h||^2<\epsilon)$', '$\epsilon^3$', '$|P(||h||^2<\epsilon) -
\epsilon^3|$'  );
xlabel('\epsilon');
ylabel('f(\epsilon)');
set(lgd,'Interpreter','latex');
```

MATLAB OUTPUT:

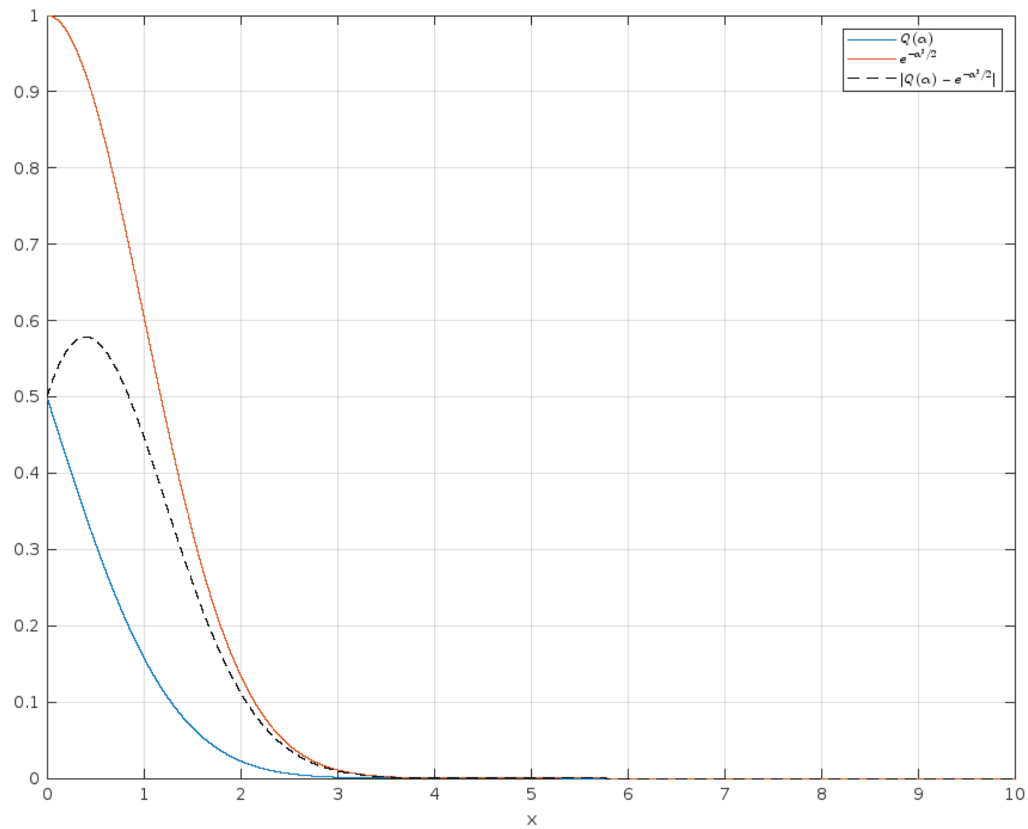


Fig1: Q-function with $e^{-x^2/2}$

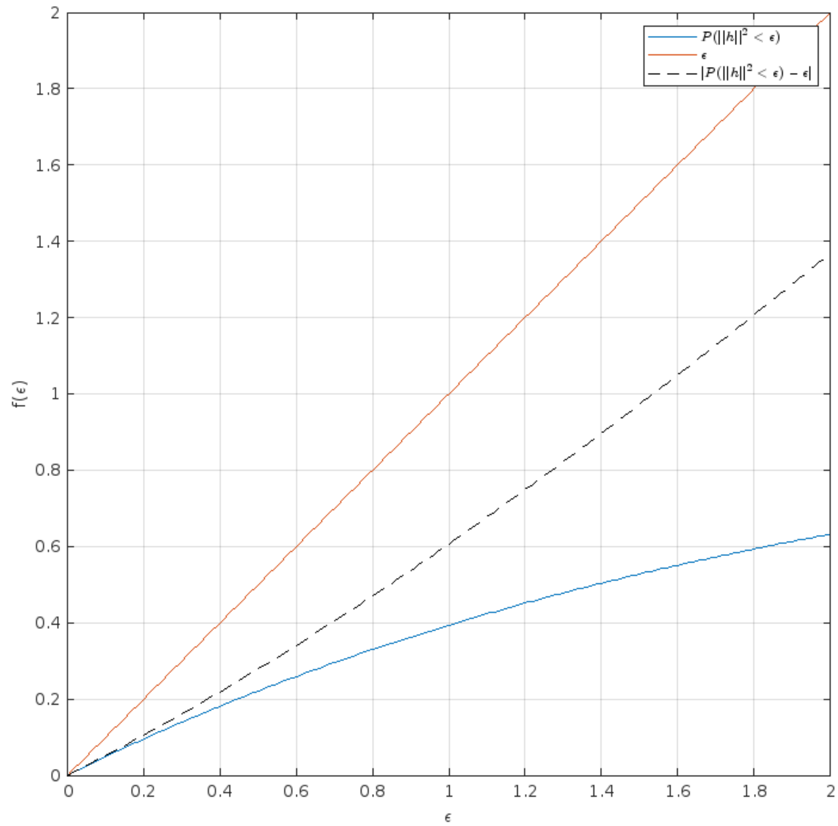


Fig 2: For k=1,Non-zero behavior of ϵ^k

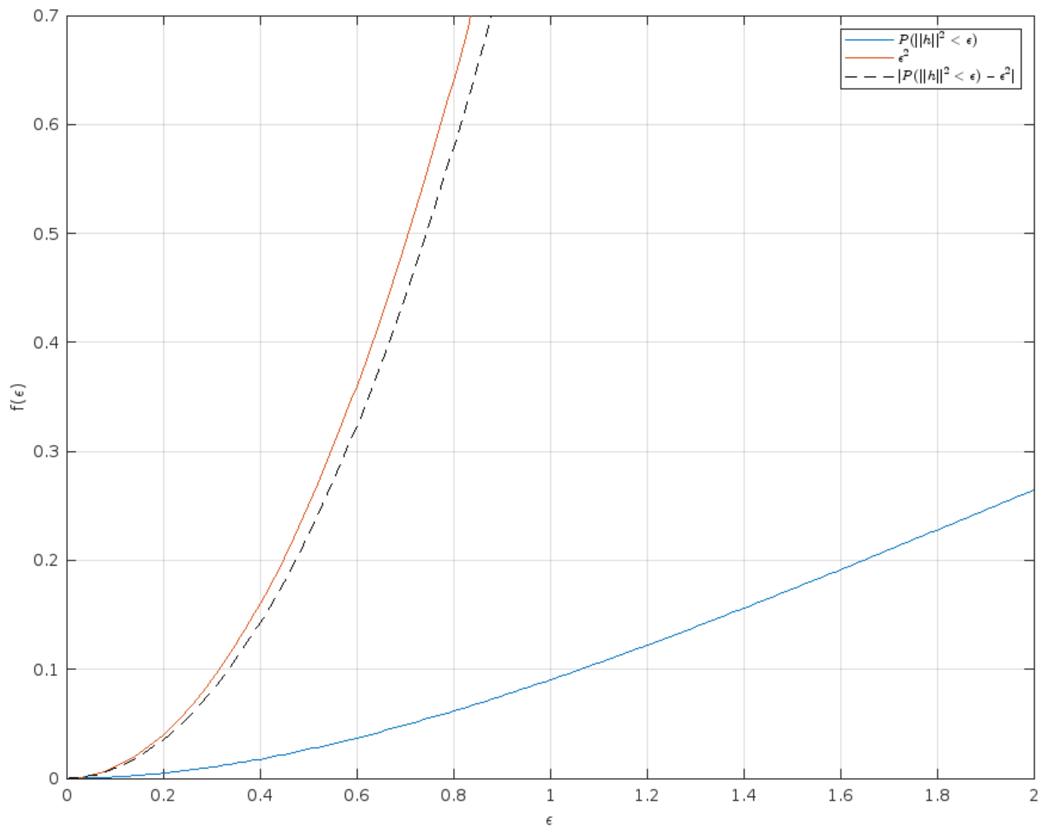


