STA243 Computational Statistics HW3

Due May 28, 2020

Spring 2020, UC Davis

Chenghan Sun (915030521)
e-mail: chesun@ucdavis.edu

Ran Sun (915458912)
e-mail: ransun@ucdavis.edu

Pledge:

Please sign below (print full name) after checking (✓) the following. If you can not honestly check each of these responses, please email me at kbala@ucdavis.edu to explain your situation.

- We pledge that we are honest students with academic integrity and we have not cheated on this homework. (✓)
- These answers are our own work. (✓)
- We did not give any other students assistance on this homework. (✓)
- We understand that to submit work that is not our own and pretend that it is our is a violation of the UC Davis code of conduct and will be reported to Student Judicial Affairs. (✓)
- We understand that suspected misconduct on this homework will be reported to the Office of Student Support and Judicial Affairs and, if established, will result in disciplinary sanctions up through Dismissal from the University and a grade penalty up to a grade of "F" for the course. (✓)

Team Member 1: *Chenghan Sun* Team Member 2: *Ran Sun*

1. Please download the MNIST handwritten digit dataset (training set images and training set labels) from `http://yann.lecun.com/exdb/mnist/`. It contains 28×28-pixel images for the hand-written digits {0,1,...,9} by storing each pixel value ranging between 0 and 255, and their corresponding true labels.

   Load the data into R. You are welcome to take a look at `https://cran.r-project.org/web/packages/readmnist/index.html` and `https://stackoverflow.com/questions/21521571/how-to-read-mnist-database-in-r` for how to do this task. Preprocess the data by compressing each image to 1/4 of the original size in the following way: Divide each 28×28 image into 2×2 non-overlapping blocks. Calculate the mean pixel value of each 2×2 block, and create a new 14×14 image. This preprocessing step will drastically help your computation. We will be clustering the digits $\{0, 1, 2, 3, 4\}$ in this homework. For visualization purpose, we view each data sample as 14×14 matrix. For using in an algorithm, treat each sample as a vector - you just simply stack each column of the 14×14 matrix into a 196 dimensional vector.

   The implementation of MNIST handwritten digit datasets pre-processing code could be found in "Problem 1" section of the submitted .R file named "CS_RS_code_hw3.R". The implementation guide is provided as below:

   Firstly, four datasets were read into the work space: training images, training labels, test images, and test labels. The loading of these datasets were implemented as the two documented hints given in problem statement; Secondly, the compression of training and test images is implemented as the function named "img_compress", which takes a argument "mat" as the data matrix to reduce each 28x28 image into 14x14 new image; Finally, since we only consider the clustering of digits {0, 1, 2, 3, 4}, these digits were extracted as working datasets. The final datasets were labeled as "train_data_comp_cluster", "train_labels_cluster", "test_data_comp_cluster", and "test_labels_cluster", respectively. A sample plot view of the processed 2x3 images was provided as Figure 1.

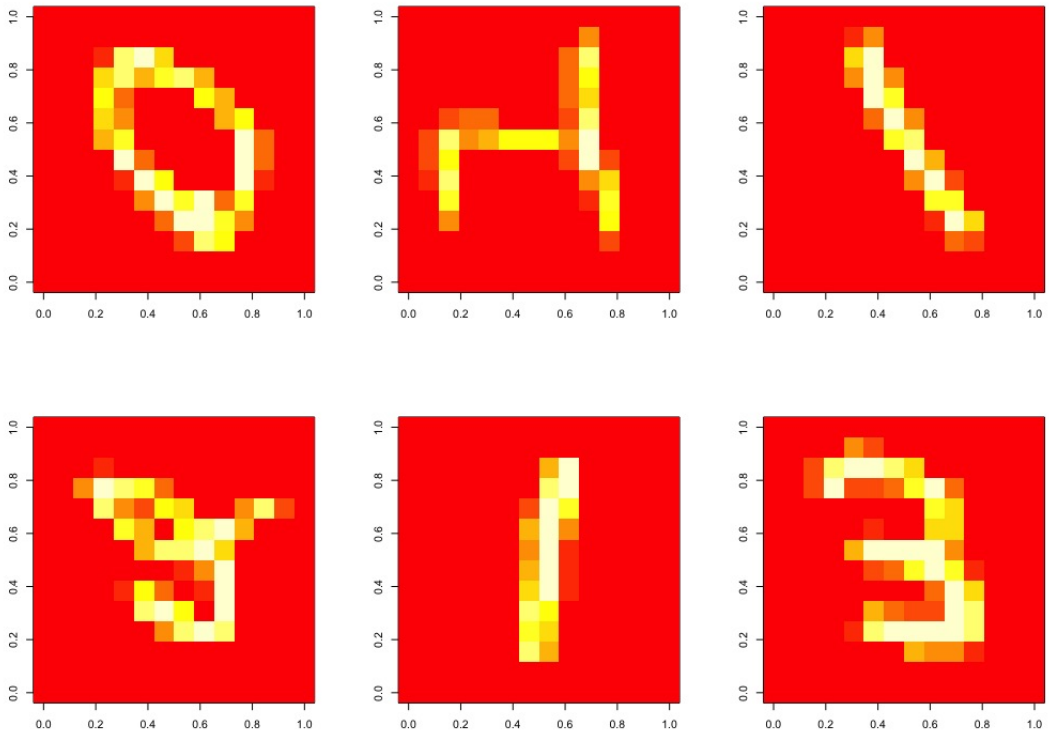

Figure 1: Example of compressed images with digits

2. (**10 Points**) Closely following the derivation in class, we now derive the EM algorithm for Gaussian mixture models. More specifically, we will use 2 models, the "mixture of spherical Gaussians" and the "mixture of diagonal Gaussians". By the end of this question, you should have derived two EM algorithms, one for each model. Below, the following denotes the meaning of each symbol:

   - each $\boldsymbol{\mu}_j$ is a d-dimensional vector representing the cluster center for cluster j
   - each $\boldsymbol{\Sigma}_j$ is a $d \times d$ matrix representing the covariance matrix for cluster j
   - $\sum_{j=1}^{k} \pi_j = 1$ and $\forall j, \pi_j \geq 0$ where all the π_j are the mixing coefficients.
   - Z_i represents the missing variable associated with $\boldsymbol{x}_i$ for $i \in \{1, \ldots, n\}$. It takes integer values $1, \ldots, k$.

