University of Southern California

EE511 Simulation Methods for Stochastic Systems

Project #7- Expectation Maximization

BY

Mohan Krishna Thota

USC ID: 6683486728

mthota@usc.edu

PROBLEM 1:

DESCRIPTION:

- A multivariate normal distribution also known as multivariate gaussian distribution is nothing but a generalization of the univariate normal distribution to higher dimensions.
- One possible definition is that a random vector is said to be k-variate normally distributed if every linear combination of its k components has a univariate normal distribution.
- Its importance derives mainly from the multivariate central limit theorem.
- The multivariate normal distribution is often used to describe, at least approximately, any set of (possibly) correlated real-valued random variables each of which clusters around a mean value.
- A random vector is said to be multivariate normal distribution if it satisfies the following conditions
- Every linear combination of its components Y = a1X1 + … + ak. Xk is normally distributed.
- That is, for any constant vector a ∈ Rk, the random variable Y = aTx has a univariate normal distribution, where a univariate normal distribution with zero variance is a point mass on its mean.
- The covariance matrix is allowed to be singular (in which case the corresponding distribution has no density).
- This case arises frequently in statistics; for example, in the distribution of the vector of residuals in the ordinary least squares regression.
- Note also that the Xi are in general not independent; they can be seen as the result of applying the matrix A to a collection of independent Gaussian variables z.

PROCEDURE:

- RandNumber(N) function is used for generation of N samples.
- An array is initialized with N elements and sigma and mean values are given.
- A total of N samples are generated of X and the sample mean and covariance is calculated. A row vector Z of length and standard normal random variables are used to initialize Z for which normrnd is used.
- Matlab's chol() function is used to generate a lower triangular square matrix and column vector X is obtained and is of length 3.

CODE:

```
function rand_Num_Gen(Number)
Arr_X = zeros(3,Number);

for l = 1:Number
    myu = [1 2 3];
    sigma = [ 3 -1 1;
             -1 5 3;
              1 3 4 ];
    A = chol(sigma,'lower');
    B = normrnd(0,1,1,3);
    Cap_X = A*B.' + myu.';
    Cap_X = transpose(Cap_X);

    for b= 1:3
        Arr_X(b,l) = Cap_X(b);
    end
end
Array_X1 = zeros(1,Number);
Array_X2 = zeros(1,Number);
Array_X3 = zeros(1,Number);

for c = 1:Number
    Array_X1(1,c) = mean(Arr_X(1,c));
    Array_X2(1,c) = mean(Arr_X(2,c));
    Array_X3(1,c) = mean(Arr_X(3,c));
 end
display('mean of samples is:');
formatSpecMean = 'The mean of the samples is %.2f\n';
mean1 = mean(Array_X1);
fprintf(formatSpecMean,mean1);
mean2 = mean(Array_X2);
fprintf(formatSpecMean,mean2);
mean3 = mean(Array_X3);
fprintf(formatSpecMean,mean3);
covariance = cov(Arr_X.');
display('The covariance calculated from samples is:');
display(covariance);
end
```

```
OUTPUT
mean of samples is:
The mean of the samples is 1.11
The mean of the samples is 2.03
The mean of the samples is 3.10
The covariance calculated from samples is:

covariance =

    2.8699   -0.7207    1.1634
   -0.7207    4.6581    2.8175
    1.1634    2.8175    3.8968
```

```
mean of samples is:
The mean of the samples is 1.00
The mean of the samples is 2.00
The mean of the samples is 3.00
The covariance calculated from samples is:

covariance =

    2.9750   -0.9845    1.0034
   -0.9845    4.9923    3.0004
    1.0034    3.0004    4.0070
```

ANALYSIS:

- From the above we could see that, the mean and the covariance of the samples generated matches very closely with the specified mean and covariance.
- The margin of difference decreases when the number of samples is increased.

PROBLEM 2:

- A mixture distribution is the probability distribution of a random variable that is derived from a collection of other random variables as follows:
- first, a random variable is selected by chance from the collection according to given probabilities of selection, and then the value of the selected random variable is realized.
- The underlying random variables may be random real numbers, or they may be random vectors (each having the same dimension), in which case the mixture distribution is a multivariate distribution.
- In cases where each of the underlying random variables is continuous, the outcome variable will also be continuous and its probability density function is sometimes referred to as a mixture density.
- The cumulative distribution function (and the probability density function if it exists) can be expressed as a convex combination (i.e. a weighted sum, with non-negative weights that sum to 1) of other distribution functions and density functions.
- The individual distributions that are combined to form the mixture distribution are called the mixture components, and the probabilities (or weights) associated with each component are called the mixture weights.