Fig 3: For k=2, Non-zero behavior of ϵ^k

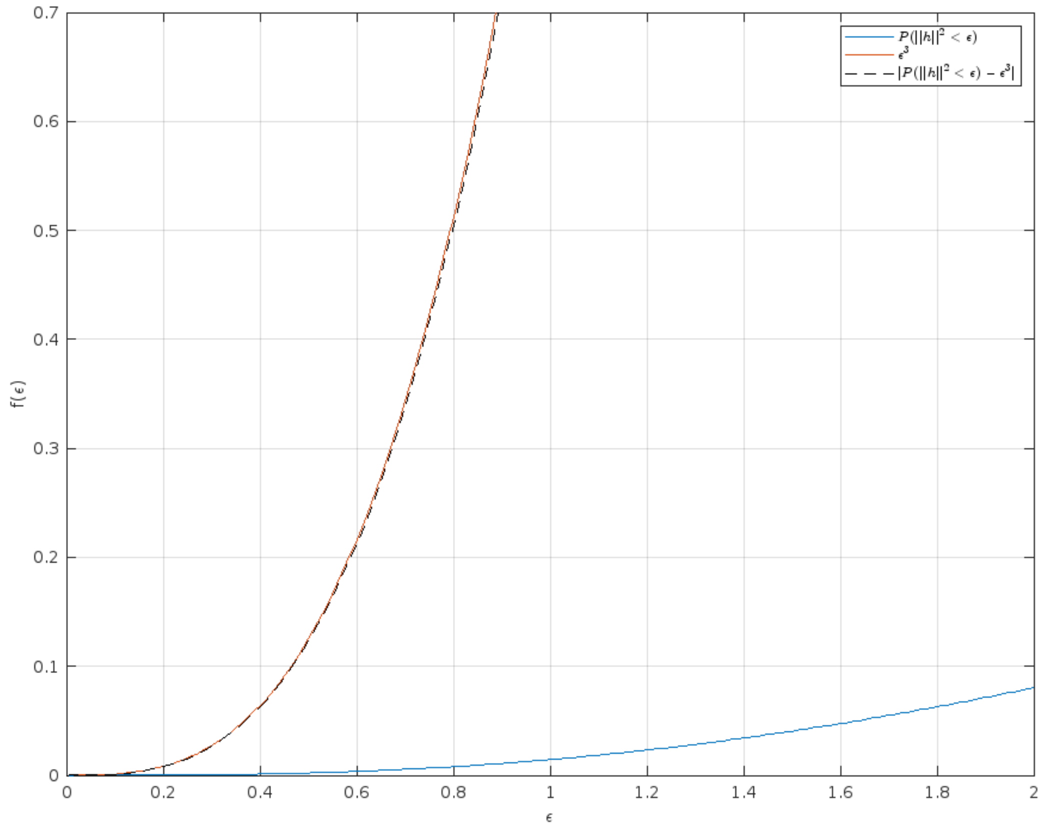


Fig4: For k=3, Non-zero behavior of ϵ^k

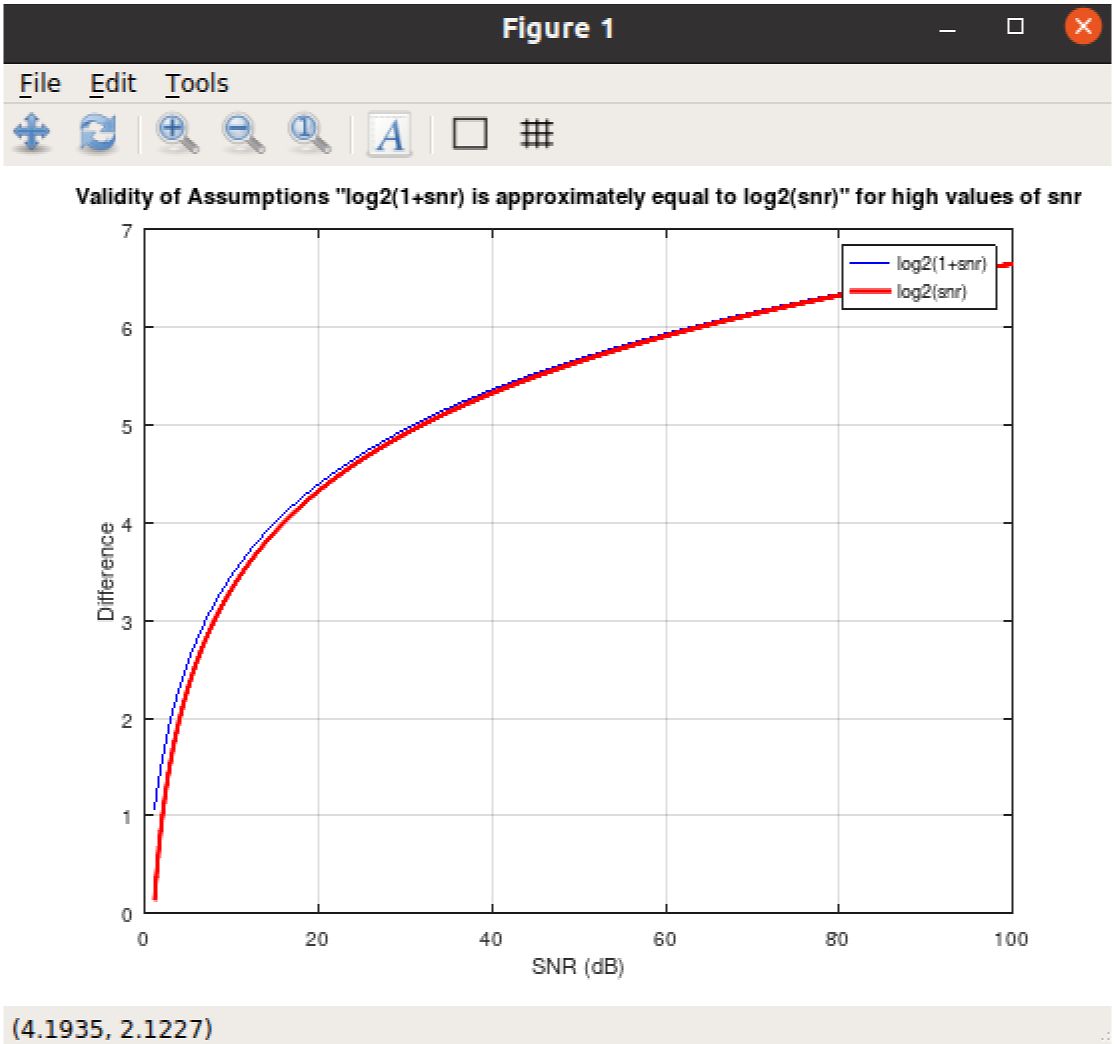
PROBLEM 6:

- log2(1 + SNR) ≈ log2(SNR) for high SNR
- log2(1 + SNR) ≈ SNR · log2(e) for low values of SNR

MATLAB CODE:

```
% case 1 : log2(1+SNR) is approximately equal to log2(SNR) for high SNR