   The parameters of the model to be estimated are $\boldsymbol{\theta} = (\{\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j, \pi_j\}_{j=1}^{k})$. To be more explicit, we will the notation $p(\boldsymbol{x}_i; \{\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j, \pi_j\}_{j=1}^{k})$ to denote the probability density function $p_{\boldsymbol{\theta}}(\boldsymbol{x}_i)$. That is, $p(\boldsymbol{x}_i; \{\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j, \pi_j\}_{j=1}^{k}) = p_{\boldsymbol{\theta}}(\boldsymbol{x}_i)$. The likelihood of the data for k clusters is:

$$
\begin{aligned}
L(\boldsymbol{\theta}) &= \prod_{i=1}^{n} p(\boldsymbol{x}_i; \{\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j, \pi_j\}_{j=1}^{k}) \qquad (1) \\
&= \prod_{i=1}^{n} \sum_{j=1}^{k} p(\boldsymbol{x}_i | Z_i = j; \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) p(Z_i = j) \\
&= \prod_{i=1}^{n} \sum_{j=1}^{k} \frac{\pi_j}{(2\pi)^{d/2} |\boldsymbol{\Sigma}_j|^{1/2}} \exp\left(-\frac{1}{2} (\boldsymbol{x}_i - \boldsymbol{\mu}_j)^T \boldsymbol{\Sigma}_j^{-1} (\boldsymbol{x}_i - \boldsymbol{\mu}_j) \right)
\end{aligned}
$$

   In a **mixture of diagonal Gaussians** model, $\boldsymbol{\Sigma}_j = \text{diag}(\sigma_{j1}^2, \ldots, \sigma_{jd}^2)$ is a diagonal matrix for all $j = 1, \ldots, k$. So, we only have d parameters to estimate for the covariance matrix of each cluster.

   In a **mixture of spherical Gaussians** model, $\boldsymbol{\Sigma}_j = \sigma_j^2 \boldsymbol{I}_d$ for all $j = 1, \ldots, k$. Here $\sigma_j > 0$ is a scalar and $\boldsymbol{I}_d$ is the $d \times d$ identity matrix. So we only have 1 parameter to estimate for the covariance matrix of each cluster.

   (i) First, we do some preliminary work that will be useful later on.

   - **Part a:** Write down the marginal distribution of the Z_i (Hint: What is the probability $p(Z_i = j)$).

     $$p(Z_i = j) = \pi_j, \; j = 1, 2, 3, \cdots, k.$$

   - **Part b:** Calculate $p(Z_i = j | \boldsymbol{x}_i)$. (Hint: Bayes Rule) By using Bayes Rule, we have

     $$
     \begin{aligned}
     p(Z_i = j | \boldsymbol{x}_i) &= \frac{p(\boldsymbol{x}_i | Z_i = j) p(Z_i = j)}{\sum_{l=1}^{K} p(\boldsymbol{x}_i | Z_i = l) p(Z_i = l)} \\
     &= \frac{p(\boldsymbol{x}_i | Z_i = j; \mu_j, \sigma_j) \pi_j}{\sum_{l=1}^{K} p(\boldsymbol{x}_i | Z_i = l; \mu_l, \sigma_l) \pi_l}
     \end{aligned}
     $$

Plug in the PDF of Gaussian distribution, we have

$$p(Z_i = j|\boldsymbol{x}_i) = \frac{\frac{1}{\sqrt{(2\pi)^d|\Sigma_j|}}\exp(-\frac{1}{2}(x_i-\mu_j)^\top {\Sigma_j}^{-1}(x_i-\mu_j))\pi_j}{\sum_{l=1}^{k}\frac{1}{\sqrt{(2\pi)^d|\Sigma_l|}}\exp(-\frac{1}{2}(\boldsymbol{x}_i-\mu_l)^\top \Sigma_l^{-1}(\boldsymbol{x}_i-\mu_l))\pi_l}$$

(ii). We now derive the E- and M- steps. We denote $\boldsymbol{\theta} := \{\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j, \pi_j\}_{j=1}^k$. From Equation 1, we can write down the log-likelihood and derive a lower bound:

$$\begin{aligned}
\ell(\boldsymbol{\theta}) &= \sum_{i=1}^{n} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}_i) \\
&= \sum_{i=1}^{n} \log \left[\sum_{j=1}^{k} p_{\boldsymbol{\theta}}(\boldsymbol{x}_i, Z_i = j)\right] \\
&= \sum_{i=1}^{n} \log \left[\sum_{j=1}^{k} \mathbf{F}_{ij} \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}_i, Z_i = j)}{\mathbf{F}_{ij}}\right],
\end{aligned}$$

where $F_{ij} > 0$ for all i, j and satisfies

$$\sum_{j=1}^{k} F_{ij} = 1 \quad for\ i = 1, \ldots, n.$$

Note that if $i = 1$, $\mathbf{F}_{1j} \in \mathbb{R}^k$ corresponds to the the distributions that we pick in the notes, denoted as q_1. Similarly for all i from 1 to n.

