Department of Electrical Engineering
The Cooper Union for the Advancement of Science and Art

MATLAB Detection Excercises

By

Sameer Chauhan & Sharang Phadke

Advisor: Prof. Keene

ECE 302 - Spring 2013

1 Introduction

The purpose of detection is to be able to separate information-bearing patterns from random patterns that distract from the transfer of information. Depending on the amount of distracting noise and quality of underlying information, this task can be difficult. A simple approach for this problem is to compare the signal to a threshold determined by a decision rule based on the characteristics of the signal and channel. In the following exercises the parameters of a MAP detection system are examined. Channel characteristics such as varying priors, SNRs, and transmission distributions are considered, and detector constraints such as the minimax rule, as well as different cost structures are analyzed.

Sometimes detection cannot be done with just a threshold because the data set contains multiple features and classes. Machine learning remedies this problem by using statistical characteristics of the datasets to classify them. A machine learning system using a MAP estimate is trained, and its performance is tested using the Iris dataset.

2 Radar Detection

In this part, several radar detection systems are considered. The primary system modeled is a detector that implements the MAP rule for a target with energy A in an AWGN channel. The characteristics of the detection rule with various prior and cost structures, as well as with various SNRs are analyzed.

2.1 The MAP Detector

The specified radar system can be characterized as follows, where x is Gaussian noise:

$$\begin{array}{lll} H_0 : y = x & y_0 \sim N(0, \sigma^2) & p_0 = 0.7 \\ H_1 : y = A + x & y_1 \sim N(A, \sigma^2) & p_1 = 0.2 \end{array}$$

Table 1: Radar Detection System Parameters

The MAP detector for this system can be derived by evaluating the general MAP rule for the two Gaussians specified above:

$$H_{MAP} = argmax\{P[H|Y = y]\}$$
$$P[H_1|Y = y] \underset{H_0}{\overset{H_1}{\gtrless}} P[H_0|Y = y]$$

$$P[Y|H_1]P[H_1] \underset{H_0}{\overset{H_1}{\gtrless}} P[Y|H_0]P[H_0]$$

$$\left(\frac{1}{\sqrt{2\pi\sigma^2}}e^{-(y-A)^2/2\sigma^2}\right)(p_1) \underset{H_0}{\overset{H_1}{\gtrless}} \left(\frac{1}{\sqrt{2\pi\sigma^2}}e^{-(y)^2/2\sigma^2}\right)(p_0)$$

The threshold, Γ, can be found by taking the log and solving for y:

$$\Gamma = \frac{2\sigma^2 \ln\left(\frac{p_0}{p_1}\right) + A}{2A}$$

In addition, the theoretical probability of error of the detector can be computed by integrating the tails of the two Gaussians:

$$\begin{aligned} P[Error] &= P[FalseAlarm] + P[Miss] \\ &= \int_{\Gamma}^{\infty} N(0,\sigma^2)\,dx + \int_{-\infty}^{\Gamma} N(A,\sigma^2)\,dx \end{aligned}$$

The detection system was simulated over 1000 iterations, each with a transmission length of 1000 symbols. A target energy of 1 was used. At each step, the theoretical probability of error of the detector was computed using the MATLAB *qfunc* function. The theoretical and simulated probabilities of error as a function of noise variance are plotted in Figure 1. The two plots match almost exactly, and show that the detector's performance deteriorates severely past a variance of 1. This is because as the threshold increases in proportion to the noise variance, and the P[Miss] increases dramatically, as more and more of the target Gaussian, $N(0, \sigma^2)$, falls below the threshold.

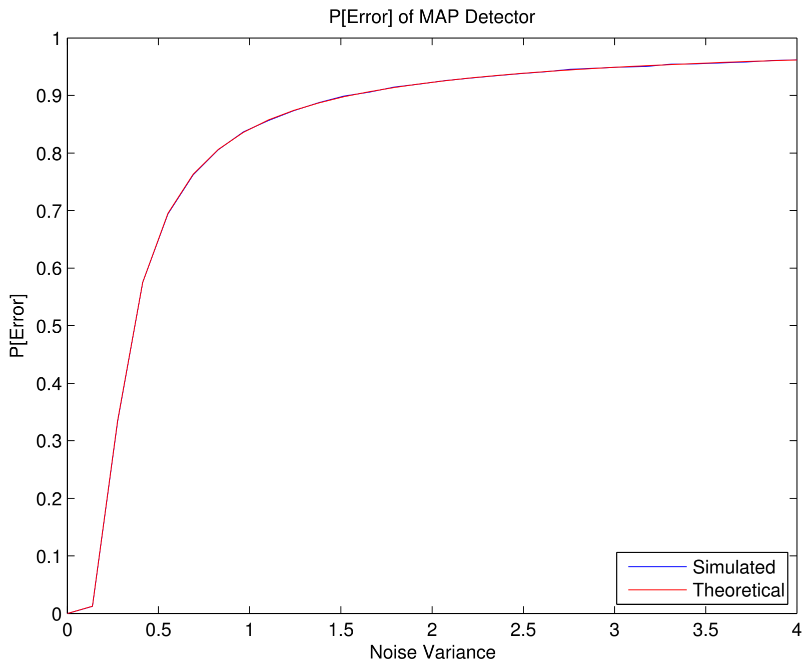

Figure 1: Theoretical and Simulated P[Error] of MAP Detector

In addition, the ROCs of the detector were generated at several noise variances from 0 to 2.5 by varying Γ. They are plotted in Figure 2. As is expected, the ROC approaches a unit step as the noise variance decreases to 0, because the SNR, A/σ^2, consequently goes to infinity. The markers on each ROC indicate the operating point of the detectors with the cost structure specified in the next section.