clc;
snr = 0:0.1:100;
x1 = log2(1 + snr);
x2 = log2(snr);
% PLOT Function
figure(1);
plot(snr(snr>1), x1(snr>1), 'b',snr(snr>1), x2(snr>1), 'r', 'LineWidth', 2);
%%(snr>1) is chosen to keep the snr positive
grid on;
legend('log2(1+snr)', 'log2(snr)');
xlabel('SNR (dB)');
ylabel('Difference');
title('Validity of Assumptions "log2(1+snr) is approximately equal to
log2(snr)" for high values of snr');
% case2: log2(1+SNR) is approximately equal to snr*log2(e) for low values of
SNR
snr = 0:0.1:1;
x1 = log2(1 + snr);
x3 = snr * log2(exp(1));
% PLOT Function
figure(2);
plot(snr, x1, 'b', snr, x3, 'r', 'LineWidth', 2);
grid on;
legend('log2(1+snr)', 'snr*log2(e)');
xlabel('SNR(dB)');
ylabel('Difference');
title('Validity of Assumptions "log2(1+snr) is approximately equal to
SNR*log2(e)" for low values of snr');
```

MATLAB OUTPUT:



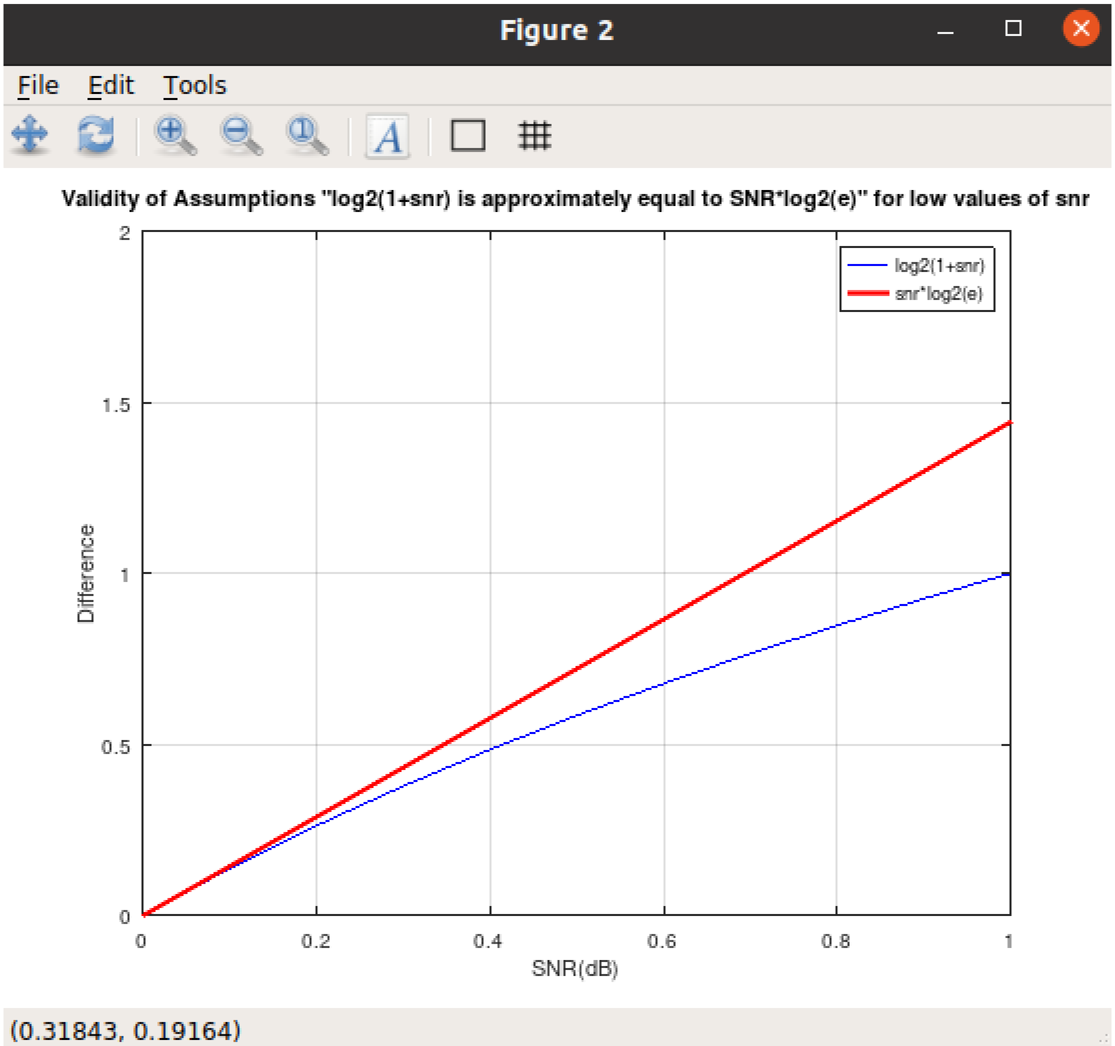


Figure 2
File
Edit
Tools
Validity of Assumptions "log2(1+snr) is approximately equal to SNR*log2(e)" for low values of snr
log2(1+snr)
snr*log2(e)
Difference
SNR(dB)
0
0.5
1
1.5
2
0
0.2
0.4
0.6
0.8
1
(0.31843, 0.19164)