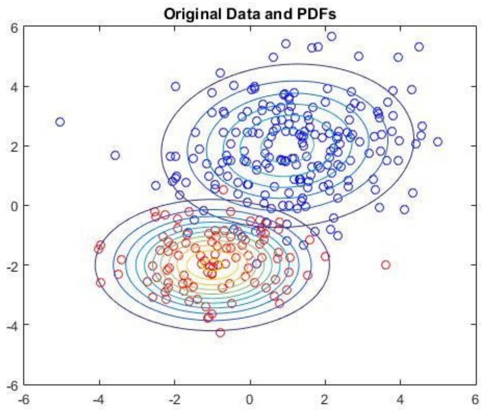
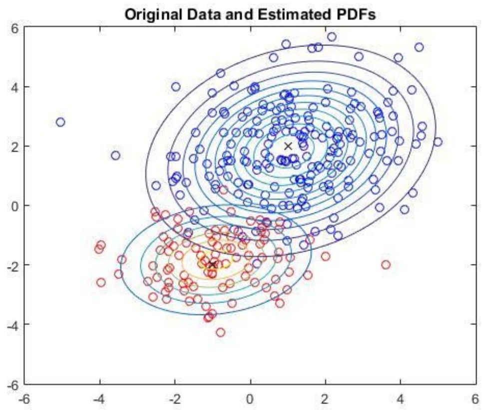
4105109 電腦視覺 (Computer Vision)

<u>Project #3 – Gaussian Mixture Model</u>

Deadline : 12/12 (Tue) 11:59 p.m.





Trace the matlab code of Gaussian Mixture Model. Write your report:

- Part 1: Trace the code to understand how EM algorithm works for GMM (GMM_Examples_v2014_08_04: GMMExample_2D.m). Get to know each parameter in this method and each variable in this code. Include the source code in your report. Write detailed program comments to explain **each line** (what it calculate? how it calculate?).
- Part 2: Write a report to explain Gaussian Mixture Model and how EM algorithm solves its parameters.

Note:

- Hand in the matlab code and the report to E-Course.
- 報告、程式註解請以中文撰寫，並繳交紙本報告一份
- Assignment format
  - Zip all your files into a single one and upload it to the E-Course website.
  - Please format the file name as:
    Student ID_proj2_verNo ex：
    602410143_proj2_v1

409510049 孫渝鈞 資工四

GMMExample_2D.m

```
%%=========================================================
%% STEP 1a: Generate data from two 2D distributions.


mu1 = [1 2];          %第一個分布的mean vector
sigma1 = [ 3 .2;      %第一個分布的convariance vector
          .2  2];
m1 = 200;             %第一個分布的data數量

mu2 = [-1 -2];        %第二個分布的mean vector
sigma2 = [2 0;        %第二個分布的convariance vector
          0 1];
m2 = 100;             %第二個分布的data數量

R1 = chol(sigma1);                       % Cholesky分解用於生成相應的data
X1 = randn(m1, 2) * R1;                  % 從標準正態分布生成data
X1 = X1 + repmat(mu1, size(X1, 1), 1);  % 將生成的data平移至目標分布

R2 = chol(sigma2);
X2 = randn(m2, 2) * R2;
X2 = X2 + repmat(mu2, size(X2, 1), 1);

X = [X1; X2];         % 將兩個分布的data合併為X

```

GMMExample_2D.m

```
%%=========================================================
%% STEP 1b: Plot the data points and their pdfs.

figure(1); % 繪製第一張圖

hold off;
plot(X1(:, 1), X1(:, 2), 'bo');    % 繪製第一個分布的data並將data標記為藍色圓點
hold on;
plot(X2(:, 1), X2(:, 2), 'ro');    % 繪製第二個分布的data並將data標記為紅色圓點

set(gcf,'color','white')           % 設置圖形背景為白色

gridSize = 100;                    % 定義網格大小, 即每個維度上的點數
u = linspace(-6, 6, gridSize);     % 生成一個包含100個在-6到6之間等間隔的點的數組
[A B] = meshgrid(u, u);            % 使用meshgrid函數生成二維網格, A和B是矩陣, 包含了u中對應x和y值的坐
gridX = [A(:), B(:)];              % 將A和B矩陣的列堆疊, 形成一個包含網格上所有點坐標的矩陣gridX

z1 = gaussianND(gridX, mu1, sigma1);    % 利用gaussianND function計算第一個分布的概率密度
z2 = gaussianND(gridX, mu2, sigma2);    % 利用gaussianND function計算第二個分布的概率密度

%將計算得到的概率密度z1, z2從一維數組轉換為二維矩陣Z1, Z2
%以便後續在圖形上進行等高線繪製
Z1 = reshape(z1, gridSize, gridSize);
Z2 = reshape(z2, gridSize, gridSize);

[C, h] = contour(u, u, Z1);        % 繪製第一個分布的概率密度等高線
[C, h] = contour(u, u, Z2);        % 繪製第二個分布的概率密度等高線

