Expectation-Maximization in R Markdown

*Cosmin Borsa**

11 October 2018

Abstract

This document is a homework assignment for the course Statistical Computing at the University of Connecticut. First, we are going to derive the updating rules in the construction of the EM algorithm in application to maximum likelihood estimation in finite mixture regression models. Second, we are going to implement the EM algorithm in R. Last, we will estimate some parameters using a given data set.

Keywords: Template; R Markdown; **bookdown**; **knitr**; **Pandoc**

```
## Warning in install.packages :
##   package 'amsmath' is not available (for R version 3.5.1)
```

1 Expectation - Maximization Algorithm

The expectation-maximization algorithm is an iterative method to find maximum likelihood estimates of parameters in a statistical model, where the model depends on unobserved variables. We will derive the updating rules in the algorithm that was given to us based on the construction of an EM algorithm.

The density of y_i which is conditional on $\mathbf{x}_i$ is given in the equation (1)

$$f(y_i|\mathbf{x}_i, \Psi) = \sum_{j=1}^{m} \pi_j \phi(y_i; \mathbf{x}_i^T \boldsymbol{\beta}_j, \sigma^2) \quad \text{with} \quad i = 1, \ldots, n \tag{1}$$

In the density formula (1) the π_js are called mixing the proportions, $\boldsymbol{\beta}_j$ is the regression coefficient vector for the j^{th} group, $\phi(\cdot; \mu, \sigma^2)$ denotes the density function of $N(\mu, \sigma^2)$, and lastly $\Psi = (\pi_1, \boldsymbol{\beta}_1, \ldots, \pi_m, \boldsymbol{\beta}_m, \sigma)^T$ collects all the unknown parameters.

We can infer the unknown parameter Ψ in (1) by

$$\hat{\Psi}_{\text{mle}} = \arg\max_{\Psi} \sum_{i=1}^{n} \log \{ \sum_{j=1}^{m} \pi_j \phi(y_i; \mathbf{x}_i^T \boldsymbol{\beta}_j, \sigma_j^2) \} \tag{2}$$

and we can obtain the values of z_{ij} by

$$z_{ij} = \begin{cases} 1 & \text{if } i\text{th observation is from } j\text{th component;} \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

*cosmin.borsa@uconn.edu; M.S. in Applied Financial Mathematics, Department of Mathematics, University of Connecticut.

1.1 E - Step

In this section we will verify the validity of the provided E-Step of the EM algorithm. To do so, we have to compute the conditional expectation of the complete log-likelihood $l_n^c(\Psi)$ with respect to z_i. However, before we do that we are going to display the equation for the complete log-likelihood $l_n^c(\Psi)$ in (4).

$$l_n^c(\Psi) = \sum_{i=1}^{n}\sum_{j=1}^{m} z_{ij} \log\left\{\pi_j \phi(y_i - \mathbf{x}_i^T \boldsymbol{\beta}_j; 0; \sigma^2)\right\} \tag{4}$$

Next we are going to computes the conditional expectation of $l_n^c(\Psi)$.

$$Q(\Psi|\Psi^{(k)}) = \mathbb{E}_z\{l_n^c(\Psi)\} = \mathbb{E}_z\left[\sum_{i=1}^{n}\sum_{j=1}^{m} z_{ij} \log\left\{\pi_j \phi(y_i - \mathbf{x}_i^T \boldsymbol{\beta}_j; 0; \sigma^2)\right\}\right] \tag{5}$$

$$Q(\Psi|\Psi^{(k)}) = \sum_{i=1}^{n}\sum_{j=1}^{m} \log\left\{\pi_j \phi(y_i - \mathbf{x}_i^T \boldsymbol{\beta}_j; 0; \sigma^2)\right\}\mathbb{E}_z\left[z_{ij}|\mathbf{x}_i, y_i; \Psi^{(k)}\right] \tag{6}$$

Since $p_{ij}^{(k+1)} = \mathbb{E}_z\left[z_{ij}|\mathbf{x}_i, y_i; \Psi^{(k)}\right]$ we have

$$Q(\Psi|\Psi^{(k)}) = \sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}^{(k+1)} \log\left\{\pi_j \phi(y_i - \mathbf{x}_i^T \boldsymbol{\beta}_j; 0; \sigma^2)\right\} \tag{7}$$

