A Practical Introduction to Finite Mixture Models in R
Semester Paper

Wayne Zeng

Supervisor: Dr. Lukas Meier

Spring Semester 2022

Abstract

Finite mixture models are a flexible, convenient and semiparametric way to model unknown distributional shapes. They are also good at modelling group structures. A gentle introduction to mixture models will be given as well as practical use cases.

Acknowledgements

I would like to thank my supervisor, Lukas Meier, for his support and patience over the last few months. The fast, precise communication we were able to maintain was invaluable for my progress and understanding throughout the writing of this paper, and is greatly appreciated.

Introduction

Classification and subgrouping are important in many fields of data analysis. In many datasets, the memberships of each datapoint may not be easily discernable per se. The basic goal in such cases is to examine a sample of measurements and thus differentiate these subgroups of individuals, even when there are no observable variables that readily index into which subgroup an individual properly belongs, leading to notions such as model-based clustering and unsupervised clustering.

Finite mixtures of distributions (FMMs or *finite mixture models)* are used to provide computationally convenient representations for modelling complex distributions of data on random phenomena. FMMs form the bases of cluster and latent class analyses, discriminant analysis and image analysis. They also provide descriptive models for distributions where a single component distribution is insufficient, i.e. they can give descriptions of entire subgroups, rather than assignments of individuals to those subgroups.

1 Formulation of Mixture Distribution

1.1 Basic Definition

Let $\mathbf{Y}_1, \ldots, \mathbf{Y}_n$ denote a random sample of size n, where $\mathbf{Y}_j$ is a p-dimensional random vector with probability density function (pdf) $f(\mathbf{y}_j)$ on $\mathbb{R}^p$.

$\mathbf{Y}_j$ contains the random variables corresponding to p measurements

Definition 1.1. The probability density function (pdf), or in the discrete case, the probability mass function, of a p-dimensional random vector $\mathbf{Y}$ takes the form

$$f(\mathbf{y}) = \sum_{i=1}^{g} \pi_i f_i(\mathbf{y}), \tag{1}$$

where the coefficients or "mixing proportions" π_i are non-negative and $\sum_{i=1}^{g} \pi_i = 1$, and $f_i(\mathbf{y})$ are the $g-$component densities, where g denotes the number of components in the mixture model.

The component densities may be parameterised with a vector θ_i of parameters. We can then rewrite the above definition 1 as

$$f(\mathbf{y}; \mathbf{\Psi}) = \sum_{i=1}^{g} \pi_i f_i(\mathbf{y}; \theta_i), \tag{2}$$

where $\mathbf{\Psi} = (\xi^\intercal, \pi_1, \pi_2, \ldots, \pi_{g-1})$ is the vector of unknown parameters and ξ is the vector of elements of θ_i

1.2 Interpretation of Mixture Models

We can generate a random vector $\mathbf{Y}$ with g-component mixture density 1 by defining a new categorical random variable Z_j such that:

$$\begin{aligned} Z_j &= \{1, \ldots, g\}, \\ P(Z_j = i) &= \pi_i, \\ P(\mathbf{Y}_j | Z_j = i) &= f(\mathbf{y}). \end{aligned}$$

Then the marginal density of $\mathbf{Y}$ is precisely $f(\mathbf{y})$.

We will later see that it is convenient to work with a g-dimensional label.