- **Part c:** Prove the following lower bound of the log-likelihood function:

$$\ell(\boldsymbol{\theta}) = \sum_{i=1}^{n} \log \left[\sum_{j=1}^{k} \mathbf{F}_{ij} \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}_i, Z_i = j)}{\mathbf{F}_{ij}}\right] \geq \sum_{i=1}^{n}\sum_{j=1}^{k} \mathbf{F}_{ij} \log \left[\frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}_i, Z_i = j)}{\mathbf{F}_{ij}}\right]. \tag{2}$$

  Hint: you can use the Jensen's inequality $\log \mathbb{E}X \geq \mathbb{E}\log X$ (You are not required to prove the Jensen's inequality).

$$\begin{aligned}
\ell(\theta) &= \sum_{i=1}^{n} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}_i) \\
&= \sum_{i=1}^{n} \log \sum_{Z_i} p_\theta(\boldsymbol{x}_i, Z_i) \\
&= \sum_{i=1}^{n} \log \sum_{j=1}^{K} F_{ij} \frac{p_\theta(\boldsymbol{x}_i, Z_i = j)}{F_{ij}} \\
&= \sum_{i=1}^{n} \log \mathbb{E}_{F_{ij}} \left(\frac{p_\theta(\boldsymbol{x}_i, Z_i)}{F_{ij}}\right)
\end{aligned}$$

By Jensen Inequality, we have

$$\sum_{i=1}^{n} \log \mathbb{E}_{F_{ij}} \left(\frac{p_\theta(\boldsymbol{x}_i, Z_i)}{F_{ij}} \right) \geq \sum_{i=1}^{n} \mathbb{E} \left(\log \frac{p_\theta(\boldsymbol{x}_i, Z_i)}{F_{ij}} \right)$$
$$= \sum_{i=1}^{n} \sum_{j=1}^{k} F_{ij} \log \frac{p_\theta(\boldsymbol{x}_i, Z_i = j)}{F_{ij}}$$
$$= \sum_{i=1}^{n} \sum_{j=1}^{k} F_{ij} \log \left(\frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}_i, Z_i = j)}{F_{ij}} \right)$$

Thus the lower bound is proved

$$\ell(\boldsymbol{\theta}) \geq \sum_{i=1}^{n} \sum_{j=1}^{k} F_{ij} \log \left(\frac{p_t(\boldsymbol{x}_i, Z_i = j)}{F_{ij}} \right).$$

- **Part d:** (E-Step) We define

$$Q(\mathbf{F}, \boldsymbol{\theta}) := \sum_{i=1}^{n} \sum_{j=1}^{k} \mathbf{F}_{ij} \log \left[\frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}_i, Z_i = j)}{\mathbf{F}_{ij}} \right] \tag{3}$$

to be the lower bound function of $\ell(\boldsymbol{\theta})$. Let $\boldsymbol{\theta}'$ be a fixed value of $\boldsymbol{\theta}$ (e.g., $\boldsymbol{\theta}'$ could be the parameter value of the current iteration). Recall from Part c that we designed $Q(\mathbf{F}, \boldsymbol{\theta})$ to be a lower bound for $\ell(\boldsymbol{\theta})$. We want to make this lower bound as tight as possible. Prove that $\ell(\boldsymbol{\theta}') = Q(\mathbf{F}, \boldsymbol{\theta}')$ when

$$\mathbf{F}_{ij} = p_{\boldsymbol{\theta}'}(Z_i = j | \boldsymbol{x}_i). \tag{4}$$

In previous section , we have the lower bound as

$$\ell(\theta) \geq Q(F, \theta) = \sum_{i=1}^{n} \sum_{j=1}^{K} F_{ij} \log \frac{p_\theta(\boldsymbol{x}_i, z_i = j)}{F_{ij}}$$

From Jensen Inequality, we have $\log \mathbb{E}X \geq \mathbb{E} \log X$, and the equity is obtained when $p(X = EX) = 1$. In our case, this condition becomes $\ell(\boldsymbol{\theta}') = Q(\mathbf{F}, \boldsymbol{\theta}')$. Therefore, we have

$$F_{ij} \propto p_{\theta'}(\boldsymbol{x}_i, Z_i = j)$$
$$\Rightarrow F_{ij} = \frac{p_{\theta'}(\boldsymbol{x}_i, Z_i = j)}{\sum_{Z_i} p_{\theta'}(\boldsymbol{x}_i, Z_i = j)}$$
$$= p_{\theta'}(Z_i = j | x_i).$$

Therefore, $\ell(\boldsymbol{\theta}') = Q(\mathbf{F}, \boldsymbol{\theta}')$ when $\mathbf{F}_{ij} = p_{\boldsymbol{\theta}'}(Z_i = j | \boldsymbol{x}_i)$

(iii). Once the E-step is derived, we now derive the M-step. First, we plug

$$\mathbf{F}_{ij} = p_{\boldsymbol{\theta}^{(t)}}(Z_i = j | \boldsymbol{x}_i)$$

into Equation 3 and define the lower bound function at $\boldsymbol{\theta}^{(t)}$ to be

$$Q(\boldsymbol{\theta}^{(t)}, \boldsymbol{\theta}) := \sum_{i=1}^{n} \sum_{j=1}^{k} p_{\boldsymbol{\theta}^{(t)}}(Z_i = j | \boldsymbol{x}_i) \log \left[\frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}_i, Z_i = j)}{p_{\boldsymbol{\theta}^{(t)}}(Z_i = j | \boldsymbol{x}_i)} \right]. \tag{5}$$

- **Part e** (M-step for mixture of spherical Gaussians) In the M-step, we aim to find $\boldsymbol{\theta}^{(t+1)}$ that maximize the lower bound function $Q(\boldsymbol{\theta}^{(t)}, \boldsymbol{\theta})$. Under the **mixture of spherical Gaussians** model, derive the M-step updating equations for $\boldsymbol{\mu}_j^{(t+1)}, \boldsymbol{\Sigma}_j^{(t+1)}$ and $\pi_j^{(t+1)}$.

