Lab 2

Clément Bonnet, Pierre Houdoin

30/10/2020

Exercise 1

Code your own EM algorithm

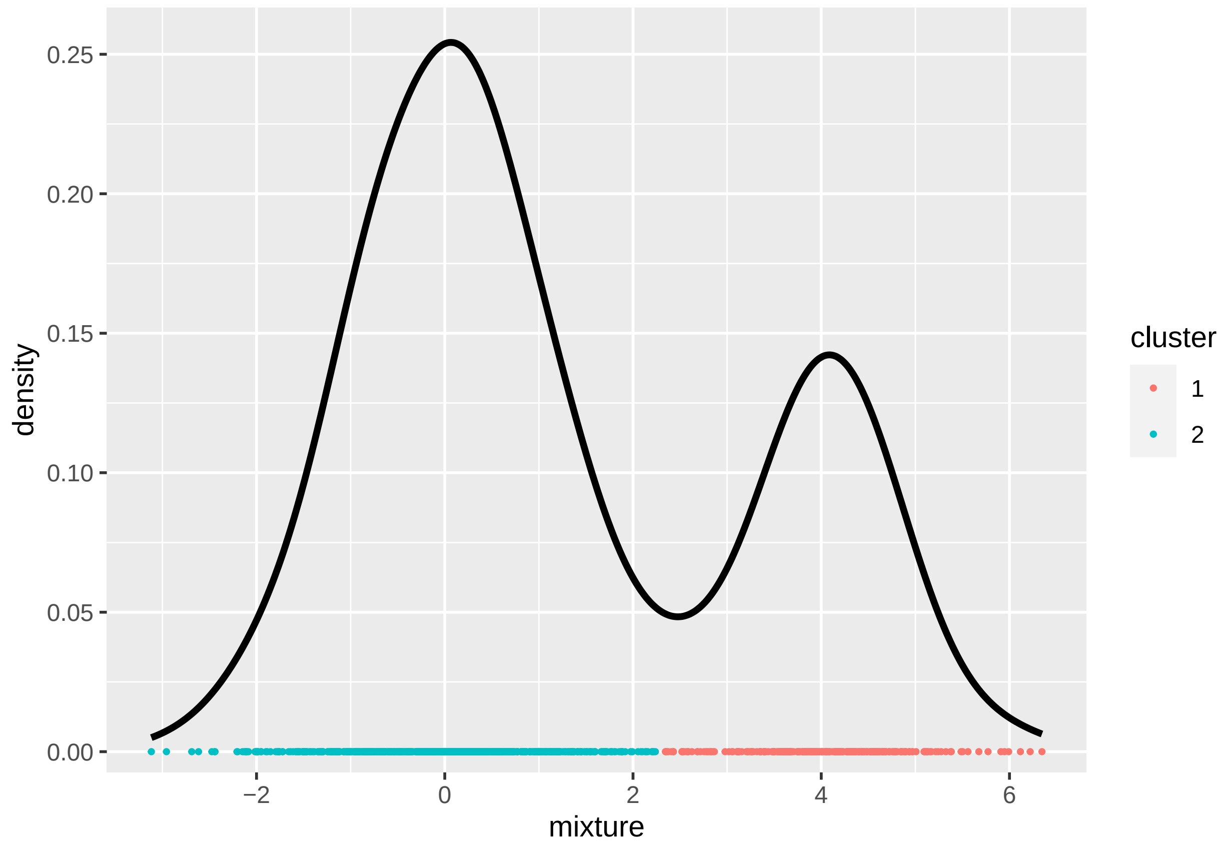
```r
EM <- function(X, K, nb_iter) {
  n <- length(X)
  # Initialization
  prop <- runif(K)
  prop <- prop/sum(prop) # normalize probabilities of clusters
  means <- runif(K, min=-1, max=1)
  sigmas <- runif(K)
  P <- matrix(nrow=n, ncol=K)
  for (t in 1:nb_iter){
    # Expectation step
    for (i in 1:n){
      for (k in 1:K){
        P[i,k] <- prop[k]/(sigmas[k]*sqrt(2*pi))*exp(-1/2 * ((X[i]-means[k])/sigmas[k])**2)
      }
    }
    P <- P/rowSums(P)
    # Maximization step
    prop <- colSums(P)/n
    means <- t(P) %*% X / (n*prop)
    for (k in 1:K){
      sigmas[k] <- sqrt(1/(n*prop[k]) * P[,k]%*%(X - means[k])**2)
    }
  }
  result <- list(means=unlist(means), sigmas=sigmas, prop=prop)
  return(result)
}
```

```r
# 2 clusters with probability 0.3 and 0.7, means 0 and 4, sigmas 1 and 0.75
distr <- sample(c(0, 1), 1000, c(0.3, 0.7), replace = TRUE)
mixture <- distr*rnorm(1000)+(1-distr)*rnorm(1000, 4, 0.75)
result <- EM(mixture, 2, 100)
result
```

Test of the EM algorithm with the a Gaussian Mixture Model made of two clusters.

