Expectation Maximization and Gaussian Mixture Models

```r
library(MASS)
library(mvtnorm)

data <- read.csv("data/gmm.csv", row.names=1) # Read data
data <- data[,2:ncol(data)]
head(data)[,1:5]
```

```
##          X1       X2         X3        X4       X5
## 1  8.074939 17.96900 -2.5016382  5.134277 3.482768
## 2 13.345054 25.28221 -1.1712892 10.777469 4.532955
## 3 11.899930 20.52731 -1.4168956  7.783895 2.992181
## 4 12.693333 25.99682  2.8120403 12.817708 8.764001
## 5 14.605637 20.88500  0.4832833  9.327261 5.427967
## 6 10.336272 20.26761 -1.0113965  7.477345 3.278330
```

Expectation Maximization for Gaussian Mixture Models

Given a GMM, our goal is to maximize the likelihood function with respect to the parameters (**means**, **covariances**, and **mixing coefficients**).

1. We begin by **initializing** $\pi_1, ..., \pi_k$, $\mu_1, ..., \mu_k$, $\Sigma_1, ...\Sigma_k$ randomly.
2. In the **expectation step**, we calculate the probability of each data point n belonging to a cluster k.

$$w_k^{(n)} := p(z^{(n)} = k | x^{(n)}; \pi, \mu, \Sigma)$$

   This can be calculated using the following:

$$w_k^{(n)} = \frac{\pi_k g_k(x)}{\sum_{j=1}^{k} \pi_j g_j(x)}$$

   where $g_k(x)$ is the multivariate Gaussian for cluster k.

3. In the **maximization step**, we update the parameters as follows:

$$\mu_k^{new} := \frac{\sum_{n=1}^{N} w_k^{(n)} x^{(n)}}{N_k}$$

$$\Sigma_k^{new} := \frac{\sum_{n=1}^{N} w_k^{(n)} (x^{(n)} - \mu_k)(x^{(n)} - \mu_k)^T}{N_k}$$

$$\pi_k^{new} := \frac{N_k}{N}$$

```r
set.seed(1)

myGMM <- function(data, k, max_iter, conv_tol) {
  # data: n x d matrix, where n = number of samples and d = number of features
  # k: number of clusters
```

```r
# max_iter: maximum number of iterations
# conv_tol: tolerance for convergence

# -------------------- Step 1: Initialization ---------------------
# Randomly select data points to use as initial means
row.index <- sample.int(n=nrow(data), size=k, replace=TRUE)
mean <- mapply(function(row) return(as.numeric(data[row,])), row=row.index, SIMPLIFY=FALSE)
# Set the covariance matrix for each cluster equal to covariance of full training set
covariance <- replicate(k, cov(data), simplify=FALSE)
# Give each cluster equal mixing coefficient
mixing <- replicate(k, 1/k)

# --------------- Step 2: Expectation Maximization ----------------

# Loop until convergence
for (iter in 1:max_iter) {

  # -------------------- Step 2a: Expectation ---------------------
  # Matrix to hold pdf for each data point for every cluster
  z <- matrix(nrow=nrow(data), ncol=k)

  for (j in 1:k) {
    # Calculate pdf for each data point for each cluster j
    z[,j] <- dmvnorm(x=data, mean=mean[[j]], sigma=covariance[[j]])
    # Calculate weighted pdf by multiplying each pdf by mixing proportion
    z[,j] <- mixing[j] * z[,j]
  }

  # Divide weighted pdf by sum of weighted pdf
  z <- z / sum(z)

  # ------------------- Step 2b: Maximization ---------------------
  old.mean <- mean

  for (j in 1:k) {
    # Update mean for cluster j
    mean[[j]] <- as.numeric( (z[,j] %*% as.matrix(data)) / sum(z[,j]) )

    # Update covariance matrix for cluster j
    k.covariance <- matrix(nrow=nrow(covariance[[j]]), ncol=ncol(covariance[[j]]))
    x.minus.mean <- sweep(data, MARGIN=2, mean[[j]], FUN="-") # Subtract cluster mean from all data p
    for (i in 1:nrow(data)) {
      k.covariance <- k.covariance + z[i,j] * ( t(as.matrix(data[i,])) %*% as.matrix(data[i,]) )
    }
    covariance[[j]] <- k.covariance / sum(z[,j])

    # Update mixing proportion for cluster j
    mixing[j] <- mean(z[,j])
  }

  # Check for convergence
  if (isTRUE(all.equal(old.mean, mean, tolerance=conv_tol))) {
    break
```

```
    }
  }
  return(list(mixing, mean, z))
}

params <- myGMM(data, k=3, max_iter=100, conv_tol=1e-6)
```

Given that there are no other bugs in *myGMM()*, it seems that initialization of Σ is not entirely the best possible since after a few iterations I get NaN. None of the following intializations worked for me:

1. $\Sigma = \sigma^2$, where $\sigma^2 = cov(data)$
2. $\Sigma = \mathbb{I}$, where $\mathbb{I}$ is the identity matrix
3. $\Sigma = \sigma^2\mathbb{I}$