1.3 Example: Mixtures of Two Normal Homoscedastic Components

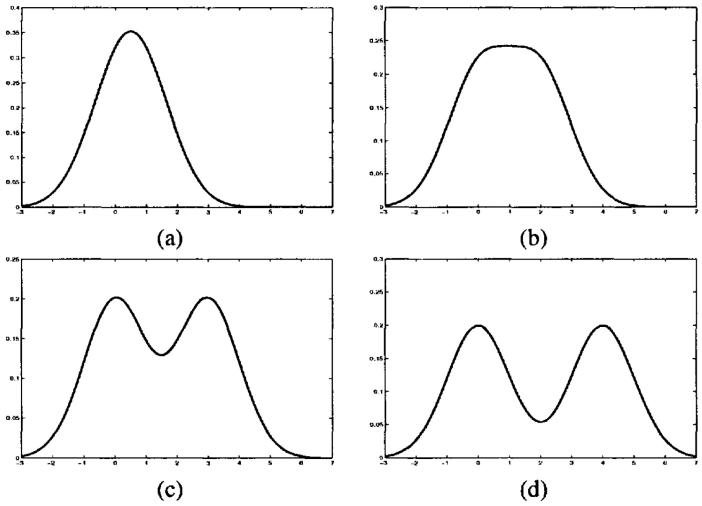
Consider a mixture of two univariate normal components with common variance σ^2 and means μ_1, μ_2 with corresponding mixture proportions π_1, π_2 such that

$$f(y_i) = \pi_1 \phi(y_j; \mu_1, \sigma^2) + \pi_2 \phi(y_j; \mu_2, \sigma^2), \quad \phi(y_j; \mu_i, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(y_j - \mu_j)^2}{2\sigma^2}}. \tag{3}$$

We consider two scenarios: one where the means of the two component normal densities are far apart vs. when they are sufficiently close together.

If the two means are far apart, then we would expect (3) to resemble two normal densities side by side. For this purpose, let us define the Mahalanobis distance:

Figure 1: TO REMAKE OWN GRAPHIC



$$\Delta = |\mu_1 - \mu_2|/\sigma$$

In the other case where the means are more closely together, it is more difficult to see the distinction between them, leading to an asymmetric density if the components are not represented in equal proportions.

1.4 Parametric Models

In this paper, we will mostly be focusing on parametric mixture models. These are models where the component densities $f_i(\mathbf{y}_j)$ belong to some parametric family. We can then rewrite this as $f_i(\mathbf{y}_i; \boldsymbol{\theta}_i)$, $\boldsymbol{\theta}_i$ being the vector of unknown parameters in the postulated form for the ith component density in the mixture.

We rewrite (3) as

$$f(\mathbf{y}_j; \boldsymbol{\Psi}) = \sum_{i=1}^{g} \pi_i f_i(\mathbf{y}_j; \boldsymbol{\theta}_j), \quad \boldsymbol{\Psi} = (\pi_1, \ldots, \pi_{g-1}, \boldsymbol{\xi}^T)^T \tag{4}$$

where $\boldsymbol{\xi}$ is the vector containing all the parameters in $\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_g$ known in advance to be distinct.

1.5 Identifiablity and Component Labels: WIP

Based on 2.2 of [2].

2 Maximum Likelihood Fitting of Mixture Models and the EM Algorithm

Maximum likelihood is the most commonly used approach to fitting mixture distributions and density estimation. This is done by searching across probability distributions and their parameters. However, the MLE requires our dataset to be complete, since the likelihood no longer can be factorised and is no longer convex, leading to a lack of a closed solution whilst maximising said likelihood. Furthermore, Bayesian inference of the parameters is made harder.

The *Expectation Maximisation Algorithm*, or EM, is used to perform MLE in the presence of latent variables. This is a general algorithm commonly used for density estimation with missing/latent data.

The EM algorithm is an iterative approach that cycles between two steps:

- The first E-step (expectation) attempts to estimate the missing or latent variables.
- The second M-step (maximisation) optimises the parameters of the model to explain the data.

We repeat this process until we reach convergence.

2.1 Latent Variable Models and Relationship to FMMs

In many statistical problems, there are many variables which cannot be measured directly. We denote these variables to be *latent*. We often link these latent variables to observable ones in order to infer the values of the latent ones.

Usually, statistical procedures and learning algorithms are used to estimate parameters of probability distributions to best fit the density of a given training dataset, mostly using maximum likelihood.