As p_{ij} stands for the probability that the observation belongs to the j^{th} component, and

$$p_{ij} = \frac{\pi_j \phi(y_i - \mathbf{x}_i^T \boldsymbol{\beta}_j^{(k)}; 0, \sigma^2)}{\sum_{j=1}^{m} \pi_j \phi(y_i - \mathbf{x}_i^T \boldsymbol{\beta}_j^{(k)}; 0, \sigma^2)} \tag{8}$$

we can see that p_{ij} is obtained by a weighting of normal densities $\phi(y_i - \mathbf{x}_i^T \boldsymbol{\beta}_j^{(k)}; 0, \sigma^2)$ using the mixing proportions π_j. Therefore,

$$\sum_{j=1}^{m} p_{ij} = \frac{\sum_{j=1}^{m} \pi_j \phi(y_i - \mathbf{x}_i^T \boldsymbol{\beta}_j^{(k)}; 0, \sigma^2)}{\sum_{j=1}^{m} \pi_j \phi(y_i - \mathbf{x}_i^T \boldsymbol{\beta}_j^{(k)}; 0, \sigma^2)} = 1 \tag{9}$$

1.2 M - Step

In this section we will verify the validity of the provided M-Step of the EM algorithm. We will maximize $Q(\Psi|\Psi^{(k)})$ to obtain $\boldsymbol{\beta}^{(k+1)}$ and $\sigma^{2^{(k+1)}}$. However, before we do that, we are going to verify that

$$\pi_j^{(k+1)} = \frac{\sum_{i=1}^{n} p_{ij}^{(k+1)}}{n} \tag{10}$$

Since $\pi_1, \ldots, \pi_m$ represent the mixing proportions, summing π_j with respect to j yields $\sum_{j=1}^{m} \pi_j = 1$. Since we are maximizing under the constraint that $\sum_{j=1}^{m} \pi_j = 1$, we are going to use the method of Lagrange multipliers. Let $\mathcal{L}(\pi_1, \ldots, \pi_m)$ be given by the equation (11) where λ is the Lagrange multiplier.

$$\mathcal{L}(\pi_1, \ldots, \pi_m) = \sum_{i=1}^{n} \sum_{j=1}^{m} p_{ij}^{(k+1)} \log \{\pi_j \phi(y_i - \mathbf{x}_i^T \boldsymbol{\beta}_j; 0; \sigma^2)\} - \lambda \left[\sum_{j=1}^{m} \pi_j - 1 \right] \tag{11}$$

We will obtain the desired $\pi_j^{(k+1)}$ by maximizing $\mathcal{L}$. Thus, we are going to take the partial derivative of $\mathcal{L}$ with respect to π_j and set it equal to 0.

$$\frac{\partial \mathcal{L}}{\partial \pi_j} = \sum_{i=1}^{n} \sum_{j=1}^{m} p_{ij}^{(k+1)} \frac{1}{\pi_j^{(k+1)}} - \lambda \tag{12}$$

Since $\sum_{j=1}^{m} p_{ij} = 1$ and $\sum_{j=1}^{m} \pi_j = 1$ the equation becomes

$$\frac{\partial \mathcal{L}}{\partial \pi_j^{(k+1)}} = n - \lambda \tag{13}$$

Next, we will set the equation (13) equal to zero, and solve it for λ. Thus, $\lambda = n$ and plugging it in equation (12) we get

$$\frac{\partial \mathcal{L}}{\partial \pi_j^{(k+1)}} = \sum_{i=1}^{n} \sum_{j=1}^{m} p_{ij}^{(k+1)} \frac{1}{\pi_j^{(k+1)}} - n \tag{14}$$