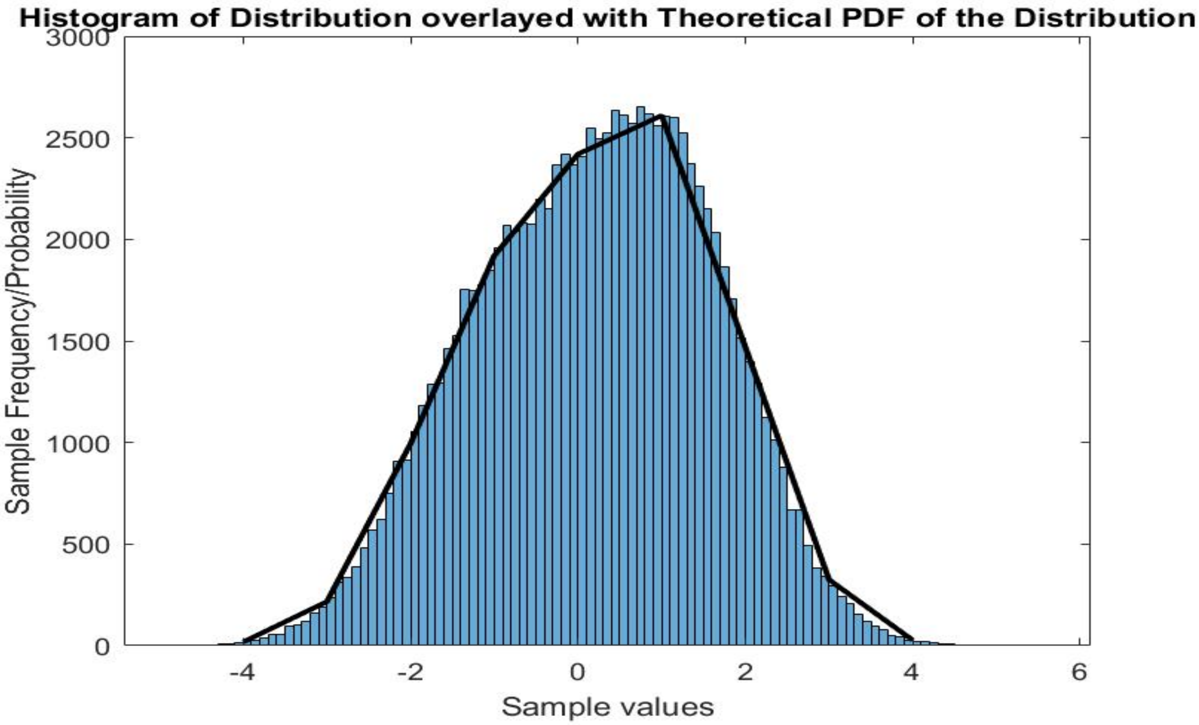
- The number of components in mixture distribution is often restricted to being finite, although in some cases the components may be countably infinite.
- More general cases (i.e. an uncountable set of component distributions), as well as the countable case, are treated under the title of compound distributions.

Methodology:

- A function MixedDistribution(N) is inherited for a total of N samples.
- An array is initialized which consists of a total of N elements when sigma and mean values are given.
- A total of N samples are generated from X and for the sample the sample mean and covariance values are calculated.
- Therefore, a row vector Z which of length # is generated and normrnd() function is used to initialize Z with standard normal random variables

CODE:

```
function MixedDistribution(N)
X = rand(N,1);
Y = zeros(N,1);
%%Distribution Generation
for i = 1:N
    if(X(i) <= 0.4)
        Y(i) = normrnd(-1,1);
    else
        Y(i) = normrnd(1,1);
    end
end

grid on;
histogram(Y);
x = -4:4;
hold on
%Histogram and Theoretical PDF
mixpdf = 10000*(0.4*normpdf(x,-1,1)+ 0.6*normpdf(x,1,1));
plot(x,mixpdf,'linewidth',2,'color','black');
title('Histogram of Distribution overlayed with Theoretical PDF of the
Distribution');
xlabel('Sample values');
ylabel('Sample Frequency/Probability');
end
```



ANALYSIS:

- From the above we could see that both the theoritoical pdf and histogram of X are overlaid well together.
- The quality of fit increases with increase in number of samples of X.

PROBLEM 3:

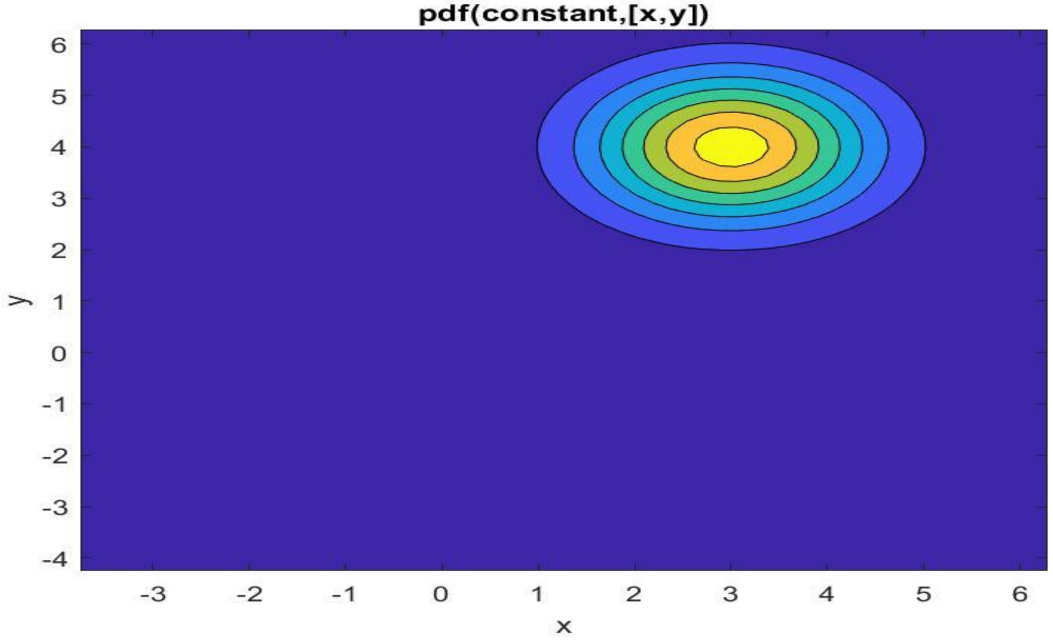
METHODOLOGY

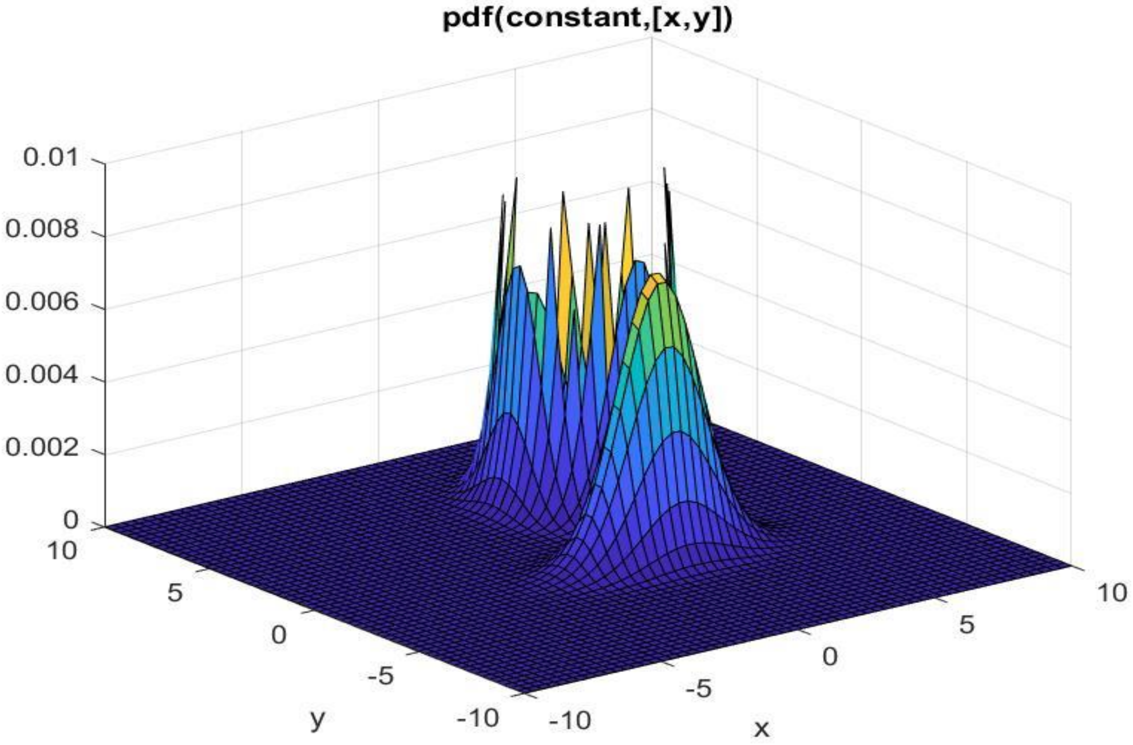
- A Gaussian Mixture Distribution object is created using gmdistribution() function which helps in creation of distribution with the given specifications.
- MATLAB's random() function is used for choosing of 300 samples randomly from distribution obj.
- Then gmdistribution.fit() function is used for the application of Expectation Maximization algorithm on the generated Gaussian Mixture Model.
- MATLAB's ezcontour() and ezsurf() functions are used to draw the contour and scatter plots.

CODE:

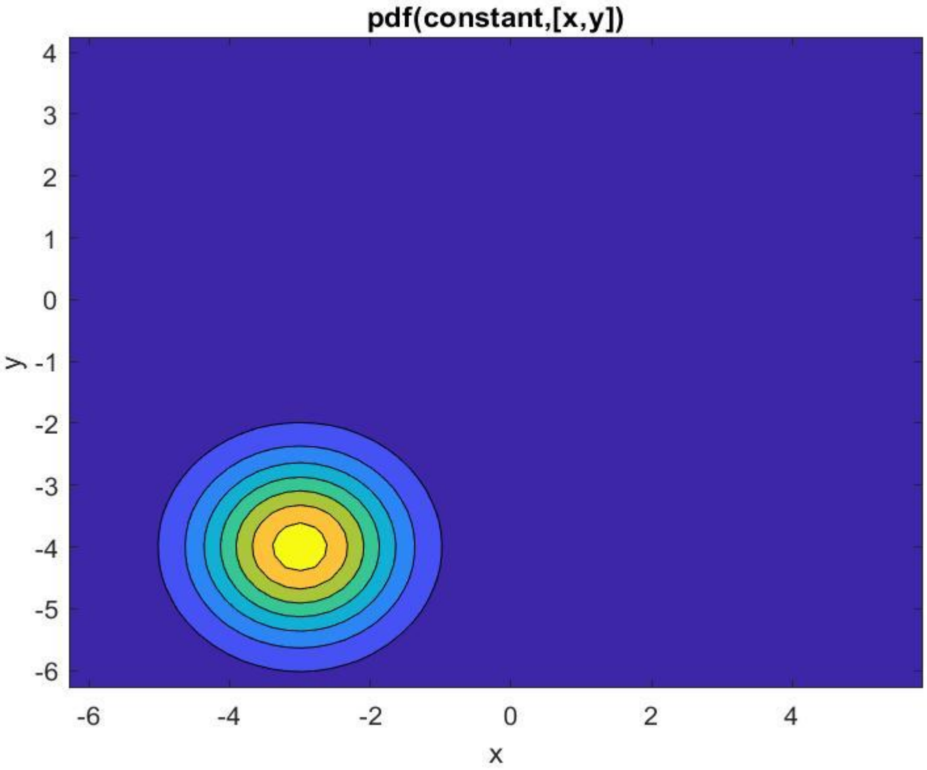
```
function Expectation_Maximization()
sigma = cat(3,[2 0;0 .5],[1 0;0 1]);
myu = [-1 2;-3 -4];
m = [0.5,0.5];

constant = gmdistribution(myu,sigma,m);
total_samples = random(constant,300);
options = statset('Display','final');
ExMax = gmdistribution.fit(total_samples, 2, 'options',options);

%% figure of 2D Projection;
figure;
%ezcontourf(@(x,y) pdf(constant,[x y]));

%% PDF surface figure;
ezsurf(@(x,y)pdf(constant,[x y]),[-10 10],[-10 10])

end
```