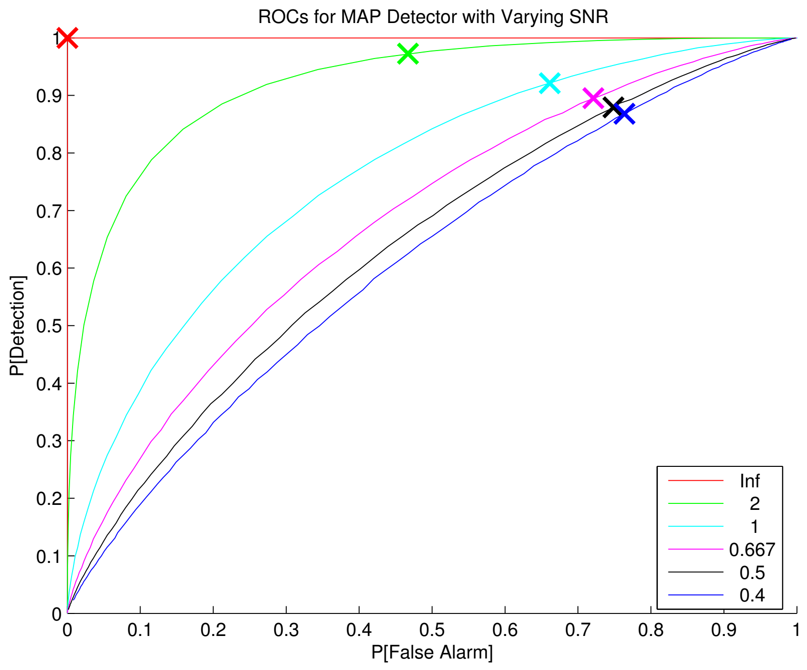

Figure 2: ROCs of MAP Detector with Varying SNR

2.2 High Miss Cost

Until now, the most basic cost structure has been assumed - the cost of any error has been 1, and the cost of no error has been 0. However, the cost structure of a detector depends on the specific application in question. In this part, a high miss cost structure is implemented. The threshold for this cost structure is derived in almost the same manner as before, modulated by the cost of making each type of error:

$$c_{00} = 0 \;\; c_{11} = 0 \;\; c_{01} = 10 \;\; c_{10} = 1$$

$$\frac{P[y|H_1]}{P[y|H_0]} \underset{H_0}{\overset{H_1}{\gtrless}} \left(\frac{c_{10} - c_{00}}{c_{01} - c_{11}}\right)\left(\frac{p_0}{p_1}\right)$$

$$\begin{aligned}\frac{P[y|H_1]}{P[y|H_0]} &\underset{H_0}{\overset{H_1}{\gtrless}} \left(\frac{1-0}{10-0}\right)\left(\frac{p_0}{p_1}\right)\\ &\underset{H_0}{\overset{H_1}{\gtrless}} \left(\frac{p_0}{10p_1}\right)\end{aligned}$$

Following the same procedure as before, a new threshold can be obtained with this cost structure:

$$\Gamma = \frac{2\sigma^2 \ln\left(\frac{p_0}{10p_1}\right) + A}{2A}$$

For a particular SNR, this threshold specifies an operating point on the ROC. The operating points of each of the systems simulated in the previous part are marked with an "x".

2.3 Unknown Priors - Minimax Decision Rule

In many cases, the priors of a system are unknown, and an operating point on the ROC must be found by establishing an additional constraint. The minimax decision rule minimizes the maximum expected cost of a detection system by intentionally guessing the priors that have the highest conditional cost. This result can be seen in the following derivation, which evaluates the Bayes risk of a detector based on the assumed priors, p_1 and p_0, and the true priors, p_1^* and p_0^*:

$$\begin{aligned}E[Cost, p_1, p_1^*] &= \sum c_{ij} p[D(y) = H_i | H_j\, true]\\ &= c_{00}p_0^* + c_{01}p_1^* + (c_{10} - c_{00})p_0^* P_F(p_1) - (c_{01} - c_{11})p_1^* P_D(p_1)\\ &= [(c_{01} - c_{00}) - (c_{10} - c_{00})P_F - (c_{01} - c_{11})P_D]p_1^* + c_{00} + (c_{10} - c_{00})P_F\end{aligned}$$

This cost function can be minimized in a minimax sense by setting its derivative with respect to p_1^* equal to 0 and finding the intercept of the resulting function, $P_D(P_F)$, and the ROC. Following this procedure, that function can be found:

$$\begin{aligned}0 &= \frac{d}{dp_1^*}\left(E[Cost, p_1, p_1^*]\right)\\ &= [(c_{01} - c_{00}) - (c_{10} - c_{00})P_F - (c_{01} - c_{11})P_D]\end{aligned}$$

Solving for P_D,

$$P_D = \left(\frac{c_{01} - c_{00}}{c_{01} - c_{11}}\right) - \left(\frac{c_{10} - c_{00}}{c_{01} - c11}\right) P_F$$

This constraint was used to find an operating point on the ROC of this system with an SNR of 1, and is indicated by the intersection of the constraint and the ROC in Figure 3:

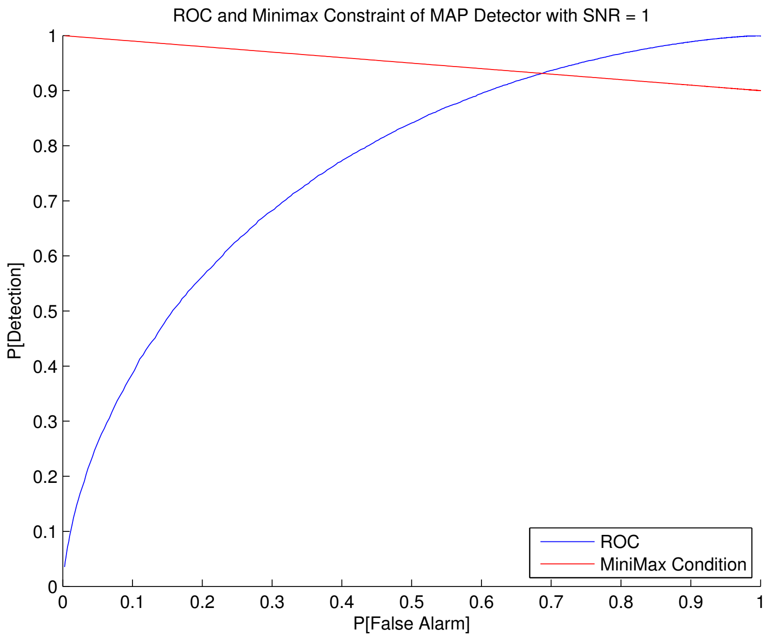

Figure 3: ROC and Minimax Constraint of MAP Detector with SNR = 1

Using this operating point, the minimax cost can be plotted as a function of the priors. Figure 4 shows the expected cost of the optimal MAP detector (with known priors) as well as the cost of implementing the minimax decision rule. Although the minimax cost is significantly higher than the expected cost for certain prior specifications, the minimax rule ensures that the detector doesn't produce extremely high cost for guessing priors wrong.

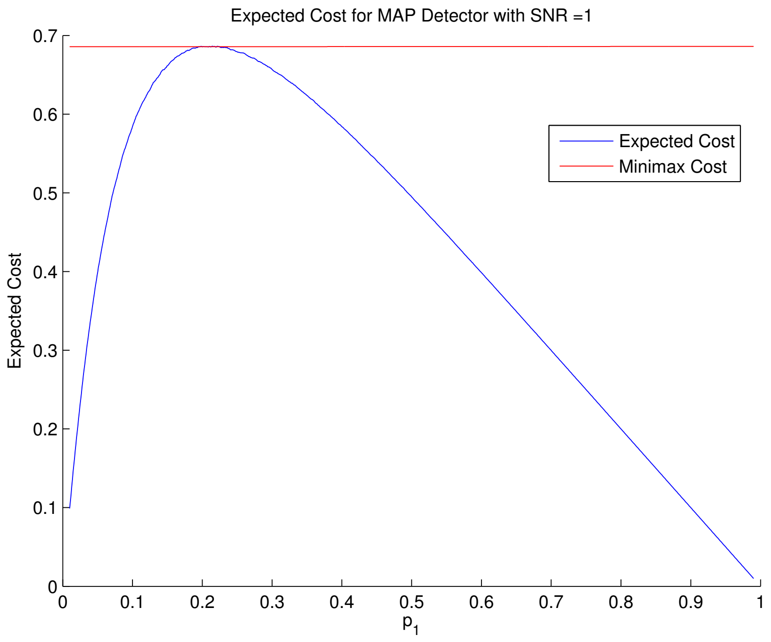

Figure 4: Expected Cost for MAP Detector with SNR = 1

2.4 Constant Mean, Different Variance

Now the system is modified so that it is the following where $\sigma_0^2 > \sigma_1^2$:

$$\begin{aligned} H_0 : y = A + z \quad & y_0 \sim N(A, \sigma_0^2) \quad p_0 = 0.8 \\ H_1 : y = A + x \quad & y_1 \sim N(A, \sigma_1^2) \quad p_1 = 0.2 \end{aligned}$$

Table 2: Constant Mean Detection System Parameters

Evaluating the general MAP rule for the two Gaussians as in Section 2.1 we obtain the following relationship:

$$P[Y|H_1]P[H_1] \underset{H_0}{\overset{H_1}{\gtrless}} P[Y|H_0]P[H_0]$$

$$\left(\frac{1}{\sqrt{2\pi\sigma_1^2}} e^{-(y-A)^2/2\sigma_1^2}\right)(p_1) \underset{H_0}{\overset{H_1}{\gtrless}} \left(\frac{1}{\sqrt{2\pi\sigma_0^2}} e^{-(y-A)^2/2\sigma_0^2}\right)(p_0)$$

$$\frac{(y-A)^2}{2}\left(\frac{\sigma_1^2 - \sigma_0^2}{\sigma_0^2\sigma_1^2}\right) \underset{H_0}{\overset{H_1}{\gtrless}} \ln\left(\frac{p_0}{p_1}\frac{\sigma_1}{\sigma_0}\right)$$

Since the system consists of two Gaussians at the same mean, the detector rule will have two thresholds. For the theoretical thresholds, the relationship was analytically solved to obtain a quadratic equation with the following values:

$$\begin{aligned} B &= A(\sigma_1^2 - \sigma_0^2) \\ \sqrt{B^2 - 4AC} &= \sqrt{4C\sigma_0^2\sigma_1^2(\sigma_1^2 - \sigma_0^2)} \\ A &= \frac{\sigma_1^2 - \sigma_0^2}{2} \end{aligned}$$

