COMS4771 Machine Learning 2015: Homework 4 Solution

November 1, 2015

Problem 1

E-step

We write $\prod_{l=1}^{M} \mu_j(l)^{x_n(l)}$ as $\mu_j^{x_n}$.

$$\begin{aligned} \tau_{n,j} &= p(z_n = j | x_n, \theta) \\ &= \frac{\pi_j p(x_n|\mu_j)}{\sum_{i=1}^{K} \pi_i p(x_n|\mu_i)} \\ &= \frac{\pi_j \mu_j^{x_n}}{\sum_{i=1}^{K} \pi_i \mu_i^{x_n}} \end{aligned}$$

M-step

$$\theta = \arg\max_{\theta} \sum_{n=1}^{N} \sum_{j=1}^{K} \tau_{n,j} \log \frac{p(x_n, z = j|\theta)}{\tau_{n,j}}$$

As $p(x_n, z = j|\theta) = p(x_n|z = j, \theta)p(z_n = j|\theta) = \pi_j \mu_j^{x_n}$ and $\tau_{n,j}$ is a constant, we can write the expression above as:

$$\begin{aligned} \underset{\mu_1,..,\mu_K,\pi_1,..,\pi_K}{\text{argmax}} \quad & \sum_{n=1}^{N} \sum_{j=1}^{K} \tau_{n,j} (\log(\mu_j^{x_n}) + \log(\pi_j)) \\ \text{subject to} \quad & \sum_{l=1}^{M} \mu_j(l) = 1 \; \forall j \in \{1, .., K\}, \\ & \sum_{i=1}^{K} \pi_i = 1. \end{aligned}$$

Thus, we can write the corresponding Lagrangian:

$$\mathcal{L}(\mu, \pi, \alpha, \beta) = \sum_{n=1}^{N}\sum_{j=1}^{K} \tau_{n,j}[x_n \log(\mu_j) + \log(\pi_j)] - \alpha(\sum_{j=1}^{K} \pi_j - 1) - \sum_{j=1}^{K} \beta_j(\sum_{l=1}^{M} \mu_j(l) - 1)$$

We want to maximize this function along π_j:

$$\begin{aligned} & \frac{\partial \mathcal{L}(\mu, \pi, \alpha, \beta)}{\partial \pi_j} = 0 \\ \Leftrightarrow \quad & \sum_{n=1}^{N} \frac{\tau_{n,j}}{\pi_j} - \alpha = 0 \\ \Leftrightarrow \quad & \pi_j = \frac{\sum_{n=1}^{N} \tau_{n,j}}{\alpha} \end{aligned}$$

If we plug this into the primal constraint on π that is $\sum_{i=1}^{K} \pi_i = 1$, we get:

$$\pi_j = \frac{\sum_{n=1}^{N} \tau_{n,j}}{\sum_{i=1}^{K}\sum_{n=1}^{N} \tau_{n,i}} = \frac{1}{N}\sum_{n=1}^{N} \tau_{n,j}$$

We want to maximize the lagrangian along $\mu_j(l)$:

$$\begin{aligned} & \frac{\partial \mathcal{L}(\mu, \pi, \alpha, \beta)}{\partial \mu_j(l)} = 0 \\ \Leftrightarrow \quad & \sum_{n=1}^{N} \tau_{n,j} x_n(l) \frac{1}{\mu_j(l)} - \beta_j = 0 \\ \Leftrightarrow \quad & \mu_j(l) = \frac{1}{\beta_j}\sum_{n=1}^{N} \tau_{n,j} x_n(l) \end{aligned}$$