  M-step

  The M-step can be written as

$$\theta^{(t+1)} = (\mu^{(t+1)}, \Sigma^{(t+1)}, \pi^{(t+1)}) = \underset{\mu,\Sigma,\pi}{\operatorname{argmax}} \quad Q((\mu, \Sigma, \pi), (\mu^{(t+1)}, \Sigma^{(t+1)}, \pi^{(t+1)}))$$

  From previous section, we know

$$Q(\theta^{(t)}, \theta) = \sum_{i=1}^{n} \sum_{j=1}^{k} p_{\theta^{(t)}}(Z_i = j|\boldsymbol{x}_i) \log \left[\frac{p_\theta(\boldsymbol{x}_i, Z_i = j)}{p_{\boldsymbol{\theta}^{(t)}}(Z_i = j|\boldsymbol{x}_i)} \right]$$

  First, we maximize over π, now we have

$$\begin{aligned} Q(\theta^{(t)}, \theta) &= \sum_{i=1}^{n} \sum_{j=1}^{k} (F_{ij}^{(t)} \log \pi_j + F_{ij}^{(t)} \log p_\theta(\boldsymbol{x}_i|Z_i = j)) \\ &= \sum_{i=1}^{n} \left(\sum_{j=1}^{k-1} F_{ij}^{(t)} \log \pi_j + F_{ik}^{(t)} \log \left(1 - \sum_{j=1}^{k-1} \pi_j \right) + \sum_{j=1}^{k} F_{ij}^{(t)} \log p_\theta \left(\boldsymbol{x}_i|Z_i = j \right) \right) \end{aligned}$$

  We have the partial derivative w.r.t. π_j as

$$\begin{aligned} \frac{\partial Q\left(\theta^{(t)}, \theta\right)}{\partial \pi_j} &= \sum_{i=1}^{n} \left(\frac{F_{ij}^{(t)}}{\pi_j} - \frac{F_{ik}^{(t)}}{1 - \sum_{j=1}^{k-1} \pi_j} \right) \\ &= \sum_{i=1}^{n} \left(\frac{F_{ij}^{(t)}}{\pi_j} - \frac{F_{ik}^{(t)}}{\pi_k} \right) \end{aligned}$$

  Set the partial derivative to 0 yields

$$\pi_j = \frac{\sum_{i=1}^{n} \pi_k F_{ij}^{(t)}}{\sum_{i=1}^{n} F_{ik}^{(t)}}$$

  Then we sum over j, and we know the fact that $\sum_{j=1}^{k} F_{ij} = 1$, thus we have

$$1 - \pi_k = \sum_{j=1}^{k-1} \pi_j$$

  Thus the update rule is

$$\pi_j^{(t+1)} = \frac{\sum_{i=1}^{n} F_{ij}^{(t)}}{n}$$

  Second, we maximize over μ and Σ. Similarly, we calculate the partial derivative of Q w.r.t. μ and Σ, respectively.

$$\frac{\partial Q\left(\theta^{(t)}, \theta\right)}{\partial \mu_{jk}} = \sum_{i=1}^{n} F_{ij}^{(t)} \frac{1}{\sigma_j^2} \left(x_{ik} - \mu_{jk} \right)$$

By setting the partial derivative to 0, we have

$$\mu_j^{(t+1)} = \frac{\sum_{i=1}^n F_{ij}^{(t)} \boldsymbol{x}_i}{\sum_{i=1}^n F_{ij}^{(t)}}$$

Similarly, the partial derivative of Q w.r.t Σ is

$$\frac{\partial \tilde{Q}\left(\theta^{(t)}, \theta\right)}{\partial \sigma_j^2} = \sum_{i=1}^n \sum_{k=1}^d F_{ij}^{(t)} \left(-\frac{1}{2\sigma_j^2} + \frac{1}{2\left(\sigma_j^2\right)^2} \left(x_{ik} - \mu_{jk}\right)^2 \right)$$

Thus we have

$$\begin{aligned}
\sigma_j^{2(t+1)} &= \frac{\sum_{i=1}^n \sum_{k=1}^d F_{ij}^{(t)} \left(x_{ik} - \mu_{jk}^{(t+1)}\right)^2}{\sum_{i=1}^n d F_{ij}^{(t)}} \\
&= \frac{\sum_{i=1}^n F_{ij}^{(t)} \left(\boldsymbol{x}_i - \mu_j^{(t+1)}\right)^\top \left(\boldsymbol{x}_i - \mu_j^{(t+1)}\right)}{d \sum_{i=1}^n F_{ij}^{(t)}}
\end{aligned}$$

To summarize, we update the parameters as

$$\begin{aligned}
\pi_j^{(t+1)} &= \frac{\sum_{i=1}^n F_{ij}^{(t)}}{n} \\
\mu_j^{(t+1)} &= \frac{\sum_{i=1}^n F_{ij}^{(t)} \boldsymbol{x}_i}{\sum_{i=1}^n F_{ij}^{(t)}} \\
\sigma_j^{2(t+1)} &= \frac{\sum_{i=1}^n F_{ij}^{(t)} \left(\boldsymbol{x}_i - \mu_j^{(t+1)}\right)^\top \left(\boldsymbol{x}_i - \mu_j^{(t+1)}\right)}{d \sum_{i=1}^n F_{ij}^{(t)}}
\end{aligned}$$

- **Part f** (M-step for mixture of diagonal Gaussians) Under the **mixture of diagonal Gaussians** model, derive the M-step updating equations for $\boldsymbol{\mu}_j^{(t+1)}, \boldsymbol{\Sigma}_j^{(t+1)}$ and $\pi_j^{(t+1)}$.

  M-step

  The derivation is similar to the previous question. The update rule for π is the same as

$$\pi_j^{(t+1)} = \frac{\sum_{i=1}^n F_{ij}^{(t)}}{n}$$

For the diagonal Gaussian case, we have

$$Q\left(\theta^{(t)}, \theta\right) = \sum_{i=1}^n \sum_{j=1}^K F_{ij}^{(t)} \log \pi_j - \sum_{i=1}^n \sum_{j=1}^K \sum_{k=1}^d F_{ij}^{(t)} \left(\frac{1}{2} \log\left(2\pi\sigma_{j_k}^2\right) + \frac{1}{2\sigma_{j_k}^2} \left(x_{ik} - \mu_{jk}\right)^2\right)$$

The partial derivative w.r.t μ yields

$$\frac{\partial \tilde{Q}\left(\theta^{(t)}, \theta\right)}{\partial \mu_{jk}} = \sum_{i=1}^n F_{ij}^{(t)} \frac{1}{\sigma_{j_k}^2} \left(x_{ik} - \mu_{jk}\right)$$

$$\mu_j^{(t+1)} = \frac{\sum_{i=1}^n F_{ij}^{(t)} \boldsymbol{x}_i}{\sum_{i=1}^n F_{ij}^{(t)}}$$