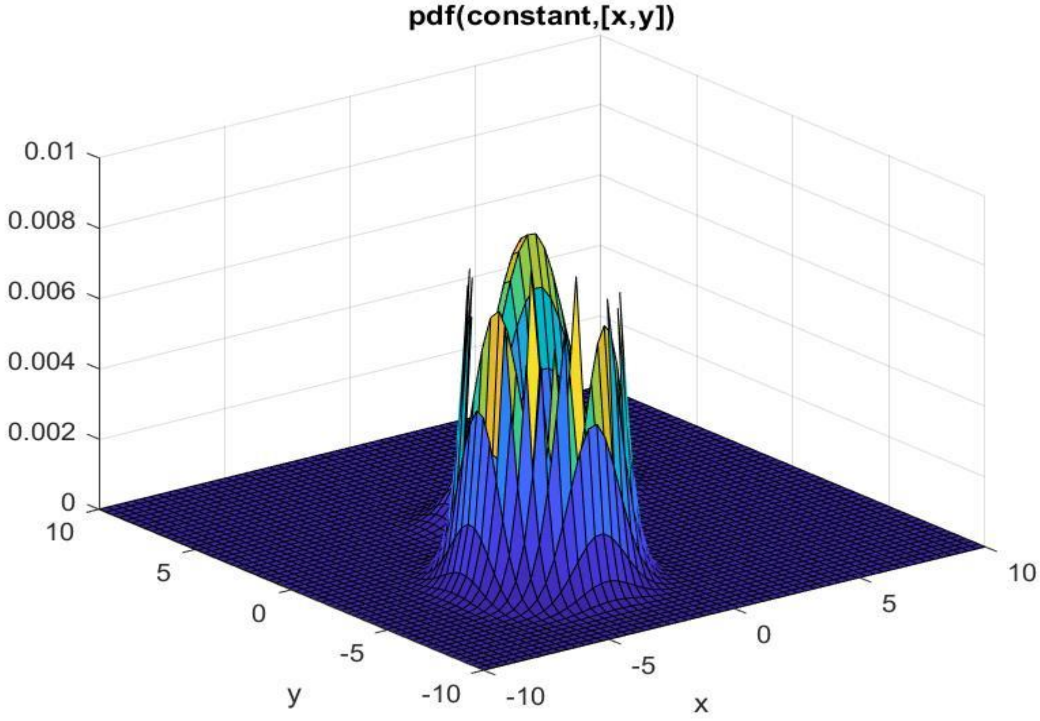
```
mu = [1 -2;3 4]; and a = [0.05,0.95];
```