Like in Section 2.1, the theoretical probability of error is the sum of P[FalseAlarm] and P[Miss] which can be computed by integrating the tails of the two distributions.

$$\begin{aligned} P[Error] &= P[FalseAlarm] + P[Miss] \\ &= 2\int_{\Gamma}^{\infty} N(A, \sigma^2)\, dx + \left(1 - 2\int_{-\infty}^{\Gamma} N(A, \sigma_0^2)\, dx\right) \end{aligned}$$

The theoretical probability of error was compared with the simulated probability of error (as obtained in Section 2.1) to obtain the following graph:

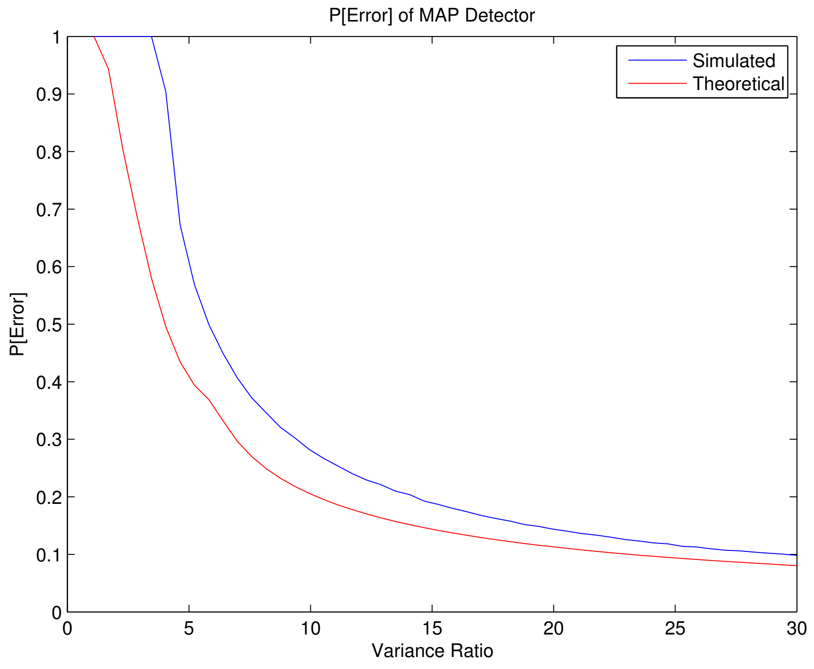

Figure 5: Probabilities of Error

It can be seen that variance ratios under 5, the probability of error is 1. This is because when the two Gaussians are very similar, the rate of false alarms is very high. As the variance ratio increases, we see that the probability of error drops dramatically. At high noise variance, the PDF of the noise flattens out causing less false alarms because the peak of the target PDF dominates the area between the two thresholds. The ROC for different SNR values was plotted by varying the threshold.

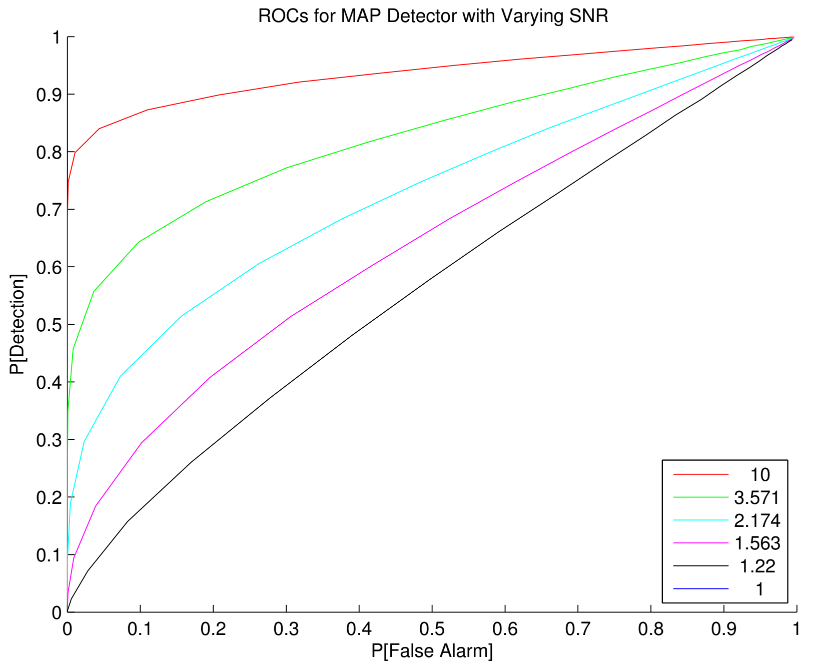

Figure 6: ROCs with Varying SNR

As SNR increases the the ROC of the detector approaches the unit step. It is interesting to note that the P_D ROC is relatively high for low values of P_F. This suggests that this is a better the one simulated in Section 2.1.