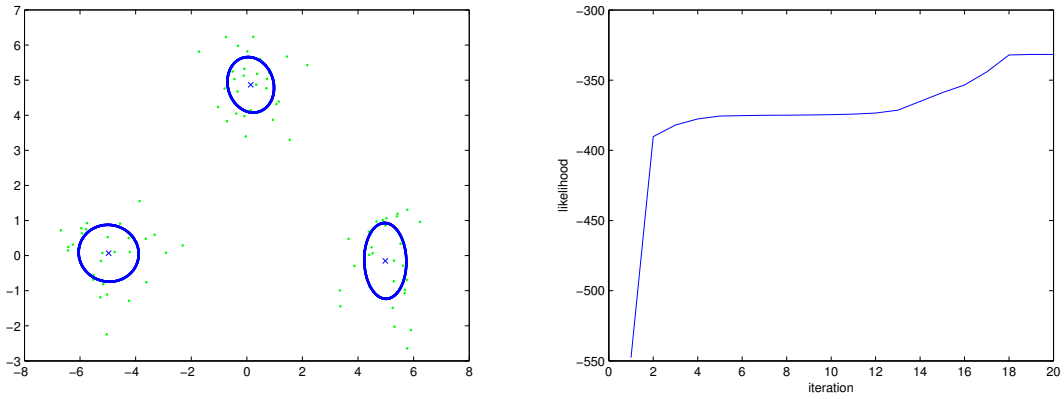
If we plug this into the constraint on μ which is $\sum_{l=1}^{M} \mu_j(l) = 1$ and given that $\sum_{l=1}^{M} x_n(l) = 1$

$$\mu_j(l) = \sum_{n=1}^{N} \frac{\tau_{n,j}}{\sum_{n'=1}^{N} \tau_{n',j}} x_n(l)$$

Problem 2

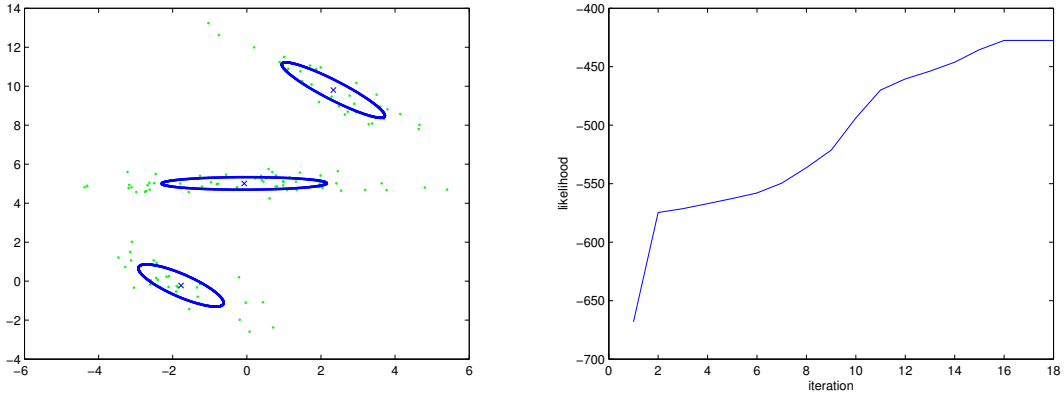
Part A:

The fitting results of EM-GMM on datasetA and datasetB are showed in Figure 1 and Figure 2.