To obtain $\pi_j^{(k+1)}$ we are going to set the equation (14) to 0 and solve for $\pi_j^{(k+1)}$. Thus,

$$n = \sum_{i=1}^{n} \sum_{j=1}^{m} p_{ij}^{(k+1)} \frac{1}{\pi_j^{(k+1)}} \tag{15}$$

which implies

$$\pi_j^{(k+1)} = \frac{\sum_{i=1}^{n} p_{ij}^{(k+1)}}{n} \tag{16}$$

Next, we will verify that

$$\boldsymbol{\beta}_j^{(k+1)} = \left(\sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^T p_{ij}^{(k+1)}\right)^{-1} \left(\sum_{i=1}^{n} \mathbf{x}_i p_{ij}^{(k+1)} y_i\right) \quad \text{with } j = 1, \ldots, m \tag{17}$$

However, before we do that, we will revise $Q(\Psi|\Psi^{(k)})$ from equation (7). Since $\phi(\cdot; \mu, \sigma^2)$ denotes the probability density function of normal distribution $N(\mu, \sigma^2)$, we can write

$$\phi(y_i - \mathbf{x}_i^T \boldsymbol{\beta}_j; 0; \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[\frac{-1}{2\sigma^2}(y_i - \mathbf{x}_i^T \boldsymbol{\beta}_j^{(k+1)})^T (y_i - \mathbf{x}_i^T \boldsymbol{\beta}_j^{(k+1)})\right] \tag{18}$$

Therefore, we can rewrite $Q(\Psi|\Psi^{(k)})$ as follows

$$Q(\Psi|\Psi^{(k)}) = \sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}^{(k+1)} \left[\log \pi_j + \log\left[\frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[\frac{-1}{2\sigma^2}(y_i - \mathbf{x}_i^T \boldsymbol{\beta}_j^{(k+1)})^T (y_i - \mathbf{x}_i^T \boldsymbol{\beta}_j^{(k+1)})\right]\right]\right] \tag{19}$$

$$Q(\Psi|\Psi^{(k)}) = \sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}^{(k+1)} \left[\log \pi_j + \log\left[\frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[\frac{-1}{2\sigma^2}(y_i^T - (\mathbf{x}_i^T \boldsymbol{\beta}_j^{(k+1)})^T)(y_i - \mathbf{x}_i^T \boldsymbol{\beta}_j^{(k+1)})\right]\right]\right] \tag{20}$$

$$Q(\Psi|\Psi^{(k)}) = \sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}^{(k+1)} \left[\log \pi_j + \log\left[\frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[\frac{-1}{2\sigma^2}(y_i^T - (\boldsymbol{\beta}_j^{(k+1)})^T \mathbf{x}_i)(y_i - \mathbf{x}_i^T \boldsymbol{\beta}_j^{(k+1)})\right]\right]\right] \tag{21}$$

$$Q = \sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}^{(k+1)} \left[\log \pi_j + \log\left[\frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[\frac{-1}{2\sigma^2}(y_i^T y_i - (\boldsymbol{\beta}_j^{(k+1)})^T \mathbf{x}_i y_i - y_i^T \mathbf{x}_i^T \boldsymbol{\beta}_j^{(k+1)} + (\boldsymbol{\beta}_j^{(k+1)})^T \mathbf{x}_i \mathbf{x}_i^T \boldsymbol{\beta}_j^{(k+1)})\right]\right]\right] \tag{22}$$

Since y_i is a scalar, $y_i^T = y_i$ and the following partial derivatives are fairly easy to compute

$$\frac{\partial\left(\boldsymbol{\beta}_j^{(k+1)} \mathbf{x}_i y_i\right)}{\partial \boldsymbol{\beta}_j^{(k+1)}} = \frac{\partial\left((\mathbf{x}_i y_i)^T \boldsymbol{\beta}_j^{(k+1)}\right)}{\partial \boldsymbol{\beta}_j^{(k+1)}} = (\mathbf{x}_i y_i)^T = y_i^T \mathbf{x}_i^T = y_i \mathbf{x}_i^T \tag{23}$$

