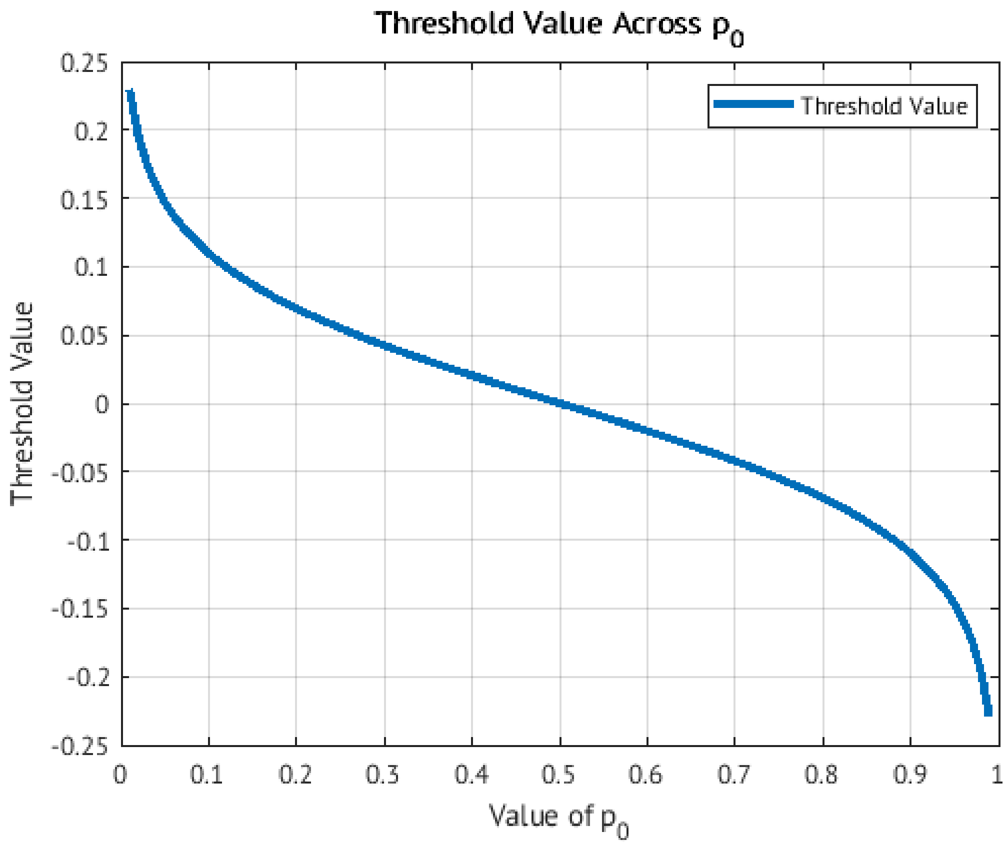
```
% S22 CMPE320 Project 4 BASK simulation
%
close all
clear

disp('CMPE320 Spring 2022 Project 4:  BASK');

% 2.1
p0=[0.01:.01:0.99]; % range of p0 given in 2.1
gamma_dB = 10; % given in 2.1
A=1; % given in 2.1
sigma2 = 10^(-gamma_dB/10); % convert dB to power, then to sigma^2, see 2.1
tauMAP = (sigma2 .* log((1 - p0) ./ p0)) ./ (2 * A); % put the functional
 expression for your tauMAP derived in 2.1
figure; % new figure
% p0 and tauMAP, per instructions in 2.1
%
% Professional labels, etc.
plot(p0, tauMAP, 'LineWidth', 3); % this is good
xlabel('Value of p_0');
ylabel('Threshold Value');
grid on;
legend('Threshold Value');
title('Threshold Value Across p_0');

%2.2

figure; %new figure
p0=0.8; % per 2.2
gamma_dB=10; % per 2.2
sigma2= 10^(-gamma_dB/10); % convert gamma_dB to sigma^2 as before
Ntrials = 500000; % set a large number of trials (500,000?)