Let us consider the most simple case of a finite mixture model, where the dataset is comprised of many points that happen to be generated by two different processes. We generate the points to be Gaussian distributed, but the distributions are similar enough that it is not obvious to which distribution a given point may belong. The important remark here is that processes used to generate such data *represents* our latent variables. This clearly influences the data, but is not observable. As seen above, we cannot simply use MLE anymore, thus EM can be employed to estimate the parameters of a FMM.

2.2 Formulation as incomplete data problem

We use the component label vector $\mathbf{Z}_j$ with the indicator variables. This notation is useful (although unintuitive at first) because it allows the MLE of the mixture distribution to be computed easily via the EM-algorithm.

We write

$$\mathbf{Y}_1, \ldots, \mathbf{Y}_n \overset{i.i.d.}{\sim} F, \tag{5}$$

where $F(\mathbf{y}_i)$ denotes the distribution function corresponding to the mixture density $f(\mathbf{y}_i)$.

The feature data $\mathbf{y}_1, \ldots, \mathbf{y}_n$ is incomplete since their associated component-indicator vectors $\mathbf{z}_1, \ldots, \mathbf{z}_n$ are not available, hence we consider this to be a latent variable model.

Furthermore, we define the complete data vector:

$$\mathbf{y}_c = (\mathbf{y}^T, \mathbf{z}^T)^T, \tag{6}$$

where

$$\mathbf{y} = (\mathbf{y}_1^T, \ldots, \mathbf{y}_n^T)^T \quad \mathbf{z} = (\mathbf{z}_1^T, \ldots, \mathbf{z}_n^T)^T \tag{7}$$

All the observations $\mathbf{y}_j$ have been completely recorded and fully observed. In addition, the component label vectors $\mathbf{z}$ are taken to be the realised values of the random vectors $\mathbf{Z}_1, \ldots, \mathbf{Z}_n$

We can assume they are distributed unconditionally as

$$\mathbf{Z}_1, \ldots, \mathbf{Z}_1, \overset{i.i.d.}{\sim} \text{Mult}_g(1, \boldsymbol{\pi}). \tag{8}$$

The ith mixing proportion π_i can be interpreted as the prior probability that the entity belongs to the ith component of the mixture distribution, whereas the posterior probability that the entity belongs to the ith component with $\mathbf{y}_j$ having been observed on it is given by

$$\begin{aligned}\tau(\mathbf{y}_j) &= P(\text{entity} \in i\text{th component} \,|\, \mathbf{y}_j) \\ &= P(Z_{ij} = 1 \,|\, \mathbf{y}_j) \\ &= \pi_i f_i(\mathbf{y}_j)/f(\mathbf{y}_j)\end{aligned}$$

This assumption means that the distribution of the complete data vector $\mathbf{Y}_c$ implies the appropriate distribution for the incomplete-data vector $\mathbf{Y}$. The final complete log likelihood for $\boldsymbol{\Psi}$ is

$$\log L_c(\boldsymbol{\Psi}) = \sum_{i=1}^{g}\sum_{j=1}^{n} z_{ij}\{\log \pi_i + \log f_i(\mathbf{y}_j; \boldsymbol{\theta}_i)\}. \tag{9}$$

2.3 Application of the EM Algorithm for Mixture Models

Let us consider the standard FMM expression:

$$f(\mathbf{y}_j; \boldsymbol{\Psi}) = \sum_{i=1}^{g} \pi_i f_i(\mathbf{y}_j; \boldsymbol{\theta}_i),$$

Where $\mathbf{y} = (\mathbf{y}_1^T, \ldots, \mathbf{y}_n^T)^T$, $\boldsymbol{\Psi}$ is the observed incomplete data vector as defined above.