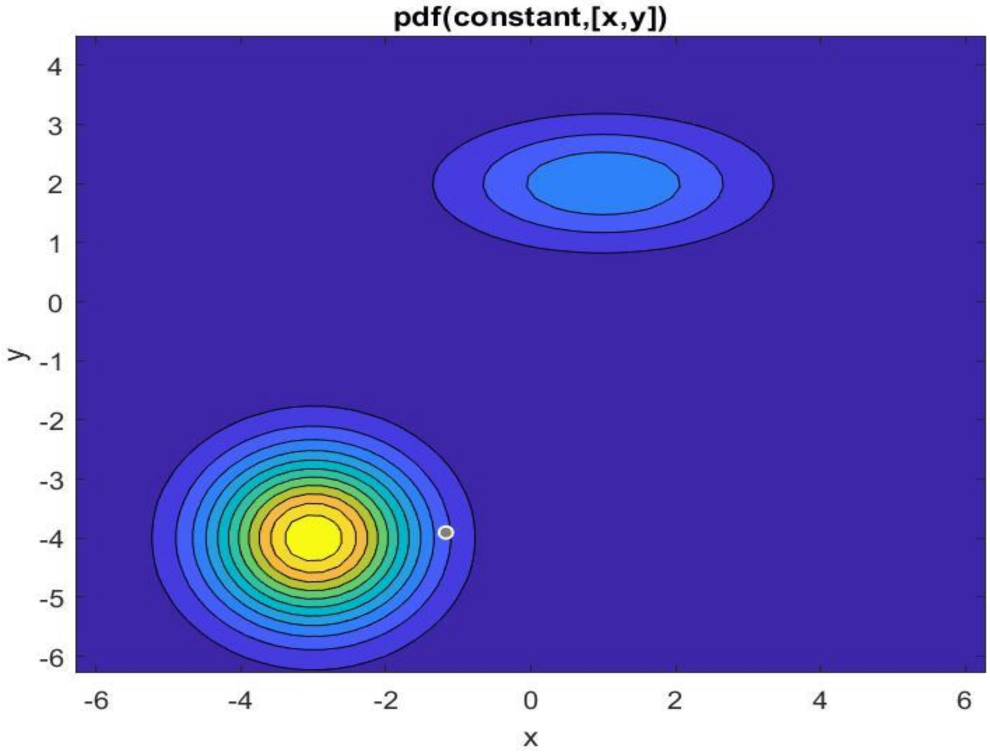


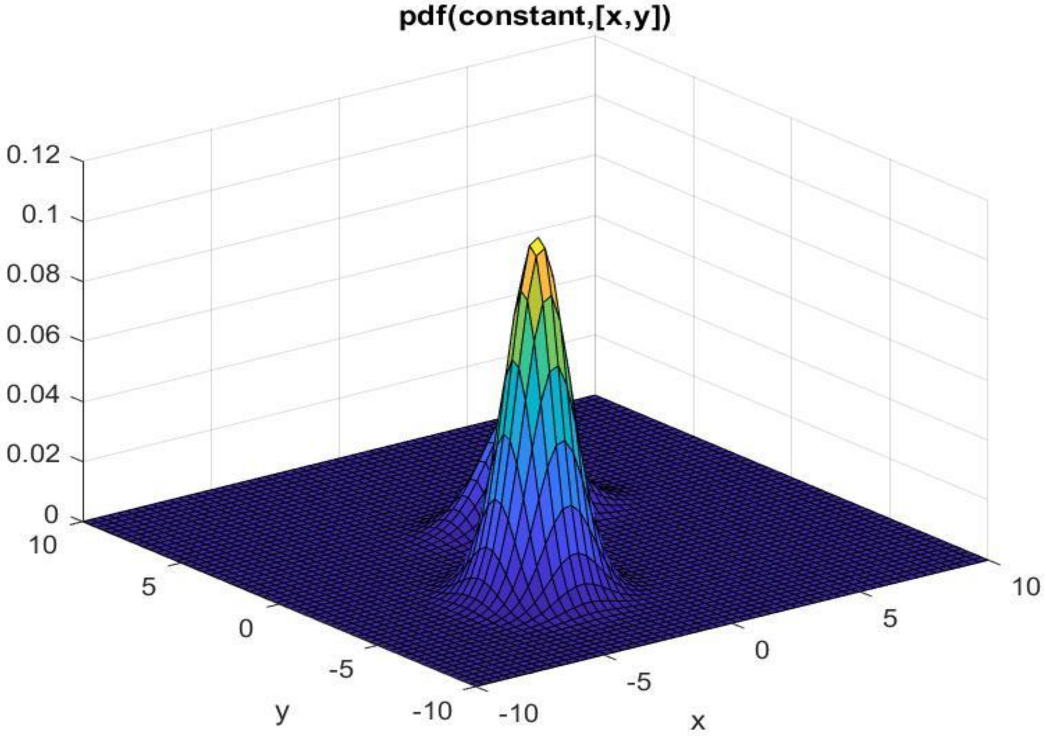
```
mu = [1 2;-3 -4]; and a= [0.05,0.95];
```