$$\frac{\partial\left(\left(\boldsymbol{\beta}_j^{(k+1)}\right)^T \mathbf{x}_i \mathbf{x}_i^T \boldsymbol{\beta}_j^{(k+1)}\right)}{\partial \boldsymbol{\beta}_j^{(k+1)}} = \left(\boldsymbol{\beta}_j^{(k+1)}\right)^T \left[\mathbf{x}_i \mathbf{x}_i^T + (\mathbf{x}_i \mathbf{x}_i^T)^T\right] = \left(\boldsymbol{\beta}_j^{(k+1)}\right)^T \left[\mathbf{x}_i \mathbf{x}_i^T + \mathbf{x}_i \mathbf{x}_i^T\right] \tag{24}$$

$$\frac{\partial\left(\left(\boldsymbol{\beta}_j^{(k+1)}\right)^T \mathbf{x}_i\mathbf{x}_i^T\boldsymbol{\beta}_j^{(k+1)}\right)}{\partial\boldsymbol{\beta}_j^{(k+1)}} = 2\left(\boldsymbol{\beta}_j^{(k+1)}\right)^T \mathbf{x}_i\mathbf{x}_i^T \tag{25}$$

Using the partial derivatives in (23) and (25), we will now compute the partial derivative of $Q(\Psi|\Psi^{(k)})$ with respect to $\boldsymbol{\beta}_j^{(k+1)}$. We want to advise the reader that the derivative is fairly complicated; therefore, equation (29) summarizes the derivation.

$$\frac{\partial Q(\Psi|\Psi^{(k)})}{\partial\boldsymbol{\beta}_j^{(k+1)}} = \sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}^{(k+1)}\left(\frac{\sqrt{2\pi\sigma^2}}{\exp\{\frac{-1}{2\sigma^2}(y_i-\mathbf{x}_i^T\boldsymbol{\beta}_j^{(k+1)})^T(y_i-\mathbf{x}_i^T\boldsymbol{\beta}_j^{(k+1)})\}}\right)\Big(\frac{1}{\sqrt{2\pi\sigma^2}}\Big(\frac{-1}{2\sigma^2}\Big(0-y_i\mathbf{x}_i^T-y_i\mathbf{x}_i^T \tag{26}$$

$$+2\left(\boldsymbol{\beta}_j^{(k+1)}\right)^T\mathbf{x}_i\mathbf{x}_i^T\Big)\Big)\exp\{\frac{-1}{2\sigma^2}(y_i-\mathbf{x}_i^T\boldsymbol{\beta}_j^{(k+1)})^T(y_i-\mathbf{x}_i^T\boldsymbol{\beta}_j^{(k+1)})\}\Big) \tag{27}$$

$$\frac{\partial Q(\Psi|\Psi^{(k)})}{\partial\boldsymbol{\beta}_j^{(k+1)}} = \sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}^{(k+1)}\left(\frac{-1}{2\sigma^2}\right)\left(-2y_i\mathbf{x}_i^T+2\left(\boldsymbol{\beta}_j^{(k+1)}\right)^T\mathbf{x}_i\mathbf{x}_i^T\right) \tag{28}$$

$$\frac{\partial Q(\Psi|\Psi^{(k)})}{\partial\boldsymbol{\beta}_j^{(k+1)}} = \sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}^{(k+1)}\left(\frac{1}{\sigma^2}\right)\left(y_i\mathbf{x}_i^T-\left(\boldsymbol{\beta}_j^{(k+1)}\right)^T\mathbf{x}_i\mathbf{x}_i^T\right) \tag{29}$$

Next, we will set the partial derivative (29) equal to 0 to solve for $\boldsymbol{\beta}_j^{(k+1)}$, and we will simplify the equation as much as we can. Thus, we multiply both sides by σ^2, and we will obtain

$$\sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}^{(k+1)}\left(\boldsymbol{\beta}_j^{(k+1)}\right)^T\mathbf{x}_i\mathbf{x}_i^T = \sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}^{(k+1)}y_i\mathbf{x}_i^T \tag{30}$$

Since $\boldsymbol{\beta}_j^{(k+1)}$ depends only on j and $p_{ij}^{(k+1)}$ is a scalar, for $j=1,\ldots,m$ we have

$$\left(\boldsymbol{\beta}_j^{(k+1)}\right)^T\sum_{i=1}^{n} p_{ij}^{(k+1)}\mathbf{x}_i\mathbf{x}_i^T = \sum_{i=1}^{n} p_{ij}^{(k+1)}y_i\mathbf{x}_i^T \tag{31}$$