Similarly,

$$\frac{\partial Q\left(\theta^{(t)}, \theta\right)}{\partial \sigma_{j_k}^2} = \sum_{i=1}^n F_{ij}^{(t)} \left(-\frac{1}{2\sigma_{j_k}^2} + \frac{1}{2\left(\sigma_{j_k}^2\right)^2} \left(x_{ik} - \mu_{jk}\right)^2 \right)$$

And we have

$$\sigma_{j_k}^{2(t+1)} = \frac{\sum_{i=1}^n F_{ij}^{(t)} \left(x_{ik} - \mu_{jk}^{(t+1)}\right)^2}{\sum_{i=1}^n F_{ij}^{(t)}}$$

To summarize, the update rule can be written as

$$\begin{aligned}
\pi_j^{(t+1)} &= \frac{\sum_{i=1}^n F_{ij}^{(t)}}{n} \\
\mu_j^{(t+1)} &= \frac{\sum_{i=1}^n F_{ij}^{(t)} \boldsymbol{x}_i}{\sum_{i=1}^n F_{ij}^{(t)}} \\
\sigma_{j_k}^{2(t+1)} &= \frac{\sum_{i=1}^n F_{ij}^{(t)} \left(x_{ik} - \mu_{jk}^{(t+1)}\right)^2}{\sum_{i=1}^n F_{ij}^{(t)}}
\end{aligned}$$

3. EM algorithm to cluster the MNIST dataset.
(i) EM algorithm for mixture of spherical Gaussians. Assume 5 clusters. Terminate the algorithm when the fractional change of the log-likelihood goes under 0.0001. (Try 3 random initializations and present the best one in terms of maximizing the likelihood function)

Implementation details:

For this part, the EM algorithm for mixture of spherical Gaussians was implemented in "Problem 3" section (i) of the submitted .R file named "CS_RS_code_hw3.R". The implementation guide is provided as below:

First of all, we defined some helper functions before the main EM - spherical Gaussians algorithm:

1. The group of functions to initialize three necessary parameters π, $\mathbf{\Sigma}$, and μ for the Gaussian kernel were "init_para_pi", "init_para_mu", and "init_para_sigma" (Note: the detailed doc-strings could be found in each function in the .R code). Noted that the EM algorithm is sensitive to the initial value settings, parameter π's were randomly sampled from uniform distribution based on the number of clusters; then based on distribution of parameter π, the other two group of parameters $\mathbf{\Sigma}$'s and μ's were simulated as the corresponding sample means and sample variances. This mindful initialization procedure effectively prevented potential drawbacks of numerical overflow issues during the simulation section, as provided in hint (3) of HW statement.

2. Based on the hints, we then constructed a helper function to compute log-likelihood by matrix F_{ij}. For this mixture of spherical Gaussians case, function named "sphe_LL_constructor" was designed (Note: the detailed doc-strings could be found inside this function in the .R code). The three parameters were given as arguments to the function to construct matrix F_{ij}; Then the log-sum-exp trick is applied sequentially for computing the log-likelihood to prevent numerical underflow as mentioned in hint (2) of HW statement. This function returns two variables: the matrix F_{ij} and the corresponding log-likelihood.

3. Based on the results of Problem 2 Part (e) M-step for mixture of spherical Gaussians, we again constructed three update functions for parameters π, $\mathbf{\Sigma}$, and μ. Those functions were named as "update_para_pi", "update_para_mu", and "update_para_sigma" (Note: the detailed doc-strings could be found in each function in the .R code). Please re-visit our answer for Part (e) in Problem 2 of this report for detailed equations to these three updating functions.

4. After all the preparations, the main function for EM - mixture of spherical Gaussians algorithm is implemented as function "EM_sphe_gaus" (Note: the detailed doc-strings could be found inside this function in the .R code). The fractional tolerance of the log-likelihood was set to 0.0001, and a maximum iterations number as set to 500. This function could be treated as an ensemble function for all the previous helper functions, and the E-step and M-step was performed iteratively to maximize the log-likelihood. The parameters of the last iteration and corresponding matrix F_{ij} and log-likelihood would be returned from this function for error rate calculations.

5. In order to make use of the true labels and calculate the error rate of the algorithm, a function named "EM_sphe_labels_pred" was implemented to firstly compute the predicted labels of each cluster based on the trained EM - mixture of spherical Gaussians model (Note: the detailed doc-strings could be found inside this function in the .R code). Eventually, a function named

"EM_pred_error" was implemented to calculate the error rate on both the training and test datasets.

6. As a last component of the code in Part (i), a function named "EM_cluster_plot" was implemented to visualize the fitted clustering results (Note: the detailed doc-strings could be found inside this function in the .R code).

Experiments details:

As required in the problem statement, we assumed 5 clusters, and conduct 3 random initializations and present the best one in terms of maximizing the likelihood function. The results of the three experiments were provided as Table 1 as below, which included the simulated number of iterations, the final value of log-likelihood, the error rate on both training and test sets.

Table 1: Three random EM experiments for mixture of spherical Gaussians

Experiments	**Iterations**	**Log-likelihood**	**Training Error Rate**	**Test Error Rate**
1	8	-31830018.651	0.124624	0.116170
2	9	-31832119.999	0.131390	0.125900
3	7	-31903582.857	0.263368	0.262502

From the results above, since experiment 1 achieved maximum log-likelihood, we recognized this case as the best performance of EM for mixture of spherical Gaussians algorithm. The signified test error rate for the best experiment = 0.116170, a very competitive result.