A = 1; % per 2.1
B = rand(1,Ntrials) <= p0; % create random variable B based on p0, as in
 Project 2
M = 2 * (B - 0.5); % "modulate" by converting B per equation equation (1.1) in
 writeup
N =  sqrt(sigma2)*randn(1,Ntrials); % generate noise from N(0,sigma2) as in
 Project 2
R=M+N; % per signal model for Project 4
Rhist = histogram(R, 'Normalization', 'pdf'); % generate appropriately
 normalized histogram for R
r = [-2:.01:2]; % fine grid for plotting fRr
rGivenA = exp(-(r-A).^2/(2*sigma2))/sqrt(2*pi*sigma2);
rGivenNegA = exp(-(r+A).^2/(2*sigma2))/sqrt(2*pi*sigma2);
fRr = rGivenA * p0 + rGivenNegA * (1-p0); % expression for the fRr you
 derived.  This is very similar to Project 2, but explicitly includes p0
thresh = (sigma2 / (2 * A)) * log((1-p0) / p0);
hold on;
plot(r, fRr, 'LineWidth', 3);
xline(thresh, 'LineWidth', 3);
hold off;
```

```
title('MAP Detector for a Given p_0');
xlabel('Value of r');
ylabel('Probability Density');
grid on;
legend('Sample Values of r', 'Theoretical Value of r', 'Threshold');


% plot the smooth fRr over the histogram, as in previous projects
% make the plot professional

% 2.3.1
% Edit the output function of section 2.1
% tau = (sigma2 / (2 * A)) .* ln((1-p0) ./ p0);
%     = 0?
tau = (sigma2 / (2 * A)) * log((1-0.5) / 0.5);

% 2.3.2
% use q function
% step 3
func = @(x) (1 / (sqrt(2*pi*sigma2)) * exp(-(x-A).^2/(2*sigma2)));
condPDFA = integral(func, -Inf, 0);

% these are tests to show the equivalent values to that above
%thing = QQ(A/(sqrt(sigma2)));
%thing2 = (erf(A/sqrt(2*sigma2)) - 1) / -2;

% step 6
func2 = @(x) (1 / (sqrt(2*pi*sigma2)) * exp(-(x+A).^2/(2*sigma2)));
condPDFA2 = integral(func2, 0, Inf);

% step 7
p_BT = 0.5 * condPDFA + 0.5 * condPDFA2;


% Prepare for simulations of 2.3.3 and 2.4.3
% This does the simulations in a loop on values of p0

%Set Parameters

gamma_dB=[0:0.5:10 10.25:.25:14]; % A^2/sigma^2 in dB
gamma = 10.^(gamma_dB/10);  % as power ratio

A=1; % assume unity amplitude, per project description

Nbits = 5000000;% lots and lots of bits, 5,000,000 if your computer can handle
 it
p0=[0.5, 0.8]; % values of p0 for ML and MAP

thresholds=zeros(length(p0),length(gamma)); %thresholds vary with sigma2 and
 p0
results = zeros(length(p0),length(gamma));