(a) Fitting results (b) Iteration

Figure 1: GMM fitting on datasetA

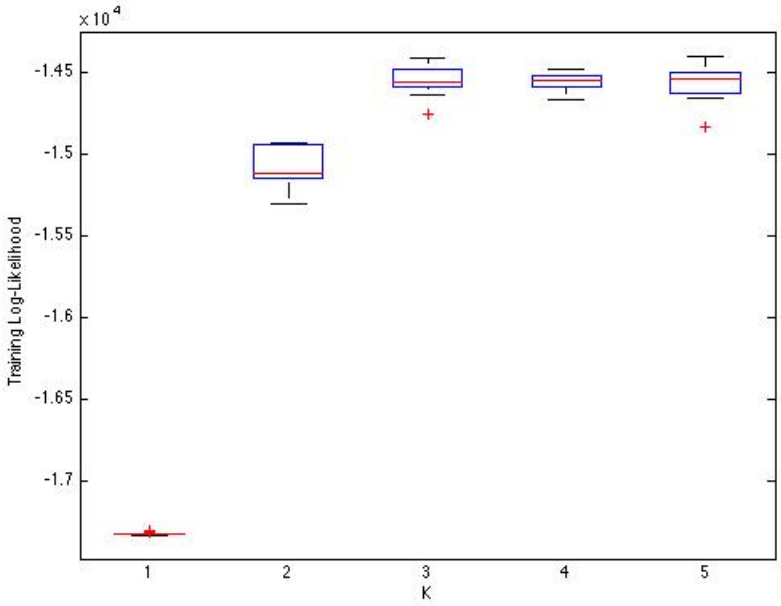


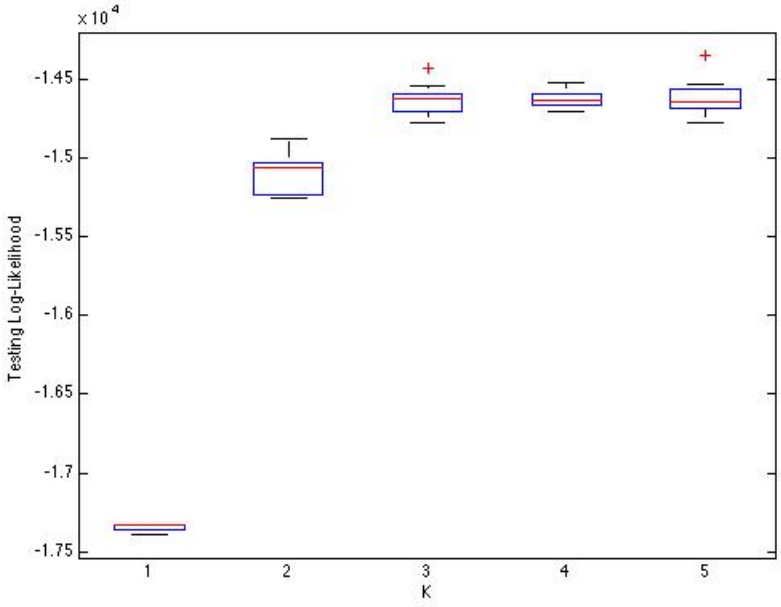
(a) Fitting results (b) Iteration

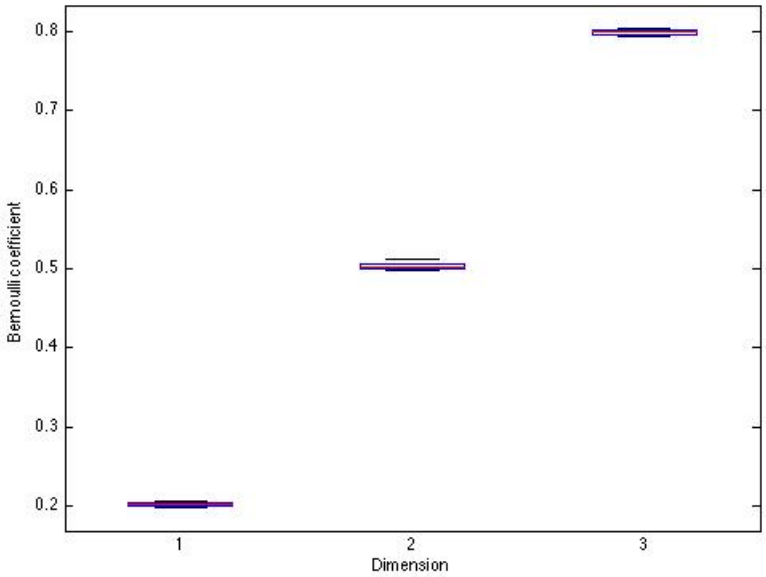
Figure 2: GMM fitting on datasetB

Part B:

The maximum average value of the testing log-likelihood is reached for K=3 (the splitting was made using half of the data for training and half of the data for testing) The Bernoulli coefficients are on average 0.2, 0.5 and 0.8.








```
function [training_1, testing_1, alpha] = EM(K)

% Load data
load problem2.mat
permutation = randperm(1000);
training_data = dataset(permutation(1:500),:);
testing_data = dataset(permutation(501:1000),:);

% Parameters definition
[N, T] = size(training_data);
tau = zeros(N, K);
alpha = zeros(K,1);
Pi = ones(K)/K;

% As the coin tosses are independent
% we can represent the data as a vector
% of N lines with the number of heads
X = (sum(training_data'))';
X_test = (sum(testing_data'))';

% Initialization
for k = 1:K
    alpha(k) = 0.5*k/K;
end

for step = 1:10
    % Expectation
    for n = 1:N
```

```
        S = 0;
        for l = 1:K
        S = S + Pi(l)*alpha(l)^X(n)*(1-alpha(l))^(T-X(n));
        end
        for k = 1:K
            tau(n,k) = Pi(k)*alpha(k)^X(n)*(1-alpha(k))^(T-X(n))/S;
        end
    end

    % Maximization
    for k = 1:K
        alpha(k) = sum(tau(:,k) .* X)/T/sum(tau(:,k));
    end

    Pi = sum(tau)/N;

    % Log likelihood estimation
    training_l = 0;
    for n = 1:N
        p = 0;
        for k = 1:K
            p = p + Pi(k)*alpha(k)^X(n)*(1-alpha(k))^(T-X(n));
        end
        training_l = training_l + log(p);
    end

    testing_l = 0;
    for n = 1:N
        p = 0;
        for k = 1:K
            p = p + Pi(k)*alpha(k)^X_test(n)*(1-alpha(k))^(T-X_test(n
               ));
        end
        testing_l = testing_l + log(p);
    end
    testing_l;
end
```

Problem 3

Figure 3: Original image

Figure 4: K = 3

Figure 5: K = 4

Figure 6: K = 5

As we randomly initialize the means, some means might end up being unused (i.e. the mean is not the closest mean to any point). This leads to NaN values when we compute the new mean.

As an example of a better way of initializing these centroids, we might make sure that no two initial means are too close by setting up a minimum distance between them.

We implemented a segmentation method based on RGB features, therefore our method is not spatially dependent leading to a fuzzy segmentation. To get better segmentation, we should consider spatial features.

```
% Load image
raw_im = Tiff('trees.tif','r');
im = raw_im.readRGBAImage();
im = im2double(im(1:200,1:200, :));
imshow(im)
[height, width, ~] = size(im);

% 1. Initialization of mu values
K = 3;
mu = zeros([K 3]);

for k = 1:K
    mu_h = randi([1 height]);
    mu_w = randi([1 width]);
    mu(k, :) = im(mu_h,mu_w, :);
end

% 2. Loop
Z = zeros(height, width, K);
n_iteration = 0;
while true
    % Compute new Z
    old_Z = Z;
    for n_h = 1:height
        for n_w = 1:width
            distance_to_mu = Inf;
            for k = 1:K
                if norm(mu(k, :) - squeeze(im(n_h, n_w, :))') <
                   distance_to_mu
                    % update minimum distance
                    distance_to_mu = norm(mu(k, :) - squeeze(im(n_h,
                       n_w, :))');
                    % clear old argmin
                    Z(n_h, n_w, :) = 0;
                    % update new argmin
                    Z(n_h, n_w, k) = 1;
                end
            end
        end
    end
    n_iteration = n_iteration + 1;
    if isequal(Z, old_Z);
        break
    end
    % Compute new mu
    for k = 1:K
        for channel = 1:3
```

```
            mu(k, channel) = (sum(sum(im(:, :, channel) .* Z(:,:,k)))
                )/sum(sum(Z(:, :, k)));
        end
    end
end

% 3. Create the segmented image

% If some means are not used they get NaN
% values when divided by zeros, we set these
% values to zeros
ind = find(isnan(mu));
n_unused_means = length(ind)/3;
mu(ind)= 0;

% Plotting the results
segmented_image = zeros(size(im));
for k = 1:K
    for channel = 1:3
        segmented_image(:, :, channel) = segmented_image(:, :,
            channel) + mu(k, channel) * Z(:,:,k);
    end
end

fprintf('Number of means: %d\n', K)
fprintf('Number of effective means: %d\n', K - n_unused_means)
fprintf('Number of iterations: %d\n', n_iteration)
imshow(segmented_image);
```