In addition, the corresponding fitted clustering labels images were provided as Figure 2 - 4 as below:

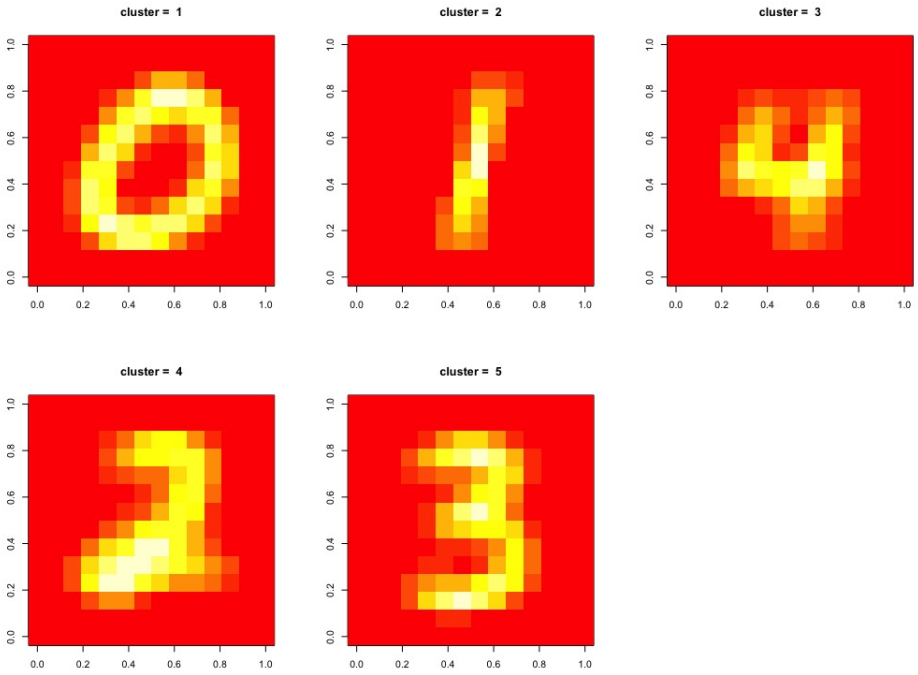

Figure 2: EM for mixture of spherical Gaussians - experiment 1

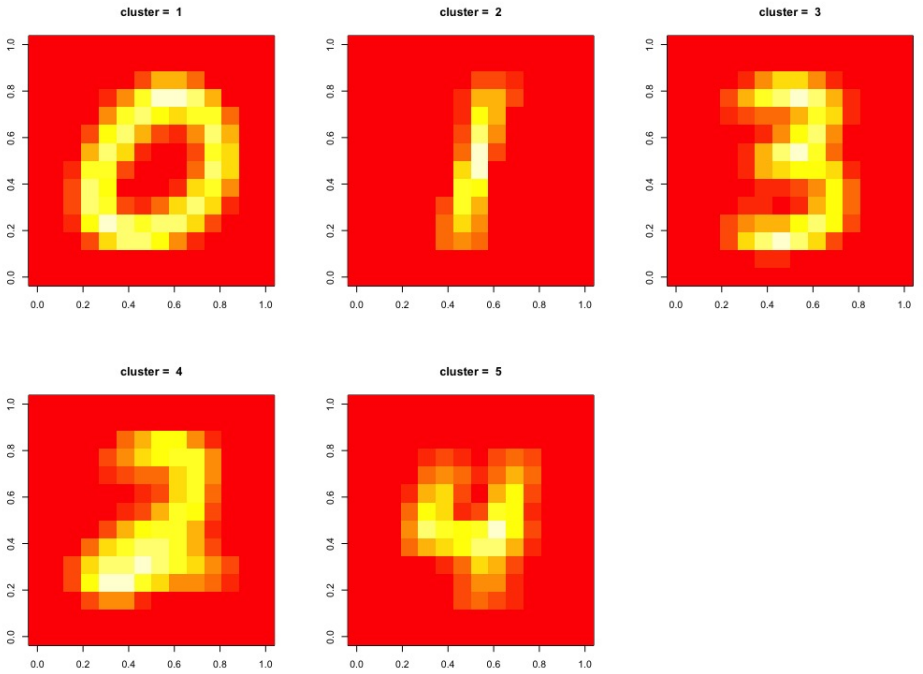

Figure 3: EM for mixture of spherical Gaussians - experiment 2

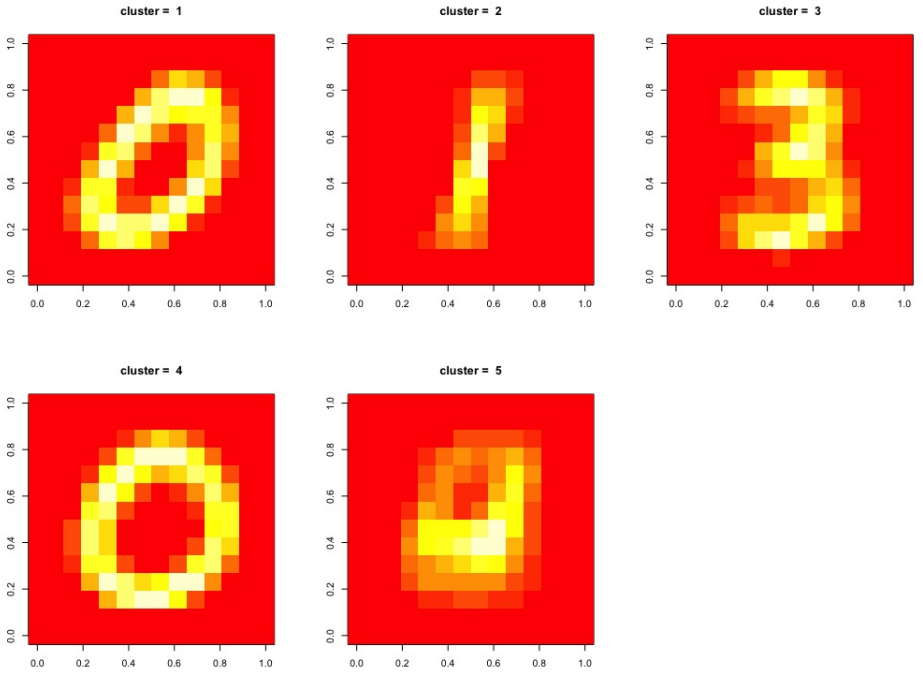

Figure 4: EM for mixture of spherical Gaussians - experiment 3

From the figures above, we concluded that experiments 1 and 2 generally captured and recovered the five digits; for experiment 3, since the test error rate was relatively higher than the first two experiments, we saw the digits 2 and 4 were kind of miss-clustered. As we mentioned,

the Figure 2 associated with the best experiment 2 represented the effectiveness of the mixture of spherical Gaussians model. Thus in conclusion, the EM for mixture of spherical Gaussians algorithm is still promising for the clustering task, by given more strict convergence tolerance.