pBT = zeros(length(p0),length(gamma));

for kP0=1:length(p0)
```

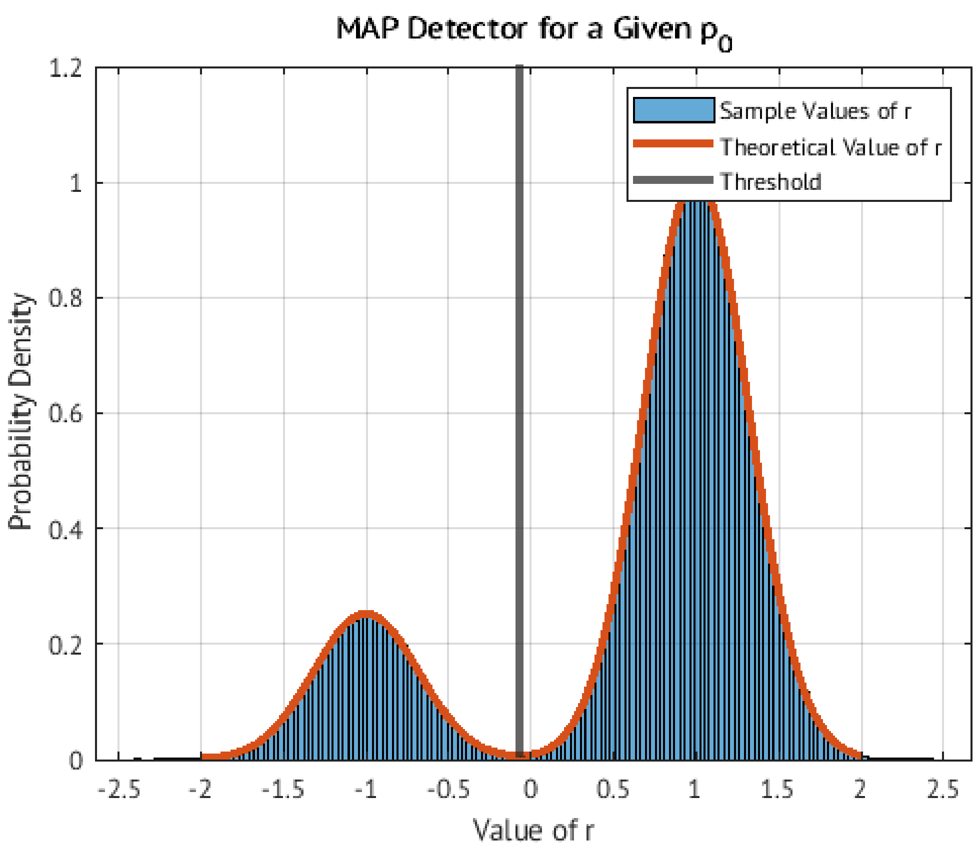
```
% START THE SIMULATION

    b= rand(1,Nbits) >= p0(kP0); % create the bits with appropriate
 probability 0 with p0, 1 with 1-p0
    m= -2 * (b - 0.5); % convert to +/-A (A=1) per equation 1.1, remeber 0->
+A, 1->-A

%Loop on SNR
    for kSNR=1:length(gamma)

    % One long continuous string of coded bits and add noise
    % This step both modulates and adds AWGN
            sigma2 = (A^2) / gamma(kSNR);  % use the energy ratio in gamma, not
 gamma_dB
            sigma =  sqrt(sigma2);  % compute sigma from variance, sigma2
            n = sigma * randn(1,Nbits); % compute noise values from
 N(0,sigma2);
            r = m + n; % create the received signal

            %Compute the thresholds as a function of sigma2, A, and p0, per
            %your derivation
            thresholds(kP0,kSNR) = (sigma2 / (2*A)) * log((1-p0(kP0)) /
 p0(kP0));

            bkhat = (r <= thresholds(kP0,kSNR)); % 1 if less than threshold, 0
 if greater
            errors = mod(bkhat - b,2); % 1 = error, 0 = no error
            results(kP0,kSNR) = sum(errors)/Nbits; % pBX for this SNR and this
 p

            pBT_given0_func = @(x) (1 / (sqrt(2*pi*sigma2)) * exp(-(x-A).^2/
(2*sigma2)));
            pBT_given1_func = @(x) (1 / (sqrt(2*pi*sigma2)) * exp(-(x+A).^2/
(2*sigma2)));
            pBT_given0 = integral(pBT_given0_func, -Inf, thresholds(kP0,
 kSNR));  % per your derivation
            pBT_given1 = integral(pBT_given1_func, thresholds(kP0, kSNR), Inf);
  % per your derivation
            pBT(kP0,kSNR) = p0(kP0) * pBT_given0 + (1-p0(kP0)) * pBT_given1; %
 Law of Total Probability;

    end; %loop on SNR

    %Plot the results for this P0 and all SNRs

        figure; % set new figure
        h=semilogy(gamma_dB,pBT(kP0,:),'k-',gamma_dB,results(kP0,:),'ro');
        set(h,'LineWidth',1.5);
        xlabel('Value of \gamma_{dB}');
        ylabel('Log_{10} of p_{BT}');
        title(['\gamma_{dB} vs Bit Error Probability for p_0 = ',
 num2str(p0(kP0))]);
```

```
        grid on;
        legend('Theoretical Bit Error Probabilty', 'Simulated Probability');

        % Make your plots professional with labels, scales, grids, legends
        % etc.