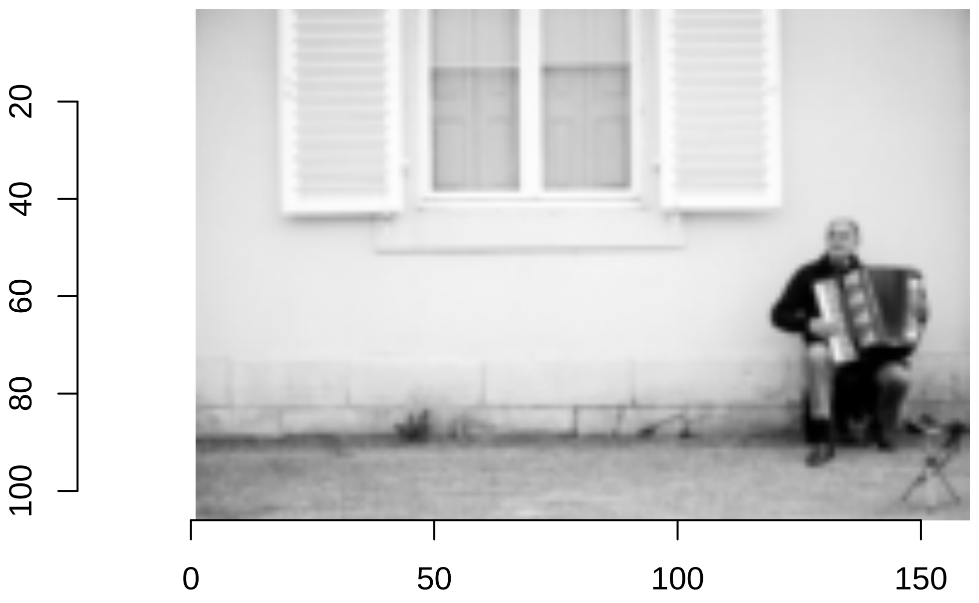
```
## $means
##             [,1]
## [1,] 0.03790566
## [2,] 4.08834447
##
## $sigmas
## [1] 1.0191324 0.7286352
##
## $prop
## [1] 0.7070377 0.2929623
```

The algorithm is able to correctly estimate the latent variables in addition to the parameters of the gaussian mixture.



Exercise 2

Use your own EM algorithm to segment a grayscale picture



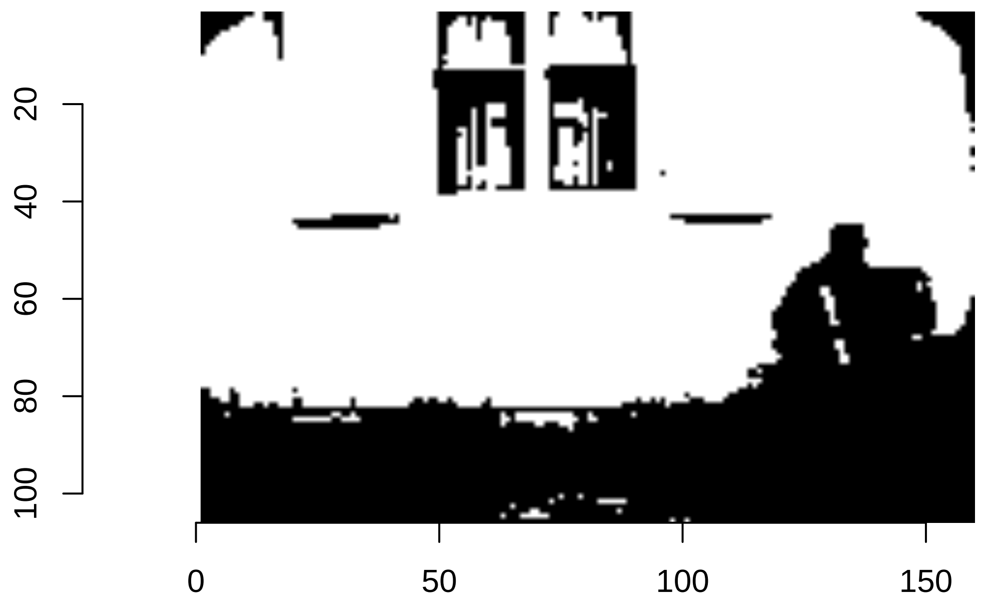
Use of the EM algorithm with 2 clusters in order to segment the picture.

```
df <- as.data.frame(image)
df <- df[df$cc == 1,]  # keep only one channel
result <- EM(df$value, 2, 100)
result
```

```
## $means
##           [,1]
## [1,] 0.8736088
## [2,] 0.6216554
##
## $sigmas
## [1] 0.04888523 0.21326983
##
## $prop
## [1] 0.597845 0.402155
```

Picture segmentation

```
cluster <- dnorm(df$value, mean=result$means[1], sd=result$sigmas[1]) > dnorm(df$value, mean=result$mea
if(sum(cluster) < length(cluster)/2){cluster <- -cluster}  # Swap black and white if needed, in order t
segmentation <- as.cimg(cluster, x=dim(image)[1], y=dim(image)[2])
plot(segmentation)
```



By using two clusters in the EM algorithm, one can segment the image. One can see that the window, the ground and the accordionist belong to the same cluster.