The log likelihood for $\boldsymbol{\Psi}$ formed *from the observed data is* given by the sum of the $f(\mathbf{y}_j; \boldsymbol{\Psi})$s:

$$\begin{aligned}\log L(\boldsymbol{\Psi}) &= \sum_{j=1}^{n} \log f(\mathbf{y}_j; \boldsymbol{\Psi}) \\ &= \sum_{j=1}^{n} \log\{\sum_{i=1}^{g} \pi_i f_i(\mathbf{y}_j; \boldsymbol{\theta}_i)\},\end{aligned}$$

and noting the difference between this expression and (9). To obtain the MLE, we would like to solve the likelihood equation

$$\frac{\partial \log L(\boldsymbol{\Psi})}{\partial \boldsymbol{\Psi}} = \mathbf{0}. \tag{10}$$

We denote the $\hat{\boldsymbol{\Psi}}$ as the MLE of $\boldsymbol{\Psi}$ and let

$$\tau_i(\mathbf{y}_i; \boldsymbol{\Psi}) = \frac{\pi_i f_i(\mathbf{y}_i; \boldsymbol{\theta}_i)}{\sum_{h=1}^{g} \pi_h f_h(\mathbf{y}_i; \boldsymbol{\theta}_h))} \tag{11}$$

be the posterior probability that $\mathbf{y}_j$ belongs to the ith component of the mixture. Then $\hat{\boldsymbol{\Psi}}$ satisfies the following expressions:

$$\hat{\pi_i} = \sum_{j=1}^{n} \tau_i(\mathbf{y}_i; \hat{\boldsymbol{\Psi}})/n \quad (i = 1, \ldots, g) \tag{12}$$

and

$$\sum_{i=1}^{g} \sum_{j=1}^{n} \tau_i(\mathbf{y}_i; \hat{\boldsymbol{\Psi}}) \partial \log_i(\mathbf{y}_i; \hat{\boldsymbol{\theta}}_i)/\partial \boldsymbol{\xi} = \mathbf{0}. \tag{13}$$

The two equations (12) and (13) build the iterative foundation of the EM algorithm. The proof of the two above equations involves Bayes' equation and can be found in [3], Section 1.4.

E-Step

The E-step treats the latent data labels $\mathbf{z}_{ij}$s as missing. It handles the addition of these unobservable labels by taking the conditional expectation of the *complete* log-likelihood $\log L_c(\boldsymbol{\Psi})$ given $\mathbf{y}$ using the current fit for $\boldsymbol{\Psi}$. For the first step, we take this initial fit to be $\boldsymbol{\Psi}^{(0)}$. This initial value can be found by plugging into the right hand side of the equations.

$$Q(\boldsymbol{\Psi}; \boldsymbol{\Psi}^{(0)}) = E_{\boldsymbol{\Psi}^{(0)}}\{\log L_c \boldsymbol{\Psi} | \mathbf{y}\}. \tag{14}$$

The subscript $\boldsymbol{\Psi}^{(0)}$ on the right hand side explicitly expresses that we are taking the expectation with respect to the current estimate $\boldsymbol{\Psi}^{(0)}$. In later iterations, we compute a new estimate for $\boldsymbol{\Psi}^{(k+1)}$ by taking the expectation w.r.t. $\boldsymbol{\Psi}^{(k)}$.

From (9), we see that $\log L_c(\boldsymbol{\Psi})$ is linear in the unobservable labels z_{ij}. This implies that

$$\begin{aligned} E_{\boldsymbol{\Psi}^{(k)}}\{Z_{ij}|\mathbf{y}\} &= \mathrm{pr}_{\boldsymbol{\Psi}^{(k)}}\{Z_{ij} = 1|\mathbf{y}\} \\ &= \tau_i(\mathbf{y}_i; \boldsymbol{\Psi}^{(k)}) \\ &= \frac{\pi_i^{(k)} f_i(\mathbf{y}_i; \boldsymbol{\theta}_i^{(k)})}{f(\mathbf{y}_j; \boldsymbol{\Psi}^{(k)})} \\ &= \frac{\pi_i^{(k)} f_i(\mathbf{y}_i; \boldsymbol{\theta}_i^{(k)})}{\sum_{h=1}^{g} \pi_h^{(k)} f_h(\mathbf{y}_i; \boldsymbol{\theta}_h^{(k)}))} \end{aligned}$$

for $i = 1, \ldots, g$; $j = 1, \ldots, n$ in a similar fashion to (11).