Problem 4

a)

Set $f(x) = \log(x)$, and the arithmetic mean of non-negative numbers is $\frac{\sum x_i}{n}$, the geometric mean is $\sqrt[n]{\prod x_i}$, by using Jensen's inequality, we have:

$$\begin{aligned} \log\left(\frac{\sum x_i}{n}\right) &\geqslant \frac{\sum \log(x_i)}{n} \\ \log\left(\frac{\sum x_i}{n}\right) &\geqslant \frac{1}{n}\log\left(\prod x_i\right) \\ \log\left(\frac{\sum x_i}{n}\right) &\geqslant \log\left(\prod x_i\right)^{\frac{1}{n}} \\ \log\left(\frac{\sum x_i}{n}\right) &\geqslant \log\left(\sqrt[n]{\prod x_i}\right) \end{aligned} \tag{1}$$

The previous formula establishes the familiar arithmetic mean-goometric mean inequality:

$$\frac{\sum x_i}{n} \geqslant \sqrt[n]{\prod x_i} \tag{2}$$

b)

We can represent the left part of the inequality as: $\sum_{i=1}^{m} \exp(\theta^T f_i) = \sum_{i=1}^{m} \frac{\alpha_i}{\alpha_i} \exp(\theta^T f_i)$, so that:

$$\begin{aligned} \sum_{i=1}^{m} \exp(\theta^T f_i) &= \sum_{i=1}^{m} \frac{\alpha_i}{\alpha_i} \exp(\theta^T f_i) \\ &= \sum_{i=1}^{m} \alpha_i \frac{\exp(\theta^T f_i)}{\alpha_i} \end{aligned} \tag{3}$$

Since $\alpha_i = \frac{\exp(\hat{\theta}^T f_i)}{\sum_{j=1}^{m} (\exp \hat{\theta}^T f_i)}$, so that $\sum \alpha_i = 1$. Then use Jensen's inequality and let $f(x) = \ln(x)$

$$\begin{aligned} f(E[x]) = \ln\left[\sum_{i=1}^{m} \alpha_i \frac{\exp(\theta^T f_i)}{\alpha_i}\right] &\geqslant E\left[f(x)\right] = \sum_{i=1}^{m} \alpha_i \ln\left[\frac{\exp(\theta^T f_i)}{\alpha_i}\right] \\ \ln\left[\sum_{i=1}^{m} \exp(\theta^T f_i)\right] &\geqslant \theta^T \sum_{i=1}^{m} \alpha_i f_i - \sum_{i=1}^{m} \alpha_i \ln \alpha_i \\ \ln\left[\sum_{i=1}^{m} \exp(\theta^T f_i)\right] &\geqslant \ln\left[\exp\left(\theta^T \sum_{i=1}^{m} \alpha_i f_i - \sum_{i=1}^{m} \alpha_i \ln \alpha_i\right)\right] \\ \sum_{i=1}^{m} \exp(\theta^T f_i) &\geqslant \exp\left(\theta^T \sum_{i=1}^{m} \alpha_i f_i - \sum_{i=1}^{m} \alpha_i \ln \alpha_i\right) \end{aligned} \tag{4}$$