Exercise 3

Mutlivariate EM

I-Presentation of the problem

A) Gaussian mixture model

We note L the likelihood and l the log-likelihood.

We observed a n-sample $x_1, ..., x_n \in \mathbb{R}^{\prime}$, each observation following a gaussian multivariate distribution. The data are extracted from K different distributions.

We aim to find $\theta = (\pi_1, ..., \pi_K, \mu_1, ..., \mu_K, \Sigma_1, ..., \Sigma_K)$ the parameters of our gaussian mixture model so that:

$$\boxed{\forall \mathbf{x} \in \mathbb{R}^{\mathbf{p}}, \mathbf{f}(\mathbf{x}) = \sum_{\mathbf{j=1}}^{\mathbf{K}} \frac{\pi_{\mathbf{j}}}{\sqrt{(\mathbf{2}\pi)^{\mathbf{p}} \det(\mathbf{\Sigma_j})}} \exp\left(-\frac{\mathbf{1}}{\mathbf{2}}(\mathbf{x} - \mu_{\mathbf{j}})^{\mathbf{T}} \mathbf{\Sigma_j^{-1}} (\mathbf{x} - \mu_{\mathbf{j}})\right)}$$

B) Expectation-maximization algorithm

We use the iterative EM algorithm to solve this problem.

First, we randomly initialize $\theta = \theta^\star$, and we compute the matrix $P_{\theta^\star} = (P(Z_i = j | X_i = x_i, \theta^\star))_{i \in [1,n], j \in [1,K]}$. Each row indicates the probability that the observation i derives from each gaussian model, similarly to soft clustering (Estimation part).

The second step consists in maximizing the likelihood of the n-sample. But since, we do not know from which distribution each observation comes from, and only a probability for each one, we maximize the expected value of the likelihood, taken over the unknown data Z.

Thus, we look for $\boxed{\boldsymbol{\pi_1, ..., \pi_K, \mu_1, ..., \mu_K, \Sigma_1, ..., \Sigma_K}}$ that maximize $\boxed{\mathbb{E}_{\mathbf{Z|X},\theta^\star}\left(\mathbf{l(X, Z|}\theta^\star\mathbf{)}\right)}$.

II-Expression of the expected value of the likelihood

A) Expression of the likelihood

We can write:

$$\begin{aligned}
L(x_1, ..., x_n, Z_1, ..., z_n|\theta) &= \prod_{i=1}^{n} L(X_i = x_i, Z_i = z_i|\theta) \\
&= \prod_{i=1}^{n} L(Z_i = z_i|\theta) L(X_i = x_i|Z_i = z_i, \theta) \\
&= \prod_{i=1}^{n} \prod_{j=1}^{K} \left(L(Z_i = j|\theta) L(X_i = x_i|Z_i = j, \theta)\right)^{\mathbb{1}_{Z_i = j}}
\end{aligned}$$

Finally, we can derive the log-likelihood:

$$\boxed{\mathbf{l(x_1, ..., x_n, Z_1, ..., z_n|}\theta\mathbf{)} = \sum_{\mathbf{i=1}}^{\mathbf{n}} \sum_{\mathbf{j=1}}^{\mathbf{K}} \mathbb{1}_{\mathbf{Z_i = j}} \left(\mathbf{l(Z_i = j|}\theta\mathbf{)} + \mathbf{l(X_i = x_i|Z_i = j,} \theta\mathbf{)}\right)}$$

B) Computation of the expected value

Now we can compute $\mathbb{E}_{Z|X,\theta^\star}\left(l(X, Z|\theta)\right)$ and maximize it in the case of multivariate gaussian distributions.

$$\begin{aligned}
\mathbb{E}_{Z|X,\theta^\star}\left(l(X, Z|\theta)\right) &= \mathbb{E}_{Z|X,\theta^\star}\left(\sum_{i=1}^{n} \sum_{j=1}^{K} \mathbb{1}_{Z_i = j} \left(l(Z_i = j|\theta) + l(X_i = x_i|Z_i = j, \theta)\right)\right) \\
&= \sum_{i=1}^{n} \sum_{j=1}^{K} \mathbb{E}_{Z|X,\theta^\star}\left(\mathbb{1}_{Z_i = j}\right) \left(l(Z_i = j|\theta) + l(X_i = x_i|Z_i = j, \theta)\right) \\
&= \sum_{i=1}^{n} \sum_{j=1}^{K} P(Z_i = j|X, \theta^\star) \left(l(Z_i = j|\theta) + l(X_i = x_i|Z_i = j, \theta)\right)
\end{aligned}$$

Now, we are going to give an explicit expression of each term. We denote $g(x, \mu, \Sigma)$ the probability density function of a gaussian multivariate distribution. We have using the Bayes rule:

$$\begin{aligned}
P(Z_i = j|X, \theta^\star) &= \frac{P(X_i = x_i|Z_i = j, \theta^\star)P(Z_i = j|\theta^\star)}{P(X_i = x_i|\theta^\star)} \\
&= \frac{\pi_j^\star g(x_i, \mu_j^\star, \Sigma_j^\star)}{\sum_{k=1}^K \pi_k^\star g(x_i, \mu_k^\star, \Sigma_k^\star)}
\end{aligned}$$