We can now easily solve for $\left(\boldsymbol{\beta}_j^{(k+1)}\right)^T$.

$$\left(\boldsymbol{\beta}_j^{(k+1)}\right)^T = \left(\sum_{i=1}^{n} p_{ij}^{(k+1)}y_i\mathbf{x}_i^T\right)\left(\sum_{i=1}^{n} p_{ij}^{(k+1)}\mathbf{x}_i\mathbf{x}_i^T\right)^{-1} \tag{32}$$

Last, we will obtain the desired formula for $\boldsymbol{\beta}_j^{(k+1)}$ by taking the transpose of both sides. Hence, for $j = 1, \ldots, m$ we have

$$\boldsymbol{\beta}_j^{(k+1)} = \left[\left(\sum_{i=1}^{n} p_{ij}^{(k+1)} y_i \mathbf{x}_i^T \right) \left(\sum_{i=1}^{n} p_{ij}^{(k+1)} \mathbf{x}_i \mathbf{x}_i^T \right)^{-1} \right]^T \tag{33}$$

$$\tag{34}$$

$$\boldsymbol{\beta}_j^{(k+1)} = \left[\left(\sum_{i=1}^{n} p_{ij}^{(k+1)} \mathbf{x}_i \mathbf{x}_i^T \right)^{-1} \right]^T \left[\left(\sum_{i=1}^{n} p_{ij}^{(k+1)} y_i \mathbf{x}_i^T \right) \right]^T \tag{35}$$

$$\tag{36}$$

$$\boldsymbol{\beta}_j^{(k+1)} = \left[\left(\sum_{i=1}^{n} p_{ij}^{(k+1)} \mathbf{x}_i \mathbf{x}_i^T \right)^T \right]^{-1} \left[\sum_{i=1}^{n} \left(p_{ij}^{(k+1)} y_i \mathbf{x}_i^T \right)^T \right] \tag{37}$$

$$\tag{38}$$

$$\boldsymbol{\beta}_j^{(k+1)} = \left[\sum_{i=1}^{n} \left(p_{ij}^{(k+1)} \mathbf{x}_i \mathbf{x}_i^T \right)^T \right]^{-1} \left[\sum_{i=1}^{n} \mathbf{x}_i \left(p_{ij}^{(k+1)} \right)^T y_i^T \right] \tag{39}$$

$$\tag{40}$$

$$\boldsymbol{\beta}_j^{(k+1)} = \left[\sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^T p_{ij}^{(k+1)} \right]^{-1} \left[\sum_{i=1}^{n} \mathbf{x}_i p_{ij}^{(k+1)} y_i \right] \tag{41}$$

Next, we will verify that

$$\sigma^{2^{(k+1)}} = \frac{\sum_{j=1}^{m} \sum_{i=1}^{n} p_{ij}^{(k+1)} (y_i - \mathbf{x}_i^T \boldsymbol{\beta}_j^{(k+1)})^2}{n} \tag{42}$$

Similar to the verification of the validity of $\boldsymbol{\beta}_j^{(k+1)}$, we are going to replace the probability density function of the normail distribution in (7) to get $Q(\Psi|\Psi^{(k)})$. Thus, we have

$$Q(\Psi|\Psi^{(k)}) = \sum_{i=1}^{n} \sum_{j=1}^{m} p_{ij}^{(k+1)} \log \left[\pi_j \frac{1}{\sqrt{2\pi\sigma^2}} \exp \left[\frac{-1}{2\sigma^2} \left(y_i - \mathbf{x}_i^T \boldsymbol{\beta}_j^k - 0 \right)^2 \right] \right] \tag{43}$$

Next, we are going to take the partial derivative of $Q(\Psi|\Psi^{(k)})$ with respect to $\sigma^{2^{(k+1)}}$. We would like to advise the reader that the derivation is fairly complicated and that only a summarized version is displayed.