3 Non-Gaussian Detection

In this part a photon detector that models the inter-arrival time of photons as an exponential random variable was simulated. In this scenario, the sender transmits photons at a rate of λ_0 or λ_1 over a one second inverval, which correspond to 0's and 1's respectively. Equiprobable priors of $p_0 = p_1 = 0.5$ were chosen for the transmission probabilities and λ_0 and λ_1 were chosen to be 5 and 8 photons/second respectively. For each second, interarrival times, t_n, were drawn from the following exponential pdf, where λ_i corresponds to the sending a 0 or 1:

$$t_n = \frac{1}{\lambda_i} e^{\frac{n}{\lambda_i}}$$

Given that t_n is an exponential random variable, he number of events, y, can be modeled as a Poisson random variable and used to compute values for the likelihood ratio test:

$$\begin{aligned} P(y|H_i) &= \frac{\lambda_i^y}{y!} e^{-\lambda} \\ L(y) &= \frac{P(y|H_1)}{P(y|H_0)} \underset{H_0}{\overset{H_1}{\gtrless}} \frac{p_0}{p_1} \\ &= \left(\frac{\lambda_1}{\lambda_0}\right)^y e^{(\lambda_0 - \lambda_1)} \underset{H_0}{\overset{H_1}{\gtrless}} 1 \end{aligned}$$

The Likelihood Ratio Test allows us to calculate the probability of detection P_D and probability of false alarms P_F, similarly to Section 2. Again, the ROC of this detector was generated by varying the threshold (Γ) from 10^{-5} to 10^5 on a logarithmic scale. For multiple ROC curves, the ratio of rates, λ_1/λ_0 was varied from 1 to 3.5. λ_0 was fixed at 5 photons/second.

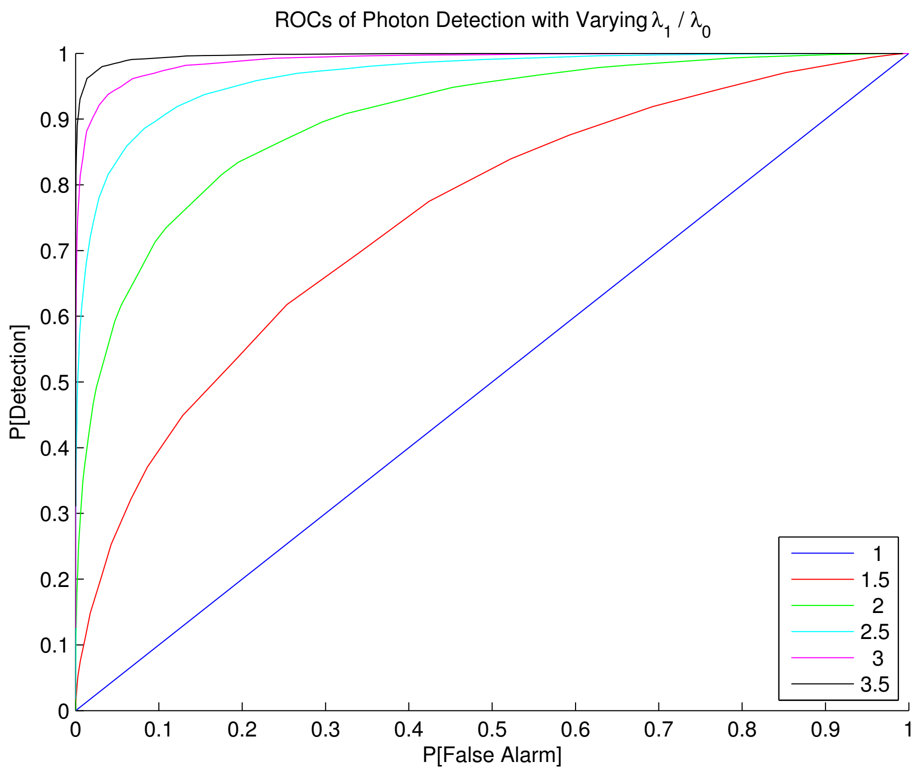

Figure 7: ROCs for Varying Transmission Rates

It can be seen that as ratio between the rates increases, the ROC approaches the unit step. This is expected because as the rates begin to differ, there is a clear discrepency between the number of events that appear in one second. This allows the MAP detection rule (equivalent to ML for equiprobable priors) to properly guess the transmitting rate and detect a 0 or 1. At $\lambda_1/\lambda_0 = 1$ the ROC is a straight line because the detector cannot properly differentiate between the two distributions. The rates, λ_1 and λ_0, are equal, which causes the detection rule to simplify to $1 \underset{H_0}{\overset{H_1}{\gtrless}} \Gamma$. Since the thresholds vary from 10^{-5} to 10^5, the detector either always chooses 1 for $\Gamma \in [10^{-5}, 1)$ or always 0 for $\Gamma \in [1, 10^5]$. For the first interval $P_D = P_F = 1$ and for the second inverval $P_D = P_F = 0$. So for the ratio of $\lambda_1/\lambda_0 = 1$, the only two points on the ROC are (0,0) and (1,1).

4 Pattern Classification and Machine Learning

In this part, we extend the idea of MAP classification by incorporating the means and variances of sample data. So far, we have done MAP classification by comparing a-posteriori probabilities to a threshold. So when given a data set, the MAP classification does not use any other statistical information about the classifcations. However, we can use machine learning principles to include information about the means and variances of each classification. Given 150 labeled samples of data with 4 features, we can train this new system on half of the data and test the classifier on the remaining data, effectively implementing a 2-fold cross-validation.