```

GMMExample_2D.m

```
axis([-6 6 -6 6])   % 設置坐標軸範圍
title('Original Data and PDFs');


%%=====================================================
%% STEP 2: Choose initial values for the parameters.

m = size(X, 1);

k = 2;  % GMM中的分量數
n = 2;  % data的維度

indeces = randperm(m);
mu = X(indeces(1:k), :);     % 從data中隨機選擇初始mean value

sigma = [];

for (j = 1 : k)
    sigma{j} = cov(X);       % 初始covariance matrix為data的協方差
end

phi = ones(1, k) * (1 / k); % 初始權重相等

%%====================================================
%% STEP 3: Run Expectation Maximization

W = zeros(m, k);     % 設置權重矩陣

```

GMMExample_2D.m

```
for (iter = 1:1000)

    fprintf('  EM Iteration %d\n', iter);

    %%=================================================
    %% STEP 3a: Expectation
    pdf = zeros(m, k);

    for (j = 1 : k)

        pdf(:, j) = gaussianND(X, mu(j, :), sigma{j});  % 計算每個data對每個分量的概率密度
    end

    pdf_w = bsxfun(@times, pdf, phi);    % 乘以權重


    W = bsxfun(@rdivide, pdf_w, sum(pdf_w, 2));     % 歸一化權重矩陣

    %%=================================================
    %% STEP 3b: Maximization
    %%
    %% Calculate the probability for each data point for each distribution.

    prevMu = mu;

    for (j = 1 : k)

        phi(j) = mean(W(:, j), 1);  % 更新權重，使用每個分布的樣本權重的平均值
```

GMMExample_2D.m

```
            mu(j, :) = weightedAverage(W(:, j), X); % 使用加權平均更新每個分布的mean value

            sigma_k = zeros(n, n);

            Xm = bsxfun(@minus, X, mu(j, :));

            for (i = 1 : m)
                sigma_k = sigma_k + (W(i, j) .* (Xm(i, :)' * Xm(i, :)));    % 使用加權更新每個分布
            end

            sigma{j} = sigma_k ./ sum(W(:, j));     % 歸一化得到每個分布的新協方差矩陣
        end

        if (mu == prevMu)   % 如果均值不再變化，結束EM算法的迭代
            break
        end

    end

    %%=====================================================
    %% STEP 4: Plot the data points and their estimated pdfs.

    figure(2);  % 繪製第二張圖
    hold off;
    plot(X1(:, 1), X1(:, 2), 'bo');
    hold on;
    plot(X2(:, 1), X2(:, 2), 'ro');

    set(gcf,'color','white')

    plot(mu1(1), mu1(2), 'kx');    % 在data散點圖上標記第一個高斯分布的mean value位置
    plot(mu2(1), mu2(2), 'kx');    % 在data散點圖上標記第二個高斯分布的mean value位置

    gridSize = 100;
    u = linspace(-6, 6, gridSize);
    [A B] = meshgrid(u, u);
    gridX = [A(:), B(:)];

    % 利用更新完的mean與covariance計算兩個分布的概率密度
    z1 = gaussianND(gridX, mu(1, :), sigma{1});
    z2 = gaussianND(gridX, mu(2, :), sigma{2});

    Z1 = reshape(z1, gridSize, gridSize);
    Z2 = reshape(z2, gridSize, gridSize);

    [C, h] = contour(u, u, Z1);     % 繪製更新後第一個分布的概率密度等高線
    [C, h] = contour(u, u, Z2);     % 繪製更新後第二個分布的概率密度等高線
    axis([-6 6 -6 6])

    title('Original Data and Estimated PDFs');
```

Part1:

在 GMMExample_2D.m 的 MATLAB code 實現了一個混合高斯模型(GMM)的期望最大化算法(EM)。

以下是每個部分的詳細解釋：

(Step 1a)生成數據：

mu1 跟 sigma1 定義了第一個二維高斯分佈的均值跟協方差矩陣

mu2 跟 sigma2 定義了第二個二維高斯分佈的均值跟協方差矩陣

m1 跟 m2 分別指定了每個分佈的數據點數量

在設定完所有數值後就通過 Cholesky 分解來生成符合相對應的高斯分佈的隨機數據，然後進行平移以匹配目標分佈。並且最後再將生成的數據合併成 X。

(Step 1b)繪製第一張數據跟機率分佈：

使用藍色與紅色來分別繪製剛剛生成的高斯分佈的數據點，並且使用 gaussianND function 來計算網格上的 probaility density，然後使用等高線繪製。

(Step 2)初始化參數：

隨機選擇初始的高斯分佈均值，並且選擇初始協方差矩陣作為協方差 [Cov(X)]，初始化權重 phi 被設定為相等。

(Step 3)執行 EM 算法：

(Step 3a)Expectation Step：

計算每個數據點屬於每個分布的機率，並將其歸一化得到權重矩陣 W。

(Step 3b)執行 EM 算法：

使用權重迭代更新 GMM 的參數，包含每個分布的均值 mu、協方差矩陣 sigma 和權重 phi。並且當更新後的均值不再變化就會停止。

(Step 4)繪製第二張估計的機率分佈：

在第二張圖上使用藍色和紅色分別繪製原始數據點，然後使用黑色交叉標記更新後的每個分布的均值位置。接著利用更新後的均值和協方差計算估計的概率密度，並使用等高線繪製。

Part 2:

GMM : 高斯混合模型是一種機率模型，通常用於描述包含多個高斯分布的複雜數據分布。GMM 的主要思想是，數據集中的每個觀察值都是由多個高斯分布中的一個生成的，而每個高斯分布對應一個特定的群體或模型組件。期望最大化算法是一種常用的方法，用於通過迭代優化模型參數，從而使模型更好地擬合數據。而 GMM 的數學表示如下：

$$p(x) = \sum_{k=1}^{k} \alpha_k N(\boldsymbol{x}|\mu_k, \Sigma_k)$$

其中α_k是 mixture component 的權重，$N(\boldsymbol{x}|\mu_k, \Sigma_k)$是高斯分佈的機率分佈函數，$\mu_k$和$\Sigma_k$分別是第 k 個高斯分佈的均值和協方差矩陣。

EM 算法：EM 算法是一種迭代優化算法，用於估計包含隱藏變量的機率模型參數。GMM 中的隱藏變量是每個數據點屬於每個高斯分布的機率。EM 算法包括兩個主要步驟：期望步驟(E step)和最大化步驟(M step)。

期望步驟(E step)：計算每個數據點屬於每個高斯分布的機率，得到權重矩陣 W。這一步是通過計算高斯分布的機率密度來實現的。

$$P(b|x,\mu_b,V_b)=\frac{\alpha_b P(x|\mu_b,V_b)}{\sum_{i=1}^{K}\alpha_i P(x|\mu_i,V_i)}$$

最大化步驟(M step)：使用權重矩陣 W 來更新模型參數，包括每個高斯分布的均值、協方差矩陣和 mixture component 的權重。會不斷的迭代更新這些步驟直到收斂，即模型參數不再發生顯著變化。

$$\mu_b^{new}=\frac{\sum_{i=1}^{N}x_i P(b|x_i,\mu_b,V_b)}{\sum_{i=1}^{N}P(b|x_i,\mu_b,V_b)}$$

$$V_b^{new}=\frac{\sum_{i=1}^{N}(x_i-\mu_b^{new})(x_i-\mu_b^{new})^T P(b|x_i,\mu_b,V_b)}{\sum_{i=1}^{N}P(b|x_i,\mu_b,V_b)}$$

$$\alpha_b^{new}=\frac{1}{N}\sum_{i=1}^{N}P(b|x_i,\mu_b,V_b)$$

結論：高斯混合模型通過 EM 算法提供了一種強大的建模方法，能夠適應複雜的數據分布。EM 算法的期望步驟和最大化步驟通過迭代優化模型參數，使得模型更好地擬合數據。這種靈活性使得 GMM 成為許多模式識別和聚類任務中的首選方法之一。