end;% loop on P0

func = @(x) (1 / (sqrt(2*pi*sigma2)) * exp(-(x-A).^2/(2*sigma2)));
condPDF = integral(func, -Inf, thresh);
% thing = QQ((A - thresh)/(sqrt(sigma2)));
% step 6
func2 = @(x) (1 / (sqrt(2*pi*sigma2)) * exp(-(x+A).^2/(2*sigma2)));
condPDF2 = integral(func2, thresh, Inf);
% thing2 = QQ((thresh + A)/(sqrt(sigma2)));
% step 7
p_BT2 = 0.8 * condPDF + 0.2 * condPDF2;

% Section 2.5

    % Combined plot
    figure; % set new figure
    % Plot both of the theoretical curves on the same axes
        h=semilogy(gamma_dB,pBT(1,:),'k-',gamma_dB,pBT(2,:),'r--'); % plot
 both theoretical on same scale
        set(h,'LineWidth',1.5);
        xlabel('Value of \gamma_{dB}');
        ylabel('Log_{10} of p_{BT}');
        title('log_{10} of p_{BT} for MAP and ML Detectors');
        grid on;
        legend('p_{BT} for p_0 = 0.5', 'p_{BT} for p_0 = 0.8');
    % Make the plot professional

    % Now plot the ratio of the theoretical errors, with p0=0.5 in
    % denominator
    figure; % another new figure
    rho = pBT(2,:) ./ pBT(1,:);
    plot(gamma_dB, rho, 'LineWidth', 3);
    ylabel('Value of \rho');
    xlabel('Value of \gamma_{dB}');
    title(['Ratio of p_{BT} for p_0 = 0.8 and 0.5']);
    grid on;