To accomplish this, the *Iris* dataset was randomly divided into two different datasets: the training dataset and testing dataset. Then for each class the priors, and means, and covariances of each feature were calculated. For an ideal division of data the priors would be equal at 0.33, however randomly dividing the data changes these values. The statistical data was used for calculating the multivariate normal probability of each data sample for each class. The MAP estimate was calculated by multiplying this likihood by the prior of each class, so each sample had a MAP estimate for all three classes. The maximum of the three MAP values were taken to determine the final classification. The following is the confusion matrix for the constructed classifer:

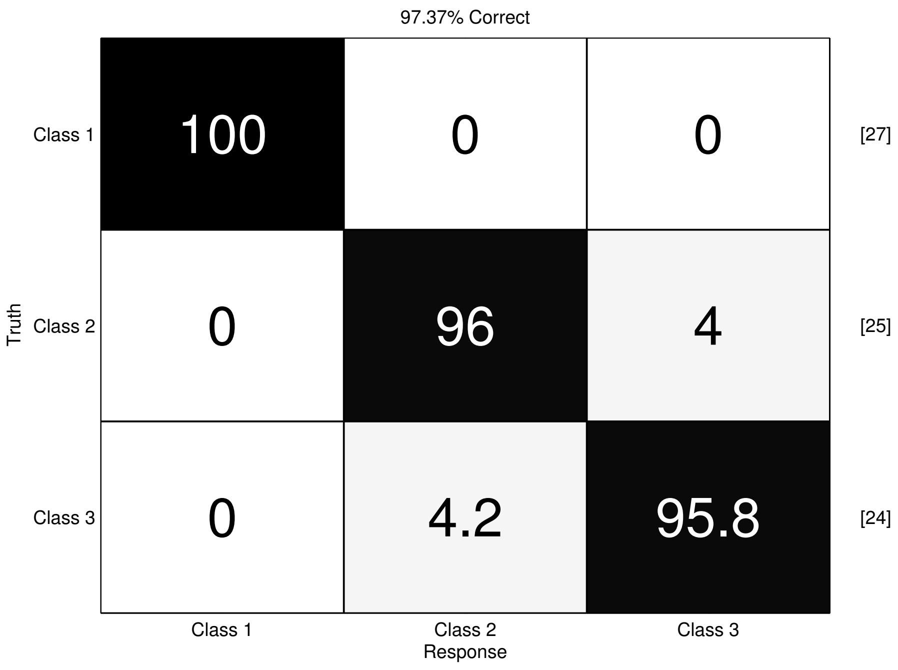

Figure 8: Confusion Matrix of a Sample Classifier Run

Since our data was randomly divided into 2 halves, we ran the classification 1000 times to obtain an average probability of error of 2.9%. The classifier always classified correctly for class 1 but sometimes confused class 2 and class 3. This is due to the fact that some data samples are very similar to both class 2 and 3.