We also compute the terms in the log-likelihood:

$$l(Z_i = j|\theta) + l(X_i = x_i|Z_i = j, \theta) = \ln(\pi_j) + \ln(g(x_i, \mu_j, \Sigma_j))$$

So, we must find θ to optimize the following quantity:

$$\boxed{\sum_{\mathbf{i=1}}^{\mathbf{n}} \sum_{\mathbf{j=1}}^{\mathbf{K}} \frac{\pi_{\mathbf{j}}^\star \mathbf{g}(\mathbf{x_i}, \mu_{\mathbf{j}}^\star, \mathbf{\Sigma_j^\star})}{\sum_{\mathbf{k=1}}^{\mathbf{K}} \pi_{\mathbf{k}}^\star \mathbf{g}(\mathbf{x_i}, \mu_{\mathbf{k}}^\star, \mathbf{\Sigma_k^\star})} (\ln(\pi_{\mathbf{j}}) + \ln(\mathbf{g}(\mathbf{x_i}, \mu_{\mathbf{j}}, \mathbf{\Sigma_j})))}$$

III-Optimization of the expected value

A) Optimization of π

Final part: we are going to find the optimal θ to maximize this quantity. We start by finding the optimal π_j. This is a constrained optimization problem since we have the constraint: $\sum_{j=1}^K \pi_j = 1$. We use the lagrangian multiplier.

We note $T_{i,j}^\star$ the quantity $\frac{\pi_j^\star g(x_i, \mu_j^\star, \Sigma_j^\star)}{\sum_{k=1}^K \pi_k^\star g(x_i, \mu_k^\star, \Sigma_k^\star)}$, so that we look for:

$$\begin{aligned}
&\max_{\pi_1,\ldots,\pi_K} \sum_{i=1}^n \sum_{j=1}^K \frac{\pi_j^\star g(x_i, \mu_j^\star, \Sigma_j^\star)}{\sum_{k=1}^K \pi_k^\star g(x_i, \mu_k^\star, \Sigma_k^\star)} (\ln(\pi_j) + \ln(g(x_i, \mu_j, \Sigma_j))) \qquad \text{with} \qquad \sum_{j=1}^K \pi_j = 1 \\
\iff &\max_{\pi_1,\ldots,\pi_K} \sum_{i=1}^n \sum_{j=1}^K T_{i,j}^\star \ln(\pi_j) \qquad \text{with} \qquad \sum_{j=1}^K \pi_j - 1 = 0
\end{aligned}$$

We use the Lagrangian multiplier to solve this problem. We define:

$$f : \begin{pmatrix} \pi_1 \\ \cdot \\ \cdot \\ \pi_K \end{pmatrix} \mapsto \sum_{i=1}^n \sum_{j=1}^K T_{i,j}^\star \ln(\pi_j)$$

$$g : \begin{pmatrix} \pi_1 \\ \cdot \\ \cdot \\ \pi_K \end{pmatrix} \mapsto \sum_{j=1}^K \pi_j - 1$$

Looking for : $\max\limits_{\pi} f(x) \quad \text{with} \quad g(x) = 0$ lead us to define $\mathcal{L}(\pi, \lambda) = f(\pi) - \lambda g(\pi)$ and look for the saddle points of $\mathcal{L}$ where the hessian matrix of f is definite negative to ensure it is a maximum. We have:

$$\nabla \mathcal{L}(\pi, \lambda) = \begin{pmatrix} \frac{1}{\pi_1} \sum_{i=1}^n T_{i,1}^\star - \lambda \\ \cdot \\ \cdot \\ \frac{1}{\pi_K} \sum_{i=1}^n T_{i,K}^\star - \lambda \\ 1 - \sum_{j=1}^k \pi_j \end{pmatrix} \qquad \text{and} \qquad H(f) = \left(-\frac{\delta_{jk}}{\pi_k^2} \sum_{i=1}^n T_{i,j}^\star \right)_{(j,k) \in [1,K]^2}$$

We can notice that the hessian of f is always definite negative since all $T^\star_{i,j}$ are positives. Thus we just need to find $(\pi_1, ..., \pi_K, \lambda)$ that cancels the gradient of the Lagrangian. We derive $K+1$ and we find:

$$\begin{cases} \pi_1 & = \frac{\sum_{i=1}^n T^\star_{i,1}}{\lambda} \\ . \\ . \\ \pi_K & = \frac{\sum_{i=1}^n T^\star_{i,K}}{\lambda} \\ \sum_{j=1}^K \pi_j & = 1 \end{cases} \iff \begin{cases} \pi_1 & = \frac{\sum_{i=1}^n T^\star_{i,1}}{\lambda} \\ . \\ . \\ \pi_K & = \frac{\sum_{i=1}^n T^\star_{i,K}}{\lambda} \\ \lambda & = \sum_{i=1}^n \sum_{j=1}^K T^\star_{i,j} = n \end{cases} \iff \begin{cases} \pi_1 & = \frac{1}{n}\sum_{i=1}^n T^\star_{i,1} \\ . \\ . \\ \pi_K & = \frac{1}{n}\sum_{i=1}^n T^\star_{i,K} \end{cases}$$