(ii) EM algorithm for mixture of diagonal Gaussians.

Assume 5 clusters. Terminate the algorithm when the fractional change of the log-likelihood goes under 0.0001. (Try 3 random initializations and present the best one in terms of maximizing the likelihood function)

Implementation details:

For this part, the EM algorithm for mixture of diagonal Gaussians was implemented in "Problem 3" section (ii) of the submitted .R file named "CS_RS_code_hw3.R". The implementation guide is provided as below:

First of all, most of the helper functions we introduced in Part (i) could be employed in this Part (ii). We only need to introduce three different functions for the EM - diagonal Gaussians algorithm:

1. For this mixture of diagonal Gaussians case, similarly to the previous introduced function "sphe_LL_constructor", a function named "diag_LL_constructor" was designed (Note: the detailed doc-strings could be found inside this function in the .R code). The only difference is that we replaced the mixture of spherical Gaussians assumption by mixture of diagonal Gaussians assumption. And based on hint (2) in HW statement, we added 0.05 to all the diagonal elements for this mixture of diagonal Gaussians case. Again, the three parameters were given as arguments to the function to construct matrix F_{ij}; Then the log-sum-exp trick is applied sequentially for computing the log-likelihood to prevent numerical underflow. This function returns two variables: the matrix F_{ij} and the corresponding log-likelihood.

2. The main function for EM - mixture of diagonal Gaussians algorithm is implemented as function "EM_diag_gaus" (Note: the detailed doc-strings could be found inside this function in the .R code). Same as before, the fractional tolerance of the log-likelihood was set to 0.0001, and a maximum iterations number as set to 500. The parameters of the last iteration and corresponding matrix F_{ij} and log-likelihood would be returned from this function for error rate calculations.

3. In order to make use of the true labels and calculate the error rate of the algorithm, a function named "EM_diag_labels_pred" was implemented to firstly compute the predicted labels of each cluster based on the trained EM - mixture of diagonal Gaussians model (Note: the detailed doc-strings could be found inside this function in the .R code).

Experiments details:

As required in the problem statement, we assumed 5 clusters, and conduct 3 random initializations and present the best one in terms of maximizing the likelihood function. The results of the three experiments were provided as Table 2 as below, which included the simulated number of iterations, the final value of log-likelihood, the error rate on both training and test sets.

Table 2: Three random EM experiments for mixture of diagonal Gaussians

Experiments	**Iterations**	**Log-likelihood**	**Training Error rate**	**Test Error Rate**
1	77	-16822325.483	0.542195	0.536875
2	27	-17415377.553	0.514577	0.519556
3	28	-17095928.470	0.579945	0.574042

From the results above, since experiment 1 achieved maximum log-likelihood, we recognized this case as the best performance of EM for mixture of diagonal Gaussians algorithm. The signified test error rate for the best experiment = 0.536875, a relatively poor result compared with the EM for mixture of spherical Gaussians algorithm as more than 50% of labels were mis-clustered.

In addition, the corresponding fitted clustering labels images were provided as Figure 5 - 7 as below: bels images were provided as Figure 2 - 4 as below:

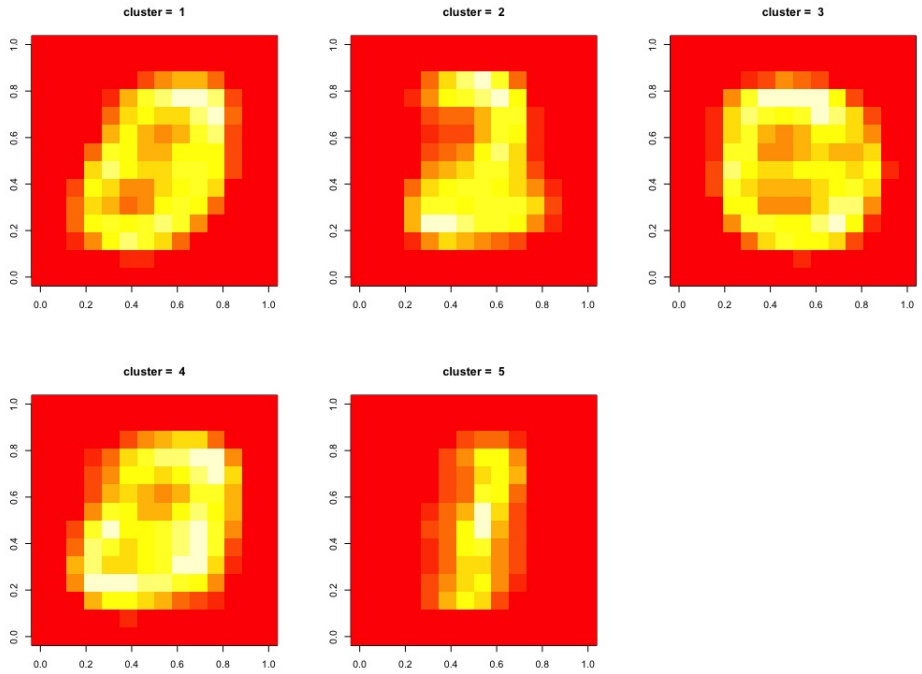

Figure 5: EM for mixture of diagonal Gaussians - experiment 1

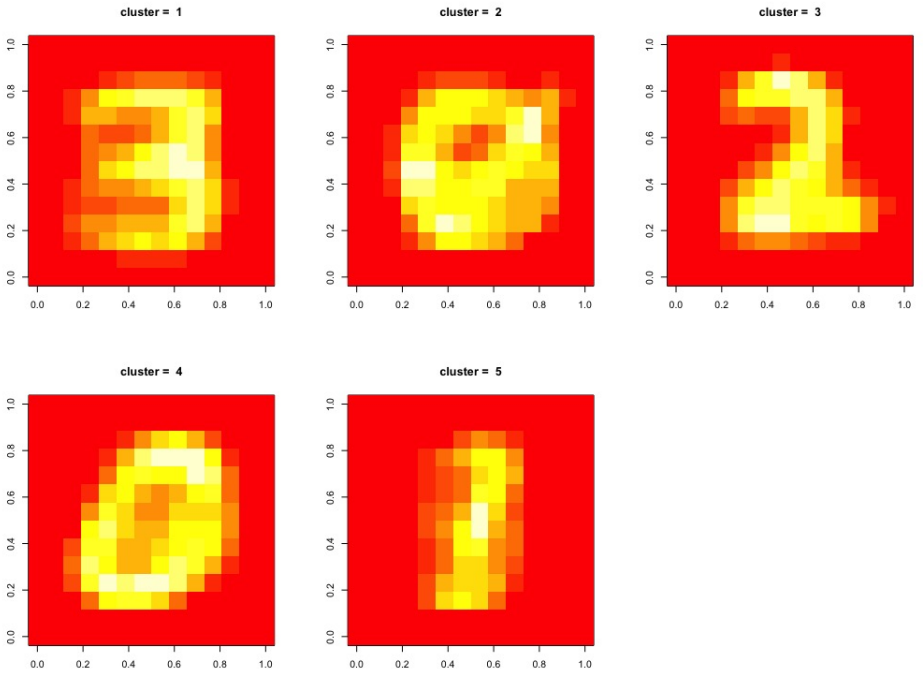

Figure 6: EM for mixture of diagonal Gaussians - experiment 2

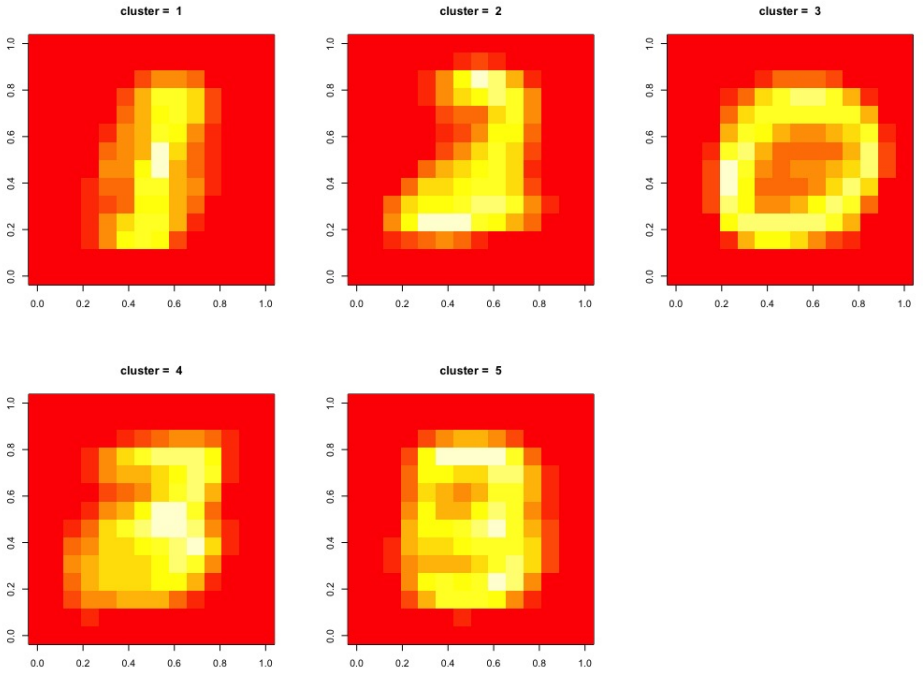

Figure 7: EM for mixture of diagonal Gaussians - experiment 3

From the figures above, we concluded that none of these three experiments could generally capture and recover the five digits. As we mentioned, the Figure 5 associated with the best experiment 2 represented the effectiveness of the mixture of diagonal Gaussians model. Thus

in conclusion, the EM for mixture of diagonal Gaussians algorithm is not recommended for the MNIST handwritten digit clustering task.

Comparison of the two models:

Recall the two model summary tables in the previous sections, we concluded that the EM for mixture of spherical Gaussians model showed systematic better prediction performance than the EM for mixture of diagonal Gaussians model, by simply comparing their test error rates from the experiments. Even for a relative worse case of experiment 3 in spherical Gaussians model, the error rate was still limited under 30%. However, all the experiments for the diagonal Gaussians model showed more than 50% test error rates, which demonstrated that using mixture of diagonal Gaussians assumption is not a good idea. In addition, the Figures in the above sections also follow our conclusion.

The models in the spherical family assume the covariances are diagonal matrices with equal diagonal elements. The models in the diagonal family lead to axis-aligned elliptical components, and the variance of each variables is allowed to vary among clusters. Thus, we need d parameters to estimate for the covariance matrix of each cluster for the diagonal case and we only need 1 parameter for the covariance matrix of each cluster for the spherical case. Diagonal covariance GMM provides a compromise between quality and model size, while spherical covariance GMM requires many components to cover the data.

In conclusion, we realized that the family of mixture of Gaussian distributions models generally might not be recommended for the MNIST handwritten digit clustering task. A major disadvantage of the GMM model is that it might not work well for high dimensional data for computational reasons. Also, the number of mixture models must be manually set. In many case, the user will not know how many mixture model to choose and need to perform experiments on the data at hand to determine the number of clusters. We recommended other possible algorithms (e.g. Convolutional Neural Networks) for clustering task of the MNIST handwritten digit dataset.