Substituting this expression into (14), we obtain the final expression for the E-step:

$$Q(\boldsymbol{\Psi}; \boldsymbol{\Psi}^{(k)}) = \sum_{i=1}^{g} \sum_{j=1}^{n} \tau_i(\mathbf{y}_i; \boldsymbol{\Psi}^{(k)}) \{\log \pi_i + \log f_i(\mathbf{y}_j; \boldsymbol{\theta}_i)\}. \tag{15}$$

M-step

The aim of the M-step is to globally maximise $Q(\boldsymbol{\Psi}; \boldsymbol{\Psi}^{(k)})$ w.r.t. $\boldsymbol{\Psi}$ over the parameter space Ω to produce the updated estimate $\boldsymbol{\Psi}^{(k+1)}$. Moreover, the mixture coefficients $\pi_i^{(k+1)}$ are calculated independently from $\boldsymbol{\xi}^{(k+1)}$.

Returning to the hypothetical case where the component labels z_{ij} would be observable. Then the MLE

of π_i would be

$$\hat{\pi}_i = \sum_{j=1}^{n} z_{ij}/n. \tag{16}$$

In the E-step, we replaced each instance of z_{ij} with its current conditional expectation $\tau_i(\mathbf{y}_i; \boldsymbol{\Psi}^{(k)})$ in the complete data log-likelihood. In an analogous fashion, we can transform (16):

$$\pi_i^{(k+1)} = \sum_{j=1}^{n} \tau_i(\mathbf{y}_i; \boldsymbol{\Psi}^{(k)})/n. \tag{17}$$

This substitution is intuitively significant, since we see that each currently assessed posterior probability of membership $\pi_i^{(k+1)}$ contains a contribution from each observed component $\mathbf{y}_i$ in the mixture model. As for the parameter vector $\boldsymbol{\xi}$, we can take the partial derivative of the equation (15) and take one of the roots:

$$\sum_{i=1}^{g} \sum_{j=1}^{n} \tau_i(\mathbf{y}_i; \boldsymbol{\Psi}^{(k)}) \partial \log f_i(\mathbf{y}_j; \boldsymbol{\theta}_i)/\partial \boldsymbol{\xi} = 0. \tag{18}$$

This begs the question of whether the above equation has a closed form solution, since the original problem did not. In many cases, such as in a Gaussian mixture model, we do mostly have closed form solutions.

2.4 Fitting Mixtures of Mixtures: WIP

Based on Section 2.9 in [1], not sure if I should add this to the GMMs section with practical example.

2.5 Starting Values and Initial Parameters: WIP

Based on Section 2.10 in [1]

- EM requires initial estimate of $\boldsymbol{\Psi}^{(0)}$. Starting and stopping strategies can affect the speed of convergence greatly. If likelihood is unbounded on the edge of parameter space Ω, estimates for the $\boldsymbol{\Psi}^{(k)}$s may diverge if initial value is chosen too close to said boundary. Further discussion can be found in [1], Section 3.8.
- Likelihood equation may also have multiple roots, hence it makes sense to have multiple starting points in order to find all local maxima.
- If there aren't any nice ones, choose root with largest MLE out of all of the maxima.
- Mention 3.9.3 [1].

2.6 MAP Estimates: WIP

3 Gaussian Mixture Models

This section is loosely based on Chapters 3.2 and 3.3 in [1].