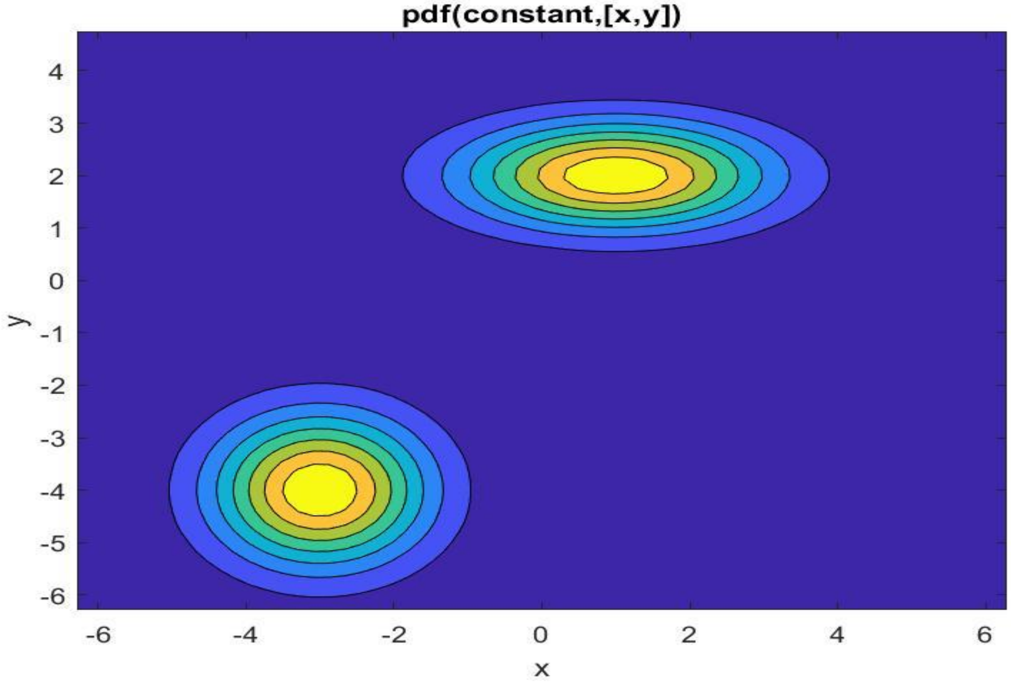


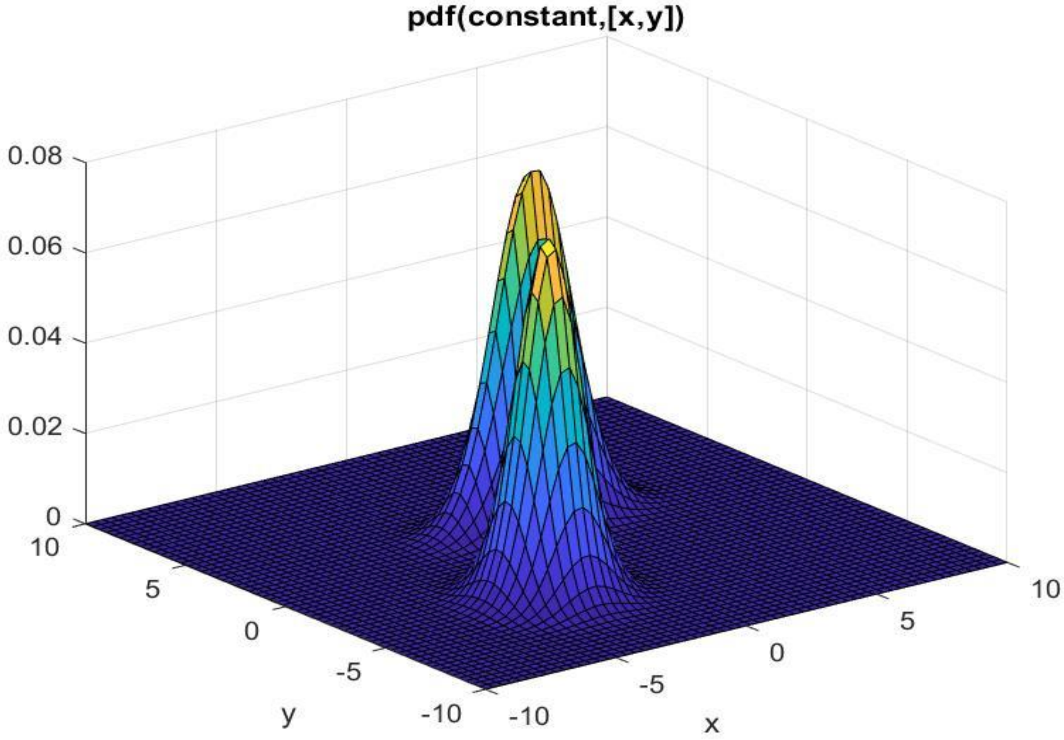
```
mu = [1 2; -3 -4]; and P = [0.25,0.75];
```



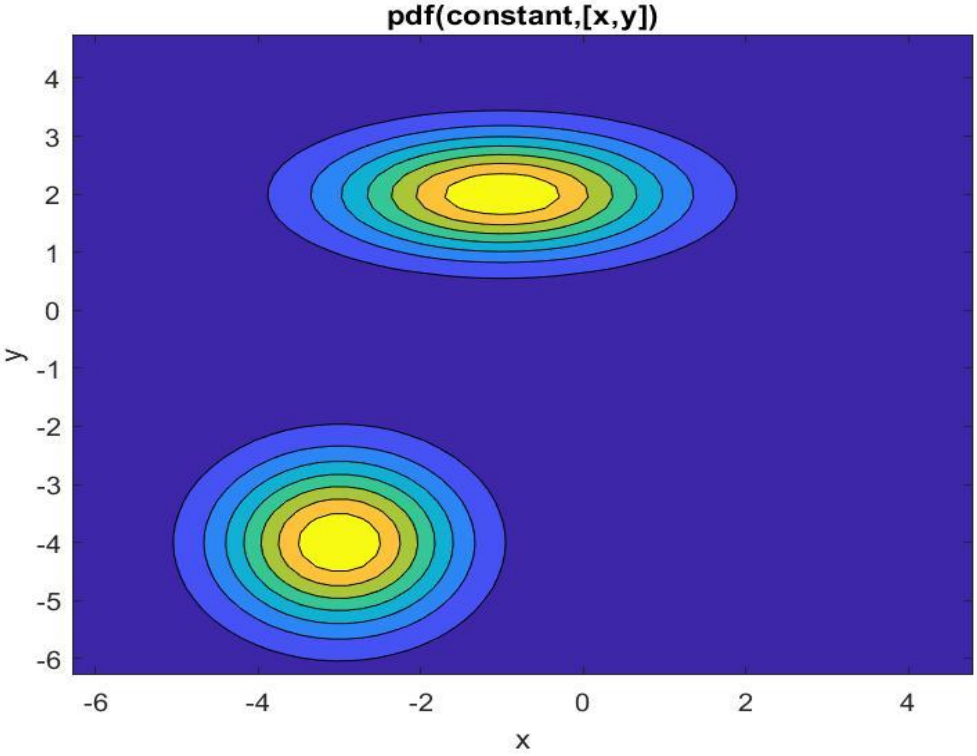


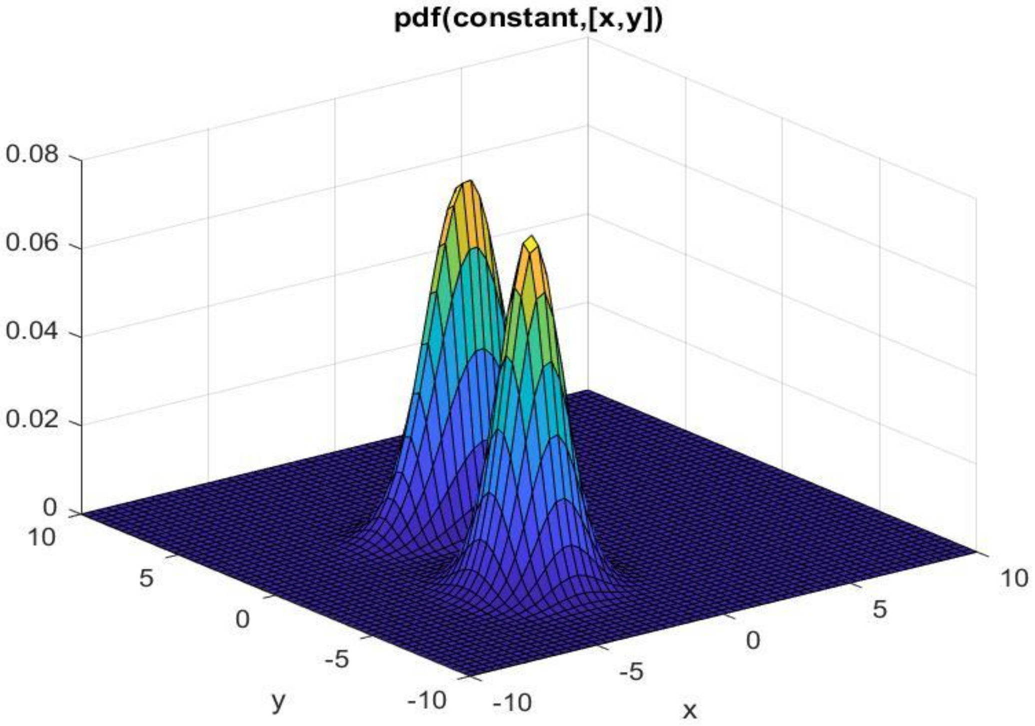
mu = [1 2;-3 -4]; and P = [0.5,0.5];





```
mu = [-1 2; -3 -4];
```