5 Appendix

5.1 Part 1

```
% Detection Exercise
% Part 1
% ECE 302
% Sameer Chauhan
% Sharang Phadke
%

clear all; close all

%% MAP Detector
p0 = 0.8; p1 = 1-p0;
A = 1;
nIter = 1000;
sigLen = 1000;

diffVar = linspace(0,4,30);
PdMAP = zeros(length(diffVar),1);
PfMAP = zeros(length(diffVar),1);
MAPthreshold = (2*diffVar*log(p0/p1) + A^2)/(2*A);

for kk = 1:length(diffVar)
    [PdMAP(kk), PfMAP(kk)]  =  mapDetect(MAPthreshold(kk), ...
        diffVar(kk), A, nIter, sigLen, p1);
end

PErrorTheory = (1-qfunc((MAPthreshold-A)./diffVar)) + ...
    qfunc(MAPthreshold./diffVar);
```

```
PErrorSim = 1-PdMAP + PfMAP;

figure(1)
plot(diffVar,PErrorSim,diffVar,PErrorTheory,'r')
title('P[Error] of MAP Detector')
xlabel('Noise Variance')
ylabel('P[Error]')
legend('Simulated','Theoretical','Location','SouthEast')


%% ROCs
diffVar = linspace(0,2.5,6);
diffThresh = -6*A:.1:6*A;
Pd = zeros(length(diffVar),1);
Pf = zeros(length(diffVar),1);
colors = ['b','r','g','c','m','k'];

figure(2)
hold on

for kk = 1:length(diffVar)
    var = diffVar(kk);
    for ii = 1:length(diffThresh)
        [Pd(ii), Pf(ii)] = mapDetect(diffThresh(ii), var, A, ...
            nIter, sigLen, p1);
    end
    plot(Pf,Pd,colors(mod(kk,length(colors))+1));
end

title('ROCs for MAP Detector with Varying SNR')
xlabel('P[False Alarm]')
ylabel('P[Detection]')
```

```
legend(num2str(round(1000*A./diffVar)'/1000),'Location','Best');

%% High Miss Cost
c10 = 1; c01 = 10;
c00 = 0; c11 = 0;

% Threshold for high miss cost structure
NoMissThreshold = (2*diffVar*log(c10*p0/(c01*p1)) + A^2)/(2*A);

for kk =1:length(NoMissThreshold)
    [PdNoMiss, PfNoMiss] = mapDetect(NoMissThreshold(kk), ...
       diffVar(kk), A, nIter, sigLen, p1);
    plot(PfNoMiss,PdNoMiss,strcat('x',colors(mod(kk,length(colors))+1)),'Mar
       2)
end

%% Varying Priors

var = 1;
p1s = linspace(.01,.99,1000)';
p0s = 1 - p1s;
NoMissThreshold = (2*var*log(c10*p0s./(c01*p1s)) + A^2)/(2*A);
ECost = zeros(length(p1s),1);
Pd = zeros(length(p1s),1);
Pf = zeros(length(p1s),1);

for kk = 1:length(p1s)
    [Pd(kk), Pf(kk), ECost(kk)] = ...
       mapDetect(NoMissThreshold(kk), var, A, nIter, sigLen, ...
       p1s(kk), c10, c01);
end
ECost(isnan(ECost)) = 0;
```

```
Pd(isnan(Pd)) = 0;
Pf(isnan(Pf)) = 1;

figure(3)
hold on
plot(p1s,smooth(ECost))

title(strcat('Expected Cost for MAP Detector with Noise ...
   \sigma^2 = ',num2str(var)))
xlabel('p_1')
ylabel('Expected Cost')


%% Unknown Priors — Minimax Decision Rule

MMslope = -(c10-c00)/(c01-c11);
MMintercept = (c01-c00)/(c01-c11);
minDiff = min(abs(Pd-(MMintercept+MMslope*Pf)));
PdMM = Pd(abs(Pd-(MMintercept+MMslope*Pf)) == minDiff);
PfMM = Pf(abs(Pd-(MMintercept+MMslope*Pf)) == minDiff);
PdMM = PdMM(1);
PfMM = PfMM(1);


PdOp = MMintercept+MMslope*Pf;

slope = (c01-c00)-(c10-c00)*PfMM-(c01-c11)*PdMM;
intercept = c00+(c10-c00)*PfMM;
MMECost = slope*p1s +intercept;

figure(3)
plot(p1s,MMECost,'r')

figure(4)
```

```
hold on
plot(Pf,smooth(Pd))
plot(Pf,PdOp,'r')
title('ROCs for MAP Detector with Noise \sigma^2 = 1')
xlabel('P[False Alarm]')
ylabel('P[Detection]')
legend('ROC','MiniMax Condition','Location','SouthEast')

%% Different Variances, Same Mean


var1 = 0.5;
varRatio = linspace(1.1,30,50);
PErrorTheory = zeros(length(varRatio),1);
PErrorSim = zeros(length(varRatio),1);
PdMAP = zeros(length(varRatio),1);
PfMAP = zeros(length(varRatio),1);

for kk = 1:length(varRatio)
    var0 = varRatio(kk)*var1;

    C = log((p0*var1)/(p1*var0));
    negB = A*var1-A*var0;
    sqrtBAC =sqrt(4*C*var0*var1*(var1-var0)) ;
    twoA = var1 - var0;

    MAPthreshold = abs(negB+sqrtBAC)/twoA;

    Pmiss = 2*qfunc((MAPthreshold-A)/var1);
    PFA = 1-2*qfunc((MAPthreshold-A)/var0);
    PErrorTheory(kk) = 2-(Pmiss + PFA);

    [PdMAP(kk), PfMAP(kk)]  =  mapDetect2(var1, var0, A, ...
       nIter, sigLen, p1);
    PErrorSim(kk) = 1-PdMAP(kk) + PfMAP(kk);
end

figure(5)

plot(varRatio,PErrorSim,varRatio,smooth(PErrorTheory),'r')
title('P[Error] of MAP Detector')
xlabel('Variance Ratio')
ylabel('P[Error]')
legend('Simulated','Theoretical','Location','NorthEast')




%% ROCs
var1 = 1;
% varRatio = linspace(1.1,30,6);

varRatio = linspace(0.1,1,6);
% diffThresh = -6*A:.1:6*A;
diffThresh = logspace(-5,5,50);
Pd = zeros(length(diffThresh),1);
Pf = zeros(length(diffThresh),1);

colors = ['b','r','g','c','m','k'];

figure(6)
hold on

for kk = 1:length(varRatio)
    var0 = varRatio(kk)*var1;
    for ii = 1:length(diffThresh)
        [Pd(ii), Pf(ii)] = mapDetect2(var1, var0, A, nIter, ...
            sigLen, p1, diffThresh(ii));;
    end
    plot(Pf,Pd,colors(mod(kk,length(colors))+1));
end

title('ROCs for MAP Detector with Varying SNR')
xlabel('P[False Alarm]')
ylabel('P[Detection]')
legend(num2str(round(1000*A./varRatio)'/1000),'Location','SouthEast');
```

5.2 Part 2

```
% Detection Exercise
% Part 2
% ECE 302
% Sameer Chauhan
% Sharang Phadke
%

clear all; close all;

%% Non-Gaussian Detection

p1 = 0.5; p0 = 1-p1;
maxNumDraws = 100;
transmissionLen = 100;

nIter = 500;
```

```
lambda0 = 5;

diffThres = logspace(-5,5,100);
Pd = zeros(length(diffThres),nIter);
Pfa = zeros(length(diffThres),nIter);
colors = ['b','r','g','c','m','k'];
diffLambda = linspace(1,3.5,length(colors));

figure(1)
hold on
for m = 1:length(diffLambda)
    lambda1 = diffLambda(m)*lambda0;
    for n = 1:length(diffThres)
        for i = 1:nIter
            % create a transmission signal
            x = zeros(transmissionLen,maxNumDraws);
            r = zeros(transmissionLen,1);
            r(rand(transmissionLen,1) ≤ p1) = 1;

            % encode the signal as draws from two exp ...
               distributions
            x(r == 1,:) = exprnd(1/lambda1,sum(r),maxNumDraws);
            x(r == 0,:) = exprnd(1/lambda0,sum(¬r),maxNumDraws);

            % limit symbols to 1 second
            cSum = cumsum(x,2) > 1;
            x(cSum) = 0;

            % decode the signal by counting the number of ...
               photons/sec
            y = x;
            y(y > 0) = 1;
```

```
                y = sum(y,2);

                % detect the distribution using a Max LRT rule
                H = (lambda1/lambda0).^y*exp(lambda0-lambda1) > ...
                   diffThres(n);

                Pd(n,i) = sum(H.*(r))/sum(r);
                Pfa(n,i) = sum(H.*(¬r))/sum(¬r);
            end
        end

PfAvg = mean(Pfa,2);
PdAvg = mean(Pd,2);

PfAvg(isnan(PdAvg))=[]; PdAvg(isnan(PdAvg))=[];
PdAvg(isnan(PfAvg))=[]; PfAvg(isnan(PfAvg))=[];

plot(smooth(PfAvg),smooth(PdAvg), colors(m))
end


legend(num2str(diffLambda'),'Location','SouthEast')
title('ROCs of Photon Detection with Varying \lambda_1 / ...
   \lambda_0');
ylabel('P[Detection]'); xlabel('P[False Alarm]');
```