$$\frac{\partial Q(\Psi|\Psi^{(k)})}{\partial \sigma^{2^{(k+1)}}} = \sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}^{(k+1)} \left(\frac{\sqrt{2\pi\sigma^{2^{(k+1)}}}}{\exp\left[\frac{-1}{2\sigma^{2^{(k+1)}}}(y_i - \mathbf{x}_i^T\boldsymbol{\beta}_j^{(k)})^2\right]} \right) \Big(\frac{-\pi}{(2\pi\sigma^{2^{(k+1)}})^{(3/2)}} \exp\left[\frac{-1}{2\sigma^{2^{(k+1)}}}(y_i - \mathbf{x}_i^T\boldsymbol{\beta}_j^{(k)})^2\right] \tag{44}$$

$$+ \frac{1}{\sqrt{2\pi\sigma^{2^{(k+1)}}}} \left(\frac{1}{2(\sigma^{2^{(k+1)}})^2}(y_i - \mathbf{x}_i^T\boldsymbol{\beta}_j^{(k)})^2 \right) \exp\left[\frac{-1}{2\sigma^{2^{(k+1)}}}(y_i - \mathbf{x}_i^T\boldsymbol{\beta}_j^{(k)})^2\right] \Big) \tag{45}$$

$$\tag{46}$$

$$\frac{\partial Q(\Psi|\Psi^{(k)})}{\partial \sigma^{2^{(k+1)}}} = \sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}^{(k+1)} \left(\frac{-1}{2\sigma^{2^{(k+1)}}} + \frac{1}{2(\sigma^{2^{(k+1)}})^2}(y_i - \mathbf{x}_i^T\boldsymbol{\beta}_j^{(k)})^2 \right) \tag{47}$$

We will now set this partial derivative in (47) equal to 0, and solve for $\sigma^{2^{(k+1)}}$.

$$\frac{1}{2\sigma^{2^{(k+1)}}} \sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}^{(k+1)} \left(-1 + \frac{1}{\sigma^{2^{(k+1)}}}(y_i - \mathbf{x}_i^T\boldsymbol{\beta}_j^{(k)})^2 \right) = 0 \tag{48}$$

We will multiply both sides by $2\sigma^{2^{(k+1)}}$ and simply the equation. Therefore,

$$\sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}^{(k+1)} = \frac{1}{\sigma^{2^{(k+1)}}} \sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}^{(k+1)}(y_i - \mathbf{x}_i^T\boldsymbol{\beta}_j^{(k)})^2$$

Next, we will multiply again both sides by $\sigma^{2^{(k+1)}}$, and divide by $\sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}^{(k+1)}$.

$$\sigma^{2^{(k+1)}} = \frac{\sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}^{(k+1)}(y_i - \mathbf{x}_i^T\boldsymbol{\beta}_j^{(k)})^2}{\sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}^{(k+1)}} \tag{49}$$

Using equation (9) we may now conclude that

$$\sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}^{(k+1)} = n \tag{50}$$

Thus, we have verified that the provided E- and M-steps were correct since

$$\sigma^{2^{(k+1)}} = \frac{\sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}^{(k+1)}(y_i - \mathbf{x}_i^T\boldsymbol{\beta}_j^{(k)})^2}{n} \tag{51}$$