The optimal π for our problem is:

$$\boxed{\begin{cases} \pi_1 & = \frac{1}{n}\sum_{i=1}^n \frac{\pi^\star_1 g(x_i,\mu^\star_1,\Sigma^\star_1)}{\sum_{k=1}^K \pi^\star_k g(x_i,\mu^\star_k,\Sigma^\star_k)} \\ . \\ . \\ \pi_K & = \frac{1}{n}\sum_{i=1}^n \frac{\pi^\star_K g(x_i,\mu^\star_K,\Sigma^\star_K)}{\sum_{k=1}^K \pi^\star_k g(x_i,\mu^\star_k,\Sigma^\star_k)} \end{cases}}$$

B) Optimization of μ

We look for:

$$\max_{\mu_1,\ldots,\mu_K} \sum_{i=1}^n \sum_{j=1}^K \frac{\pi^\star_j g(x_i,\mu^\star_j,\Sigma^\star_j)}{\sum_{k=1}^K \pi^\star_k g(x_i,\mu^\star_k,\Sigma^\star_k)} \left(\ln(\pi_j) + \ln(g(x_i,\mu_j,\Sigma_j))\right)$$
$$\iff \max_{\mu_1,\ldots,\mu_K} \sum_{i=1}^n \sum_{j=1}^K T^\star_{i,j} \ln(g(x_i,\mu_j,\Sigma_j))$$

We are going to give an explicit expression of $\ln(g(x_i,\mu_j,\Sigma_j))$.

$$\ln(g(x_i,\mu_j,\Sigma_j)) = -\frac{1}{2}\left(p\ln(2\pi) + \ln(|\Sigma_j|\right) - \frac{1}{2}(x_i-\mu_j)^T\Sigma_j^{-1}(x_i-\mu_j)$$

Thus, we look for solving the following problem :

$$\max_{\mu_1,\ldots,\mu_K} \sum_{i=1}^n \sum_{j=1}^K T^\star_{i,j}\left(-\frac{1}{2}\left(p\ln(2\pi) + \ln(|\Sigma_j|\right) - \frac{1}{2}(x_i-\mu_j)^T\Sigma_j^{-1}(x_i-\mu_j)\right)$$
$$\iff \min_{\mu_1,\ldots,\mu_K} \sum_{i=1}^n \sum_{j=1}^K T^\star_{i,j}\left(p\ln(2\pi) + \ln(|\Sigma_j|) + (x_i-\mu_j)^T\Sigma_j^{-1}(x_i-\mu_j)\right)$$
$$\iff \min_{\mu_1,\ldots,\mu_K} \sum_{i=1}^n \sum_{j=1}^K T^\star_{i,j}\left(\ln(|\Sigma_j|) + (x_i-\mu_j)^T\Sigma_j^{-1}(x_i-\mu_j)\right)$$
$$\iff \min_{\mu_1,\ldots,\mu_K} \sum_{i=1}^n \sum_{j=1}^K T^\star_{i,j}(x_i-\mu_j)^T\Sigma_j^{-1}(x_i-\mu_j)$$
$$\iff \sum_{j=1}^K \min_{\mu_j\in\mathbb{R}^p} \sum_{i=1}^n T^\star_{i,j}(x_i-\mu_j)^T\Sigma_j^{-1}(x_i-\mu_j)$$

We need to optimize K independant quantities, such equality being true because each term is positive and there are no correlations between the value of μ_j. We are going to compute the gradient for each problem and cancel each of them out, to find the saddle points. We will then keep those where the hessian is definite positive.

We define : $\forall j \in [1, K], f_j : \mu_j \mapsto \sum_{i=1}^{n} T^\star_{i,j}(x_i - \mu_j)^T \Sigma_j^{-1}(x_i - \mu_j)$ and we have :

$$\begin{aligned}
\nabla f_j(\mu_j) &= \sum_{i=1}^{n} T^\star_{i,j} \left(\frac{\partial(x_i - \mu_j)}{\partial \mu_j} \Sigma_j^{-1}(x_i - \mu_j) + \frac{\partial(x_i - \mu_j)}{\partial \mu_j} (\Sigma_j^{-1})^T (x_i - \mu_j) \right) \\
&= -\sum_{i=1}^{n} T^\star_{i,j} \left(\Sigma_j^{-1} + \Sigma_j^{-1} \right) (x_i - \mu_j) \\
&= -2\Sigma_j^{-1} \sum_{i=1}^{n} T^\star_{i,j}(x_i - \mu_j)
\end{aligned}$$