pdf(constant,[x,y])
4
3
2
1
0
-1
-2
-3
-4
-5
-6
y
-6
-4
-2
0
2
4
x


pdf(constant,[x,y])
0.08
0.06
0.04
0.02
0
10
5
0
-5
-10
y
-10
-5
0
5
10
x

Observation:

- From the above results, we could see that with the change in the value of μ, the proximity f changes the contour plots of the clusters to each other
- As the values of the weight's changes, the distribution is effected by each weight. As the values of the ∑ changes, the shapes of the contours also change.

PROBLEM 4:

DEFINTION:

- A mixture model is a probabilistic model for representing the presence of subpopulations within an overall population, without requiring that an observed data set should identify the sub-population to which an individual observation belongs.

METHODOLOGY:

- Here the kmeans() function is used to generate the clusters and they are plotted together by using hold function after the data collection from data list.
- Then Matlab's gmdistribution.fit() function is used to apply Expectation Maximization algorithm on Gaussian Mixture Model.
- Now ezcontour() and ezsurf() functions are used to draw contour and scatter plots.

CODE:

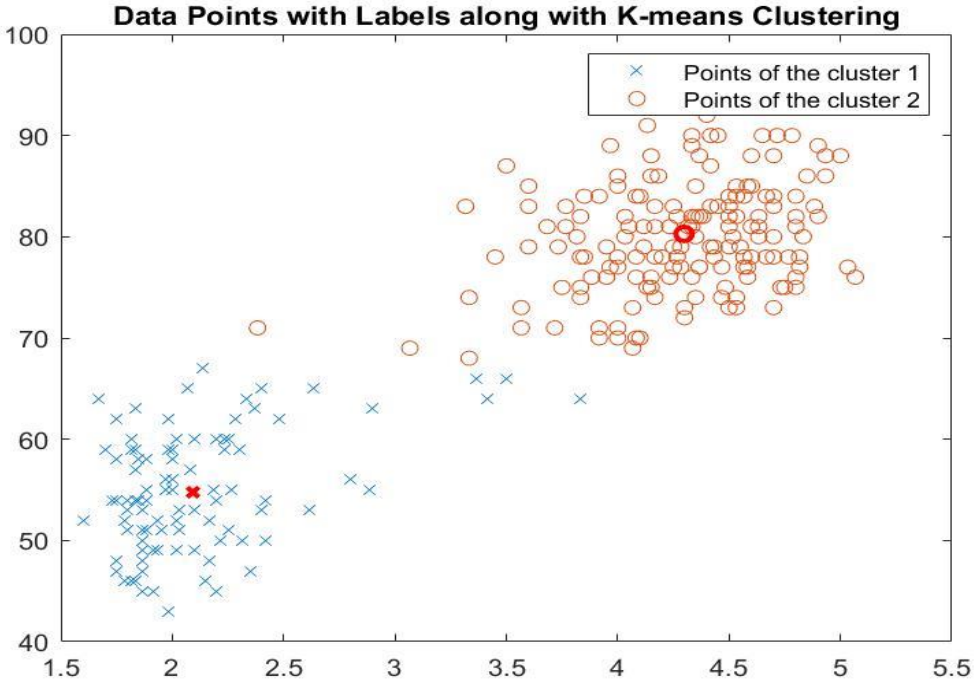
```
X = [ 3.600 79;
 1.800 54;
3.333 74;
2.283 62;
 4.533 85;
 2.883 55;
4.700 88;
3.600 85;
1.950 51;
4.350 85;
1.833 54;
3.917 84;
 4.200 78;
 1.750 47;
4.700 83;
2.167 52;
1.750 62;
4.800 84;
1.600 52;
4.250 79;
 1.800 51;
 1.750 47;
3.450 78;
3.067 69;
4.533 74;
3.600 83;
1.967 55;
4.083 76;
 3.850 78;
 4.433 79;
4.300 73;
4.467 77;
3.367 66;
4.033 80
3.833 74;
2.017 52;
 1.867 48;
 4.833 80;
1.833 59;
4.783 90;
4.350 80;
1.883 58;
4.567 84;
1.750 58;
 4.533 73;
 3.317 83;
3.833 64;
2.100 53;
4.633 82;
2.000 59;
4.800 75;
4.716 90;
 1.833 54;
 4.833 80;
1.733 54;
4.883 83;
3.717 71;
```