The most common case of FMMs is where all of the mixture components are multivariate Gaussian. In the context of clustering, it is appropriate to use Gaussian mixture models (GMMs) whenever the clusters appear to have elliptical or symmetric structures. Any non-normal features in the data can then be easily identified and signal some other underlying group structure. Multivariate normal distributions are relatively simple and also make the M-step of the EM algorithm easier to compute. For clusters with heavier tails, we can also consider t-tailed mixture models (WHICH MAY BE DISCUSSED LATER IN THE PAPER).

3.1 Heteroscedastic Components

The original FMM equation (2) expressed with Gaussian mixture components takes the form

$$f(\mathbf{y}_j;\boldsymbol{\Psi}) = \sum_{i=1}^{g} \pi_i \phi(\mathbf{y};\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i), \tag{19}$$

Note that $\boldsymbol{\xi}$ in the above case is comprised of the component means $\boldsymbol{\mu}_i$ and component covariance matrices $\boldsymbol{\Sigma}_i$, known in the heteroscedastic case to be distinct without restrictions.

The EM-Algorithm for GMMs is modified in the following way:

E-step: We modify $\tau_i(\mathbf{y}_i;\boldsymbol{\Psi})$

$$\tau_i(\mathbf{y}_i;\boldsymbol{\Psi}) = \frac{\pi_i \phi(\mathbf{y};\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i)}{\sum_{h=1}^{g} \pi_h \phi(\mathbf{y};\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i)}, \quad i = 1,\ldots,g, \quad j = 1,\ldots,n \tag{20}$$

M-step: the π_is are defined identically to the normal FMMs, see equation (17).

For $\boldsymbol{\mu}_i^{(k+1)}$ and $\boldsymbol{\Sigma}_i^{(k+1)}$ let us define the following notation:

$$\tau_{ij}^{(k)} = \tau_i(\mathbf{y}_j;\boldsymbol{\Psi}^{(k)}), \quad i = 1,\ldots,g, \quad j = 1,\ldots,n. \tag{21}$$

With this notation and using equation (18), replacing every f with ϕ, we obtain the following:

$$\boldsymbol{\mu}_i^{(k+1)} = \sum_{j=1}^{n} \tau_{ij}^{(k)} \mathbf{y}_j \Big/ \sum_{j=1}^{n} \tau_{ij}^{(k)}, \qquad \boldsymbol{\Sigma}_i^{(k+1)} = \sum_{j=1}^{n} \tau_{ij}^{(k)} (\mathbf{y}_j - \boldsymbol{\mu}_i^{(k+1)})(\mathbf{y}_j - \boldsymbol{\mu}_i^{(k+1)})^T \Big/ \sum_{j=1}^{n} \tau_{ij}^{(k)} \tag{22}$$

3.2 Homoscedastic Components

Now we would like to impose that all covariance matrices are identical, i.e.

$$\boldsymbol{\Sigma}_i = \boldsymbol{\Sigma}$$

Then the M-Step simplifies down to

$$\boldsymbol{\Sigma}^{(k+1)} = \sum_{i=1}^{g} \left(\sum_{j=1}^{n} \tau_{ij}^{(k)} \right) \boldsymbol{\Sigma}_i^{(k+1)} / n. \tag{23}$$

The other coefficients π_i and $\boldsymbol{\mu}_i$ are identical to the heteroscedastic case.

4 t-Mixtures: WIP

Based loosely on 8.1 in [2], Chapter 7 in [1], add example with crabs that was mentioned in intro, flexmix