We can notice that the Hessian will always be positive, so we just need to find a point to cancel the gradient. We have to solve the following equation:

$$\begin{aligned}
& \nabla f_j(\mu_j) = 0 \\
\iff \quad & -2\Sigma_j^{-1} \sum_{i=1}^{n} T^\star_{i,j}(x_i - \mu_j) = 0 \\
\iff \quad & \sum_{i=1}^{n} T^\star_{i,j}(x_i - \mu_j) = 0 \qquad \text{because the kernel of gaussian covariance matrix is } \{0\} \\
\iff \quad & \sum_{i=1}^{n} T^\star_{i,j}\mu_j = \sum_{i=1}^{n} T^\star_{i,j} x_i \\
\iff \quad & \mu_j = \sum_{i=1}^{n} \frac{T^\star_{i,j}}{\sum_{i=1}^{n} T^\star_{i,j}} x_i
\end{aligned}$$

The optimal μ for our problem is:

$$\boxed{\begin{cases} \mu_1 & = \sum_{i=1}^{n} \frac{T^\star_{i,1}}{\sum_{i=1}^{n} T^\star_{i,1}} x_i \\ . & \\ . & \\ \mu_K & = \sum_{i=1}^{n} \frac{T^\star_{i,K}}{\sum_{i=1}^{n} T^\star_{i,K}} x_i \end{cases}}$$

C) Optimization of Σ

We look for:

$$\min_{\Sigma_1, \dots, \Sigma_K} \sum_{i=1}^{n} \sum_{j=1}^{K} T^\star_{i,j} \left(\ln(|\Sigma_j|) + (x_i - \mu_j)^T \Sigma_j^{-1} (x_i - \mu_j) \right)$$

We remember that the μ_j are the optimum found previously. We should also add the constraint that Σ should be a covariance matrix, so symmetric positive-definite, but since the problem becomes very complex, we are going to optimize it as a whole, and hope that the solution is positive-definite. Again, we use the positivity of the terms in the sum to decompose the optimization problem:

$$\begin{aligned}
&\min_{\Sigma_1,\dots,\Sigma_K} \sum_{i=1}^{n}\sum_{j=1}^{K} T^\star_{i,j}\left(\ln(|\Sigma_j|) + (x_i-\mu_j)^T\Sigma_j^{-1}(x_i-\mu_j)\right)\\
\iff &\sum_{j=1}^{K}\min_{\Sigma_j}\sum_{i=1}^{n} T^\star_{i,j}\left(\ln(|\Sigma_j|) + (x_i-\mu_j)^T\Sigma_j^{-1}(x_i-\mu_j)\right)\\
\iff &\sum_{j=1}^{K}\min_{\Sigma_j^{-1}}\sum_{i=1}^{n} T^\star_{i,j}\left(-\ln(|\Sigma_j|) + (x_i-\mu_j)^T\Sigma_j(x_i-\mu_j)\right)
\end{aligned}$$

We define: $\forall j \in [1,K], f_j : \Sigma_j \mapsto \sum_{i=1}^{n} T^\star_{i,j}\left(-\ln(|\Sigma_j|) + (x_i-\mu_j)^T\Sigma_j(x_i-\mu_j)\right)$ and we have:

$$\begin{aligned}
\frac{\partial f_j}{\partial \Sigma_j}(\Sigma_j) &= \sum_{i=1}^{n} T^\star_{i,j}\left(-\Sigma_j^{-1} + (x_i-\mu_j)(x_i-\mu_j)^T\right)\\
&= \sum_{i=1}^{n} T^\star_{i,j}\left((x_i-\mu_j)(x_i-\mu_j)^T - \Sigma_j^{-1}\right) \qquad \text{and}\\
\frac{\partial f_j}{\partial \Sigma_j}(\Sigma_j) &= 0\\
\iff \qquad 0 &= \sum_{i=1}^{n} T^\star_{i,j}\left((x_i-\mu_j)(x_i-\mu_j)^T - \Sigma_j^{-1}\right)\\
\iff \Sigma_j^{-1} &= \sum_{i=1}^{n} \frac{T^\star_{i,j}}{\sum_{i=1}^{n} T^\star_{i,j}}(x_i-\mu_j)(x_i-\mu_j)^T
\end{aligned}$$

We finally check that this matrix is symmetric positive-definite, it is the case, and we take the inverse, since we optimized the minimum over Σ_j^{-1}. Finally, we get that the optimal Σ for our problem is:

$$\boxed{\begin{cases}
\Sigma_1 &= \sum_{i=1}^{n} \frac{T^\star_{i,1}}{\sum_{i=1}^{n} T^\star_{i,1}}(x_i-\mu_1)(x_i-\mu_1)^T\\
.\\
.\\
\Sigma_K &= \sum_{i=1}^{n} \frac{T^\star_{i,K}}{\sum_{i=1}^{n} T^\star_{i,K}}(x_i-\mu_K)(x_i-\mu_K)^T
\end{cases}}$$

IV-Conclusion

The final algorithm to solve our problem can be described as following:

1) Randomly initialize $\theta^\star = (\pi^\star_1, ..., \pi^\star_K, \mu^\star_1, ..., \mu^\star_K, \Sigma^\star_1, ..., \Sigma^\star_K)$
2) Compute the coefficients $T^\star_{i,j}$ with the following relation (Expectation step):

$$T^\star_{i,j} = \frac{\pi^\star_j \exp\left(-\frac{1}{2}(x_i-\mu^\star_j)^T\Sigma_j^{-1^\star}(x_i-\mu^\star_j)\right)}{\sum_{k=1}^{K}\frac{\pi^\star_k}{\sqrt{|\Sigma^\star_k\Sigma_j^{\star-1}|}}\exp\left(-\frac{1}{2}(x_i-\mu^\star_k)^T\Sigma_k^{\star-1}(x_i-\mu^\star_k)\right)}$$