2 Implement of the EM Algorithm

In this section we are going to implement the EM algorithm in R. To do that we have defined a function `regmix_em` which has a couple of inputs. The inputs of the functions are `y`, the response vector, `xmat` which is the design matrix, `pi.init` which stores the initial values of the π_j's in a $K \times 1$ vector, `beta.init` which saves initial values of the β_j's in a $p \times K$ matrix where p is `ncol(xmat)` and K is the number of components in the mixture, `sigma.init` which gives the initial values of σ, and lastly a control list for controlling the maximum number of iterations and the convergence tolerance.

```
regmix_em <- function(y, xmat, pi.init, beta.init, sigma.init,
  control = list(maxit = 1000, tol = .Machine$double.eps^0.3)){

  xmat <- as.matrix(xmat)

  n <- nrow(xmat)
  p <- ncol(xmat)
  m <- length(pi.init)

  pi <- pi.init
  beta <- beta.init
  sigma <- sigma.init

  maxit <- control$maxit
  tol <- control$tol
  convergent <- 1

  P <- matrix(NA, nrow = n, ncol = m)
  betanew <- matrix(NA, nrow = p, ncol = m)

  for (i in 1:maxit) {
    for (j in 1:n) {
      P[j,] <- pi * dnorm(y[j] - xmat[j,] %*% beta, mean = 0,
      sd = sigma)/sum(pi * dnorm(y[j] - xmat[j,] %*% beta,
      mean = 0, sd = sigma))
    }

    pi.new <- colMeans(P)

    for (j in 1:m) {
      betanew[,j] <- solve(t(xmat) %*% diag(P[,j]) %*% xmat) %*%
      t(xmat) %*% diag(P[,j]) %*% y
    }

    sigmanew <- sqrt(sum(P * (y %*% t(rep(1, m)) - xmat %*% betanew)^2)/n)

    convergent <- sum(abs(pi.new - pi)) + sum(abs(betanew - beta)) +
    abs(sigmanew - sigma)
```

```
    if(convergent < tol) break

    pi <- pi.new
    beta <- betanew
    sigma <- sigmanew

  }

  if(i == maxit)
  message("The maximum number of iterations was reached!")

  return(list(pi = pi.new, beta = betanew, sigma = sigmanew,
  convergent = convergent, iter = i))

}
```

3 Data Generation and Parameters Estimation

In this section we are going to generate data from the mixture regression model using the function regmix_sim

```
regmix_sim <- function(n, pi, beta, sigma) {
    K <- ncol(beta)
    p <- NROW(beta)
    xmat <- matrix(rnorm(n * p), n, p) # normal covaraites
    error <- matrix(rnorm(n * K, sd = sigma), n, K)
    ymat <- xmat %*% beta + error # n by K matrix
    ind <- t(rmultinom(n, size = 1, prob = pi))
    y <- rowSums(ymat * ind)
    data.frame(y, xmat)
}
```

I'm going to use the initial data provided in the exercise to compute the parameters.

```
n <- 400
pi <- c(.3, .4, .3)
bet <- matrix(c( 1,  1,  1, -1, -1, -1), 2, 3)
sig <- 1
set.seed(1205)
dat <- regmix_sim(n, pi, bet, sig)
regmix_em(y = dat[,1], xmat = dat[,-1], pi.init = pi / pi / length(pi),
beta.init = bet * 0, sigma.init = sig / sig,
control = list(maxit = 500, tol = 1e-5))
```

```
## $pi
## [1] 0.3333333 0.3333333 0.3333333
##
## $beta
```

```
##             [,1]       [,2]       [,3]
## [1,]  0.3335660  0.3335660  0.3335660
## [2,] -0.4754645 -0.4754645 -0.4754645
##
## $sigma
## [1] 1.732492
##
## $convergent
## [1] 0
##
## $iter
## [1] 2
```

As we can see, if the inital values of the β_j vector with $j = 1, \ldots, m$ is 0, then, the algorithm stops after 2 iterations. So, in order to improve the estimation, we have changed the vector of initial values of β_j. Instead of assigning a null vector, we are going to assign the vector `matrix(c( 1, 1, 1, -1, -1, -1), 2, 3)` to `beta.init`.

```
regmix_em(y = dat[,1], xmat = dat[,-1], pi.init = pi / pi / length(pi),
beta.init = matrix(c( 1,  1,  1, -1, -1, -1), 2, 3),
sigma.init = sig / sig, control = list(maxit = 500, tol = 1e-5))
```

```
## $pi
## [1] 0.3858218 0.2687873 0.3453909
##
## $beta
##           [,1]       [,2]       [,3]
## [1,] 0.8796608  0.9911852 -0.9136977
## [2,] 0.9341964 -1.2424569 -1.1990372
##
## $sigma
## [1] 1.023598
##
## $convergent
## [1] 8.597268e-06
##
## $iter
## [1] 49
```

After these small changes, the algorithm stops after 49 iterations, and delivers better estimates of the parameters.