5 Order of a Mixture Model

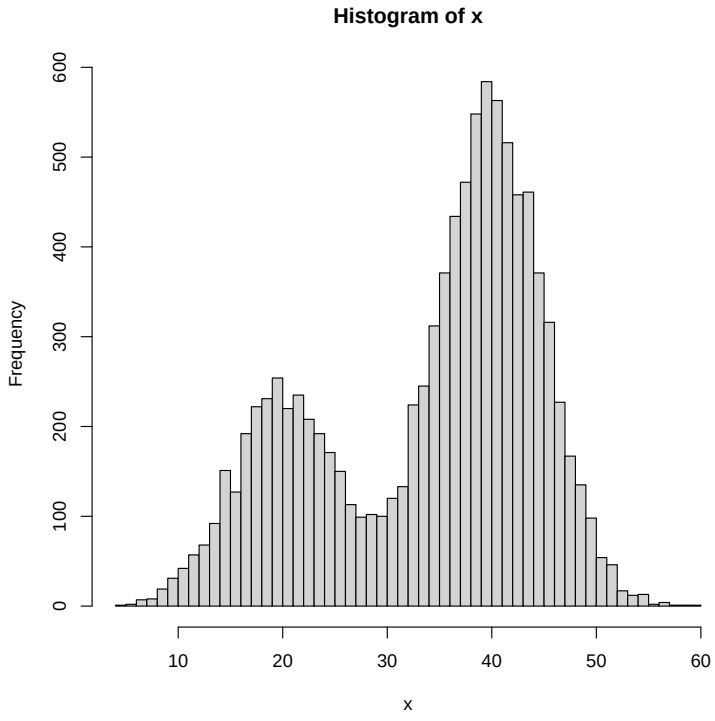
Based on 10.1 in [2].

6 Implementation with Examples in R: Iris Dataset

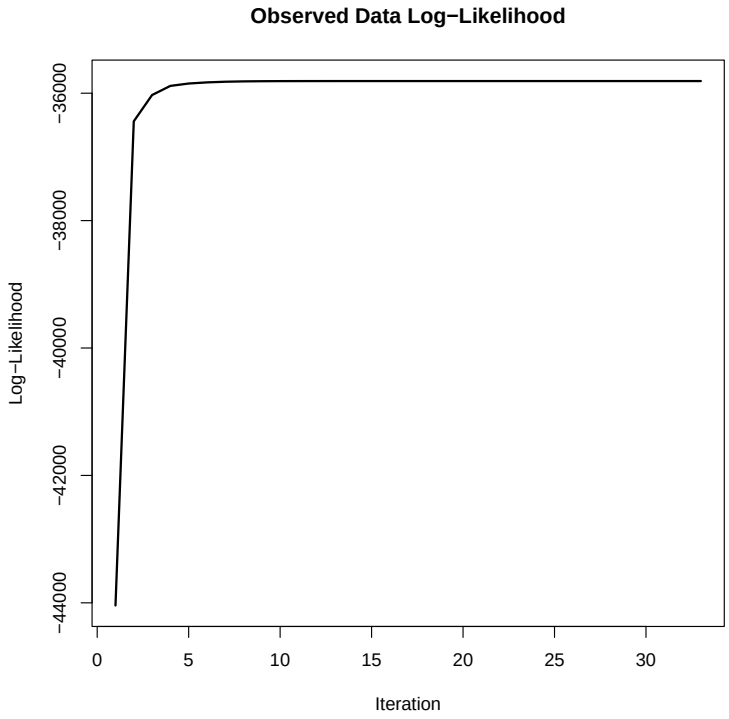
There are multiple packages mentioned in the paper which are often used for finite mixture models and model-based clustering, such as `mixtools`, `EMMIX` (package written by the authors) and `flexmix`. Unfortunately, `EMMIX` is not available on Mac versions of `R`, so we will explore finite mixture models using `mixtools` and `flexmix`.