3) Compute $\theta = (\pi_1, ..., \pi_K, \mu_1, ..., \mu_K, \Sigma_1, ..., \Sigma_K)$ with the following relations (Maximization step):

$$\begin{cases} \pi_1 & = \frac{1}{n}\sum_{i=1}^n T^\star_{i,1} \\ . & \\ . & \\ \pi_K & = \frac{1}{n}\sum_{i=1}^n T^\star_{i,K} \end{cases} \quad \begin{cases} \mu_1 & = \sum_{i=1}^n \frac{T^\star_{i,1}}{\sum_{i=1}^n T^\star_{i,1}} x_i \\ . & \\ . & \\ \mu_K & = \sum_{i=1}^n \frac{T^\star_{i,K}}{\sum_{i=1}^n T^\star_{i,K}} x_i \end{cases} \quad \begin{cases} \Sigma_1 & = \sum_{i=1}^n \frac{T^\star_{i,1}}{\sum_{i=1}^n T^\star_{i,1}} (x_i-\mu_1)(x_i-\mu_1)^T \\ . & \\ . & \\ \Sigma_K & = \sum_{i=1}^n \frac{T^\star_{i,K}}{\sum_{i=1}^n T^\star_{i,K}} (x_i-\mu_K)(x_i-\mu_K)^T \end{cases}$$

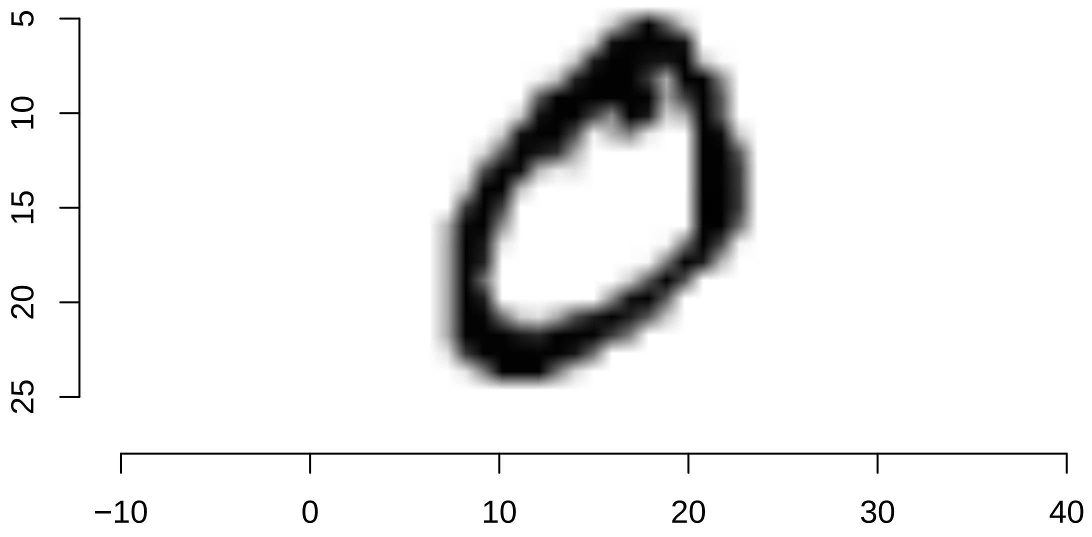
Repeat steps 2 and 3 until convergence.

Exercise 4

This digit does not exist

```r
n_examples <- 5000  # limit to 5000 examples
if(file.exists("mnist_train.csv")){
  dataset <- read.csv("mnist_train.csv")
} else{
  dataset <- read.csv(url("https://pjreddie.com/media/files/mnist_train.csv"))
}
mnist <- dataset[1:n_examples,]
labels <- mnist[,1]
train <- mnist[,2:785]
```