    legend('Ratio of p_{BT} for MAP and ML Detectors');
    % compute the ratio of the MAP probability of error (pBT(2,:)) and the
    % ML probability of error (pBT(1,:)) and plot with gamma_dB on the
    % x-axis


    % Make the plot professional
```

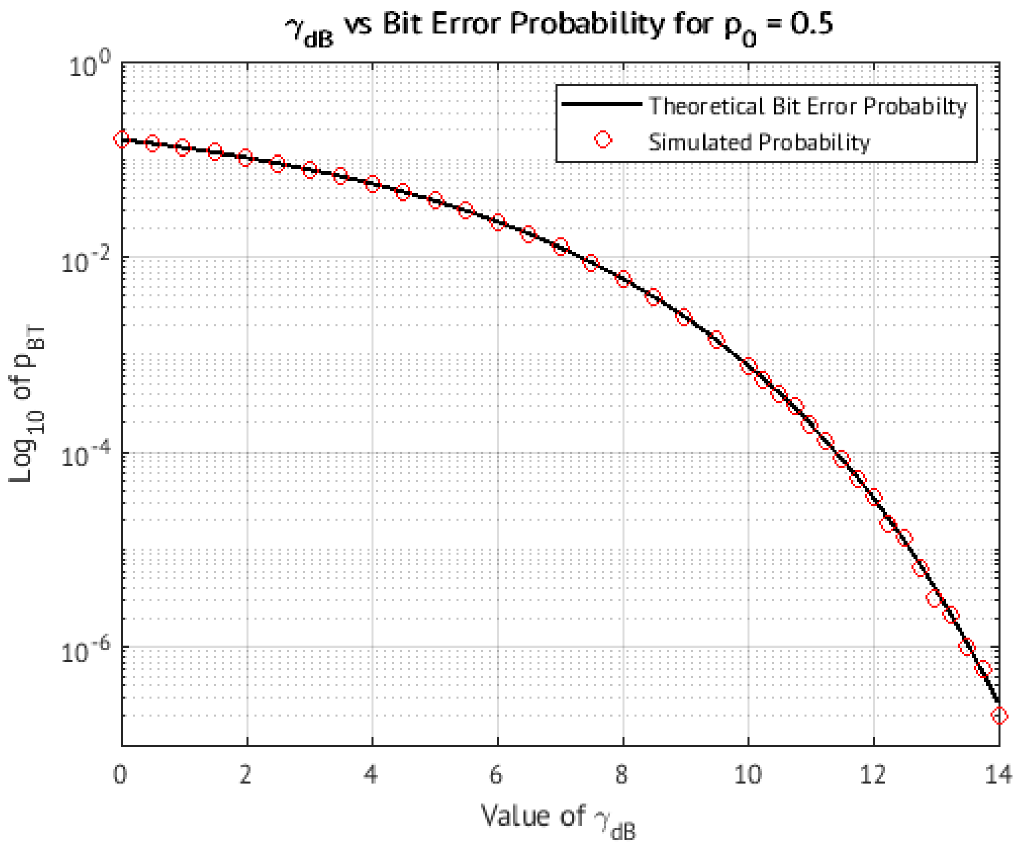
CMPE320 Spring 2022 Project 4: BASK

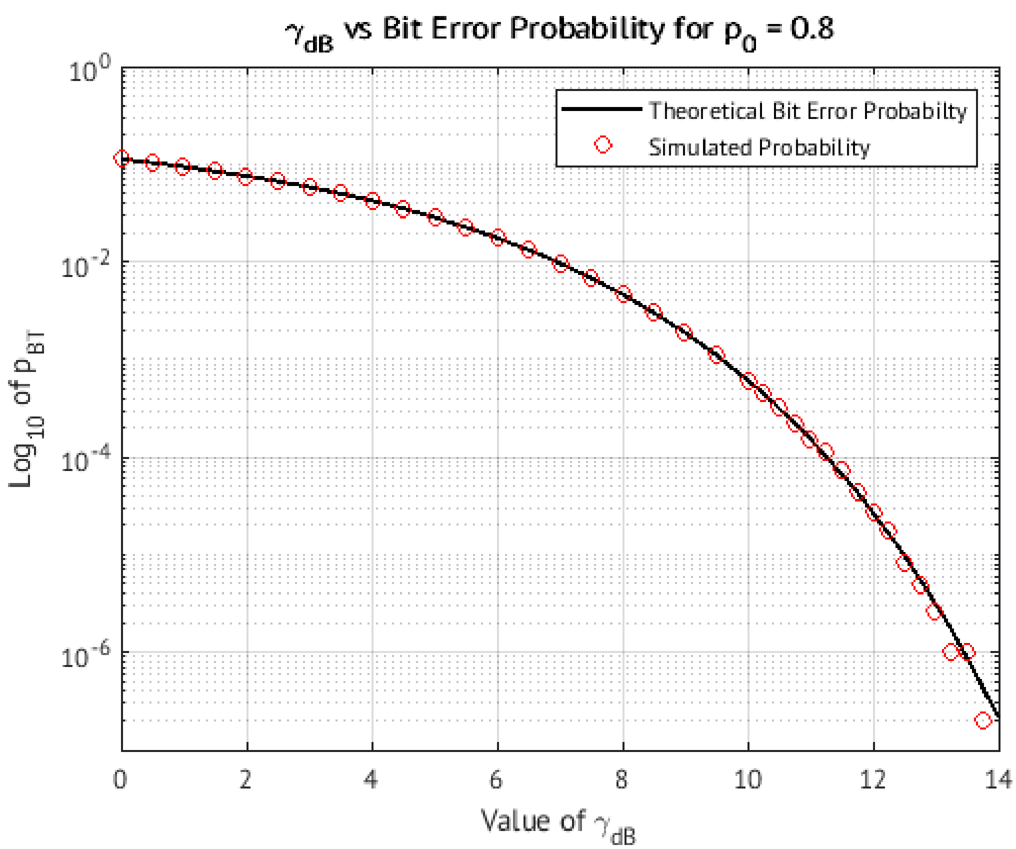


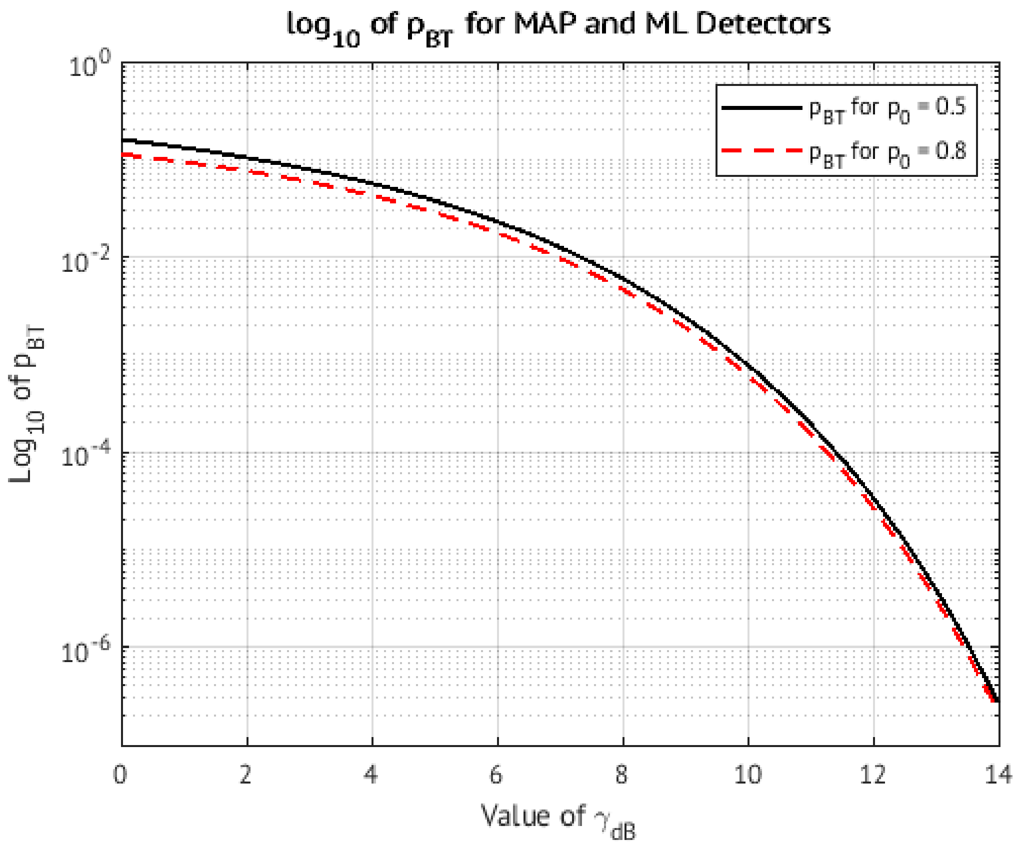
Threshold Value Across p_0
Threshold Value
Value of p_0
Threshold Value



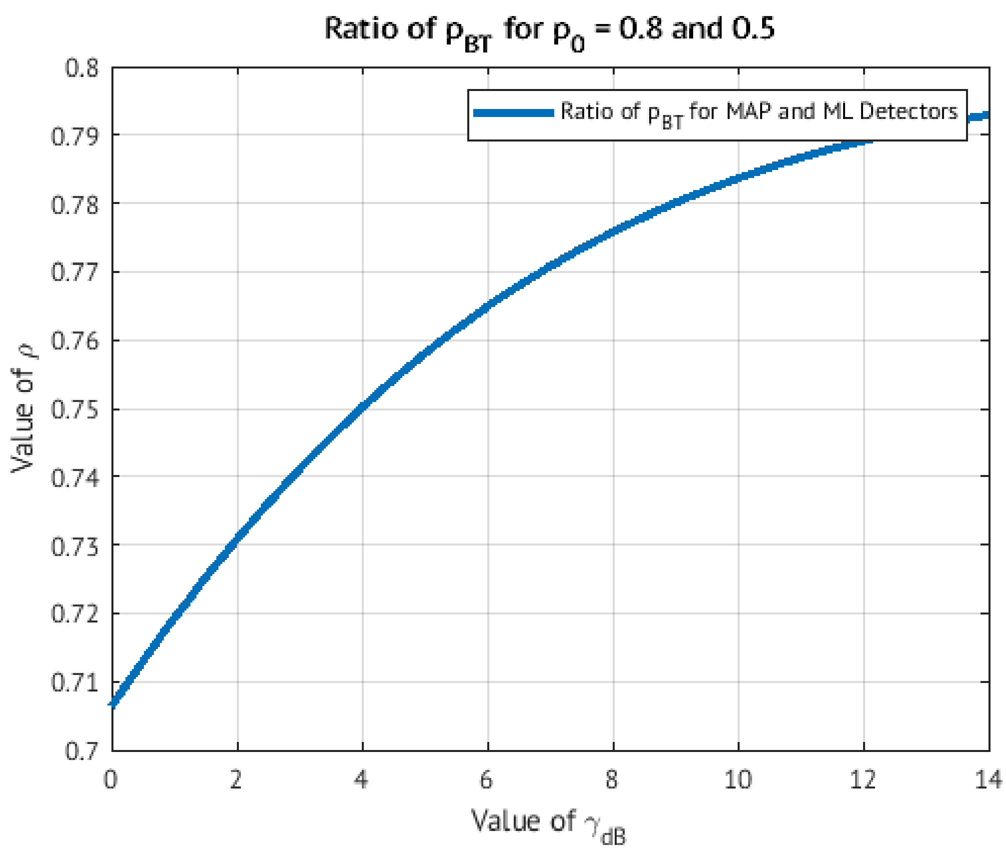
MAP Detector for a Given p_0
Probability Density
Value of r
Sample Values of r
Theoretical Value of r
Threshold


γ_{dB} vs Bit Error Probability for p_0 = 0.5
Theoretical Bit Error Probabilty
Simulated Probability
10^0
10^{-2}
10^{-4}
10^{-6}
Log_{10} of p_{BT}
0
2
4
6
8
10
12
14
Value of γ_{dB}


γ_{dB} vs Bit Error Probability for p_0 = 0.8
Theoretical Bit Error Probabilty
Simulated Probability
10^0
10^{-2}
10^{-4}
10^{-6}
Log_{10} of p_{BT}
0
2
4
6
8
10
12
14
Value of γ_{dB}



log10 of ρBT for MAP and ML Detectors
ρBT for ρ0 = 0.5
ρBT for ρ0 = 0.8
Log10 of ρBT
Value of γdB



Ratio of ρBT for ρ0 = 0.8 and 0.5
Ratio of ρBT for MAP and ML Detectors
Value of ρ
Value of γdB

Published with MATLAB® R2021b