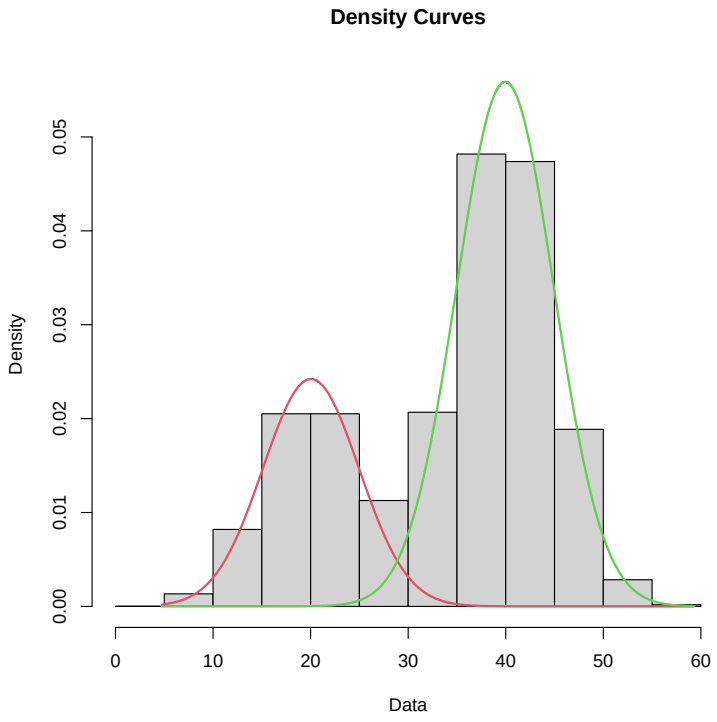
6.1 Primary Example by hand

```
set.seed(123)
X1 = rnorm(3000, 20, 5)
X2 = rnorm(7000, 40, 5)
x = c(X1, X2)
hist(x, nclass=50)


# Using mixtools
library(mixtools)

## mixtools package, version 1.2.0, Released 2020-02-05
## This package is based upon work supported by the National Science Foundation under Grant
No.  SES-0518772.
```




```
test1 <- normalmixEM(x)

## number of iterations= 32

plot(test1, density=TRUE)
```






```r
# Calculating by hand
mem <- kmeans(x,2)$cluster
mu1 <- mean(x[mem==1])
mu2 <- mean(x[mem==2])
sigma1 <- sd(x[mem==1])
sigma2 <- sd(x[mem==2])
pi1 <- sum(mem==1)/length(mem)
```

```r
pi2 <- sum(mem==2)/length(mem)

# modified sum only considers finite values
sum.finite <- function(x) {
  sum(x[is.finite(x)])
}

Q <- 0
# starting value of expected value of the log likelihood
Q[2] <- sum.finite(log(pi1)+log(dnorm(x, mu1, sigma1)))
  + sum.finite(log(pi2)+log(dnorm(x, mu2, sigma2)))
```

```
## [1] -95411.23
```

```r
k <- 2

while (abs(Q[k]-Q[k-1])>=1e-6) {
  # E step
  comp1 <- pi1 * dnorm(x, mu1, sigma1)
  comp2 <- pi2 * dnorm(x, mu2, sigma2)
  comp.sum <- comp1 + comp2

  p1 <- comp1/comp.sum
  p2 <- comp2/comp.sum

  # M step
  pi1 <- sum.finite(p1) / length(x)
  pi2 <- sum.finite(p2) / length(x)

  mu1 <- sum.finite(p1 * x) / sum.finite(p1)
  mu2 <- sum.finite(p2 * x) / sum.finite(p2)

  sigma1 <- sqrt(sum.finite(p1 * (x-mu1)^2) / sum.finite(p1))
  sigma2 <- sqrt(sum.finite(p2 * (x-mu2)^2) / sum.finite(p2))

  p1 <- pi1
  p2 <- pi2

  k <- k + 1
  Q[k] <- sum(log(comp.sum))
}
```

References

[1] David Peel Geoffery J. McLachlan. *Finite Mixture Models*. Wiley, 2000.

[2] Suren I. Rathnayake Geoffery J. McLachlan Sharon X. Lee. "Finite Mixture Models". In: *Annual Review of Statistics and Its Application* (2019). DOI: https://doi.org/10.1146/annurev-statistics-031017-100325.

[3] Thriyambakam Krishnan Geoffery J. McLachlan. *The EM Algorithm and Extensions*. Wiley, 1997.