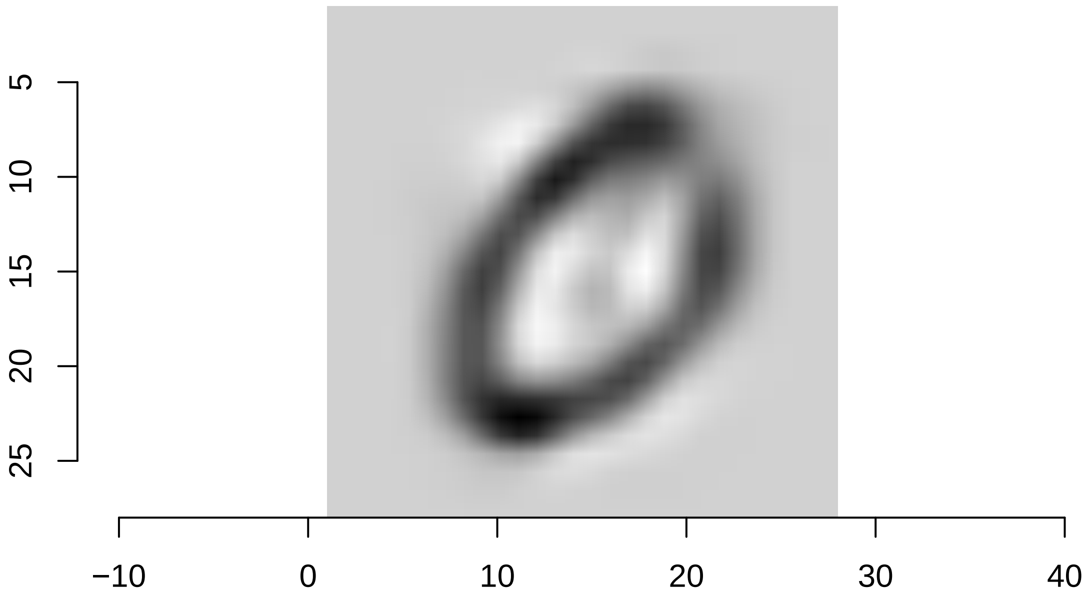
Loading the MNIST dataset Example for digit 0:



We use Principal Component Analysis (PCA) to reduce the dimension of the training set from 784 to 30 features.

```
# Principal Component Analysis
n_comp <- 30  # keep 30 components out of 784
pca <- prcomp(train)
train_pca <- pca$x[,1:n_comp]
reconstruction_mat <- pca$rotation[,1:n_comp]
train_hat <- train_pca %*% t(reconstruction_mat)  # reconstruct the original dataset
train_hat <- scale(train_hat, center=-colMeans(train), scale=FALSE)
```

Reconstruction of digit 0:



We now fit the low-dimensional training set to the multidimensional EM algorithm with $K = 10$ clusters.

```
library(mclust)
result <- Mclust(train_pca, G=10)
```

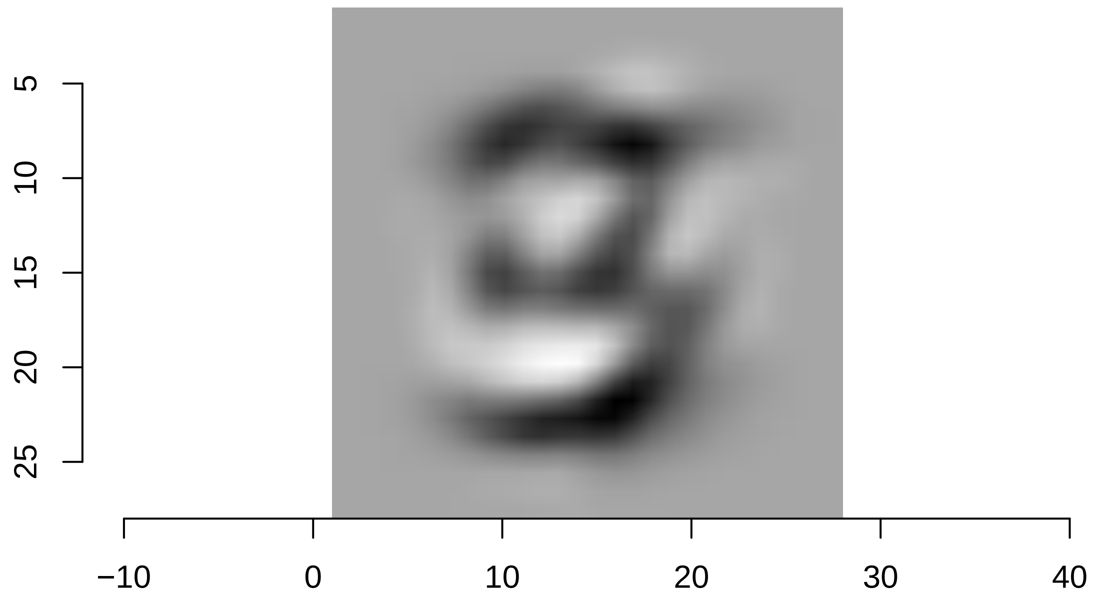
Having fitted 10 clusters to the training set, one can now generate fake digits using the Gaussian Mixture Model parameters.

```
library(MASS)
N <- 1000    # number of fake digits to generate
clusters <- sample(1:10, size=N, prob=result$parameters$pro, replace=TRUE)  # sample clusters
gen_samples <- array(dim=c(N,30))
for (i in 1:N){
  gen_samples[i,] <- mvrnorm(n=1,mu=result$parameters$mean[,clusters[i]],Sigma=result$parameters$varian
}
gen_hat <- gen_samples %*% t(reconstruction_mat)  # goes back to the original dimension
```

```r
gen_hat <- scale(gen_hat, center=-colMeans(train), scale=FALSE)
```

Example of a random generated digit

```r
i <- sample(1:N, 1)
plot(as.cimg(unlist(-gen_hat[i,]), x=28, y=28))
```



A quick website has been created to display generated digits. The URL is the following: https://clementbonnet.com/thisdigitdoesnotexist/.