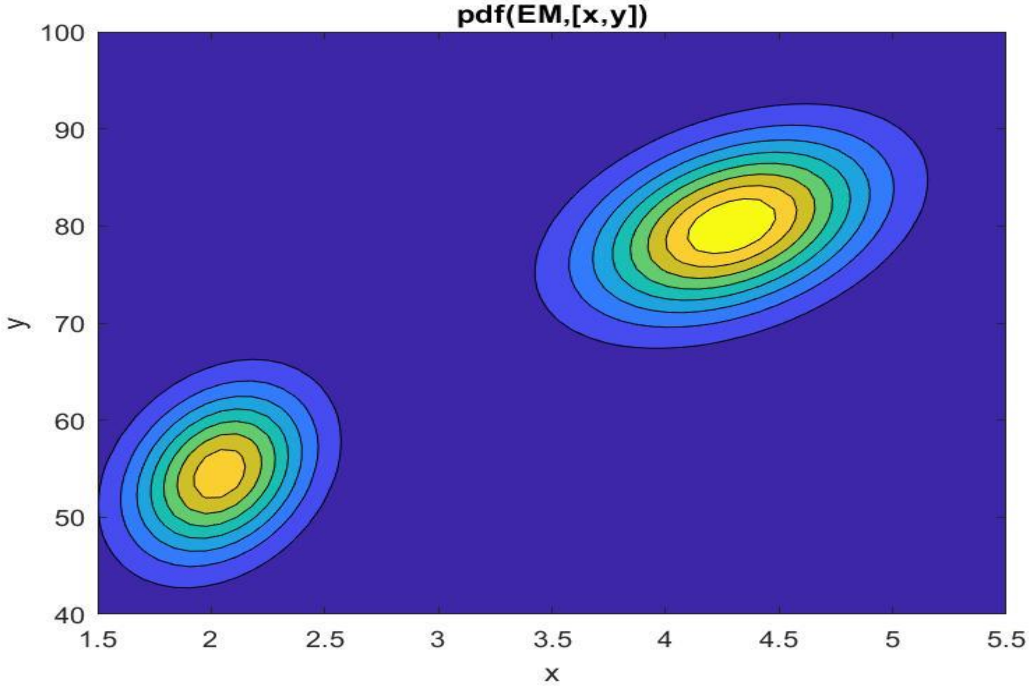
```
1.667 64;
4.567 77;
 4.317 81;
 2.233 59;
 4.500 84;
1.750 48;
4.800 82;
1.817 60;
4.400 92;
4.167 78;
4.700 78;
 2.067 65;
 4.700 73;
4.033 82;
1.967 56;
4.500 79;
4.000 71;
1.983 62;
5.067 76;
 2.017 60;
 4.567 78;
3.883 76;
3.600 83;
4.133 75;
4.333 82;
4.100 70;
2.633 65;
 4.067 73;
 4.933 88;
3.950 76;
4.517 80;
2.167 48;
4.000 86;
2.200 60;
4.333 90;
 1.867 50;
 4.817 78;
1.833 63;
4.300 72;
4.667 84;
3.750 75;
1.867 51;
4.900 82;
 2.483 62;
 4.367 88;
2.100 49;
4.500 83;
4.050 81;
1.867 47;
4.700 84;
1.783 52;
 4.850 86;
 3.683 81;
4.733 75;
2.300 59;
4.900 89;
4.417 79;
```

```
1.700 59;
4.633 81;
 2.317 50;
 4.600 85;
1.817 59;
4.417 87;
2.617 53;
4.067 69;
 4.250 77;
1.967 56;
 4.600 88;
 3.767 81;
1.917 45;
4.500 82;
2.267 55;
4.650 90;
1.867 45;
4.167 83;
 2.800 56;
 4.333 89;
1.833 46;
4.383 82;
1.883 51;
4.933 86;
2.033 53;
3.733 79;
 4.233 81;
 2.233 60;
4.533 82;
4.817 77;
4.333 76;
1.983 59;
4.633 80;
2.017 49;
 5.100 96;
 1.800 53;
5.033 77;
4.000 77;
2.400 65;
4.600 81;
3.567 71;
4.000 70;
 4.500 81;
 4.083 93;
1.800 53;
3.967 89;
2.200 45;
4.150 86;
2.000 58;
3.833 78;
 3.500 66;
 4.583 76;
2.367 63;
5.000 88;
1.933 52;
4.617 93;
1.917 49;
```

```
1.883 54;
1.850 54;
4.283 77;
3.950 79;
2.333 64;
4.150 75;
2.350 47;
4.933 86;
2.900 63;
4.583 85;
3.833 82;
2.083 57;
4.367 82;
 2.133 67;
4.350 74;
2.200 54;
4.450 83;
3.567 73;
4.500 73;
4.150 88;
3.817 80;
3.917 71;
4.450 83;
2.000 56;
4.283 79;
4.767 78;
4.533 84;
1.850 58;
4.250 83;
1.983 43;
2.250 60;
4.750 75;
4.117 81;
2.150 46;
4.417 90;
1.817 46;
4.467 74;];% kmeans and scatter plot

% To find the assignment y and the means C of each cluster
[kmeans1,kmeans2] = kmeans(X,2);

% Plotting of results
figure(2)
plot(X(kmeans1==1,1),X(kmeans1==1,2), 'x');
hold on

plot(X(kmeans1==2,1),X(kmeans1==2,2), 'o');
plot(kmeans2(1,1),kmeans2(1,2), 'rx','LineWidth',2);
plot(kmeans2(2,1),kmeans2(2,2), 'ro','LineWidth',2);

legend('Points of the cluster 1','Points of the cluster 2')
title('Data Points with Labels along with K-means Clustering')
hold off % GMM Distribution
EM = gmdistribution.fit(X,2); % 2D projection
figure;
ezcontourf(@(x,y) pdf(EM,[x y]),[1.5 5.5, 40 100]);
```

RESULT:





ANALYSIS:

- From the above figures we could see that the clusters are superated well.
- The scatter plots are also spherical which is clear from the above figures.