5.3 Part 3

```
% Detection Exercise
% Part 3
% ECE 302
% Sameer Chauhan
% Sharang Phadke
%

clear all; close all;
load Iris.mat;


numTrials = 1000;
PError = zeros(numTrials,1);

for m = 1:numTrials
    % divide up the dataset
    r = randi([0,1],length(labels),1);
    trainingDS = features(r==1,:);
    testingDS = features(r==0,:);
    trainingL = labels(r==1);
    testingL = labels(r==0);

    % train MVNs for each of the 3 classes
    Ls = unique(trainingL);
    Mu = cell(3,1);
    Cov = cell(3,1);
    priors = zeros(length(Ls),1);

    for n = 1:length(Ls)
        Mu{n} = mean(trainingDS(trainingL==Ls(n),:));
        Cov{n} = cov(trainingDS(trainingL==Ls(n),:));
        priors(n) = mean(trainingL==Ls(n));
    end

    % compute likelihoods
    Likelihoods = cell(1,3);
    for n = 1:length(Ls)
        Likelihoods{n} = mvnpdf(testingDS,Mu{n},Cov{n});
    end

    % compute MAP
    MAP = ...
       cell2mat(Likelihoods).*repmat(priors',length(testingL),1);
    [¬,Guess] = max(MAP,[],2);
    PError(m) = mean(Guess ≠ testingL);
end

avgPError = mean(PError)
prtScoreConfusionMatrix(Guess,testingL)
```

5.4 mapDetect

```
function [ Pd, Pf, ECost] = mapDetect( threshold, var, A, ...
   nIter, sigLen, p1, c10, c01, c00, c11)
%MAPDETECT Computes the P[Detection], P[False Alarm], and the ...
   expected cost
%of a detector with a signal of power A and a specified ...
   threshold in a
%chanel with AWGN of variance var with a particular prior and ...
   cost
%structure
%
% Usage:
%   [ Pd, Pf, ECost] = mapDetect( threshold, var, A, nIter, ...
   sigLen, p1)
```

```
%   [ Pd, Pf, ECost] = mapDetect( threshold, var, A, nIter, ...
    sigLen, p1, c10, c01)
%   [ Pd, Pf, ECost] = mapDetect( threshold, var, A, nIter, ...
    sigLen, p1, c10, c01, c00, c11)
%

p0 = 1-p1;
x = var*randn(nIter,sigLen);
r = rand(nIter,sigLen);
a = zeros(nIter,sigLen) + A*(r ≤ p1);
y = a + x;

H = zeros(nIter,sigLen);
H(y>threshold) =1;

detection = sum(H.*(a),2)./sum(a,2);
falarm = sum(H.*(¬a),2)./sum(¬a,2);

Pd = mean(detection);
Pf = mean(falarm);

if nargin > 6
    if nargin < 10
        c11 = 0;
    end
    if nargin < 9
        c00 = 0;
    end

    ECost = c00*p0 + c01*p1...
        + (c10-c00)*p0*Pf - (c01-c11)*p1*Pd;
end
```

```
end
```

5.5 mapDetect2

```
function [ Pd, Pf, ECost] = mapDetect2(var1, var0, A, nIter, ...
    sigLen, p1, threshold)
%MAPDETECT2 Computes the P[Detection], P[False Alarm], and the ...
    expected cost
%of a detector with a signal of power A and a specified ...
    threshold in a
%chanel with AWGN of variance var with a particular prior and ...
    cost
%structure, with the same mean and different variances
%
% Usage:
%    [ Pd, Pf, ECost] = mapDetect( var1, var0, A, nIter, ...
    sigLen, p1)
%

p0 = 1-p1;

y = zeros(nIter*sigLen,1);
r = rand(nIter*sigLen,1);
event = r ≤ p1;

x = var1*randn(sum(event),1);
z = var0*randn(sum(¬event),1);

y(event) = A + x;
```

```
y(¬event) = A + z;
y = (-(y-A).^2./(2*var1^2) + (y-A).^2./(2*var0^2));
LRT = log((p0*var1)/(p1*var0));

y = reshape(y,nIter,sigLen);
event = reshape(event,nIter,sigLen);

if nargin == 7
    LRT = threshold;
end



H = zeros(nIter,sigLen);
H(y>LRT) = 1;

detection = sum(H.*(event),2)./sum(event,2);
falarm = sum(H.*(¬event),2)./sum(¬event,2);

Pd = mean(detection);
Pf = mean(falarm);

end
```