An Introduction to mixR

Youjiao Yu

June 01, 2021

Abstract

The package **mixR** performs maximum likelihood estimation (MLE) for finite mixture models for families including Normal, Weibull, Gamma and Lognormal via EM algorithm. It also conducts model selection by using Bayesian Information Criterion (BIC) or bootstrap likelihood ratio test (LRT). The data used for mixture model fitting can be raw data or binned data. The model fitting is accelerated by using R package **Rcpp**.

Contents

1 Background

1.1 Mixture models

Finite mixture models can be represented by

$$f(x;\Phi) = \sum_{j=1}^{g} \pi_j f_j(x;\theta_j)$$

where $f(x;\Phi)$ is the probability density function (p.d.f.) or probability mass function (p.m.f.) of the mixture model, $f_j(x;\theta_j)$ is the p.d.f. or p.m.f. of the jth component of the mixture model, π_j is the proportion of the jth component, θ_j is the parameter of the jth component which can be a scalar or a vector, $\Phi = (\pi_1, \theta_1, \ldots, \pi_g, \theta_g)$ is a vector of all the parameters in the mixture model, and g is the total number of components in the mixture model. The MLE of Φ can be obtained using the EM algorithm (Dempster, Laird, and Rubin 1977).

1.2 Mixture model selection by BIC

One critical problem for a mixture model is how to estimate g when there is no such *a priori* knowledge. As EM algorithm doesn't estimate g itself, a commonly used approach to estimate g is to fit a series of mixture models with different values of g and then select g using information criteria such as Akaike Information Criterion (AIC), Bayesian Information Criterion (BIC), Deviance Information Criterion (DIC), or Integrated Complete-data Likelihood (ICL). Among all information criteria, BIC has shown to outperform other ones in model selection (Steele and Raftery 2010). BIC is defined as

$$BIC = k\log(n) - 2\log(\hat{L})$$

in which k is the total number of parameters in the mixture model, n is the size of data, and $\hat{L}$ is the estimated maximum likelihood of the model. The model which has the lowest BIC is regarded as the optimal one.

1.3 Mixture model selection by bootstrap LRT

A mixture model with $g = g_1$ components is a nested model of a mixture model with $g = g_2 (g_1 < g_2)$ components, as the former model can be regarded as the later one with $\pi_j = 0$ for $g_2 - g_1$ components and $p_j > 0$ for all the remaining g_1 components. LRT is a common tool for assessing the goodness of fit of the nested model ($H_0 : g = g_1$) compared to the full model ($H_a : g = g_2$). However the regularity condition of the LRT, which requires that the parameter space of the model in the null hypothesis H_0 should lie in the interior of the parameter space of the model in the alternative hypothesis H_a, doesn't hold for the mixture models (Feng and McCulloch 1996), and therefore the test statistic of LRT, denoted as $w(x)$ doesn't follow a known Chi-square distribution under H_0. McLachlan (1987) proposed the idea of applying the method of bootstrapping (Efron and Tibshirani 1994) for approximating the distribution of $w(x)$. The general steps of bootstrap LRT are as follows.

1. For the given data x, estimate Φ under both H_0 and H_a to get $\hat{\Phi}_0$ and $\hat{\Phi}_1$. Calculate the observed log-likelihood $\ell(x; \hat{\Phi}_0)$ and $\ell(x; \hat{\Phi}_1)$. The LRT statistic is defined as $w_0 = -2(\ell(x; \hat{\Phi}_0) - \ell(x; \hat{\Phi}_1))$.
2. Generate random data of the same size as the original data x from the model under the H_0 using estimated parameter $\hat{\Phi}_0$, then repeat step 1 using the simulated data. Repeat this process for B times to get a vector of the simulated likelihood ratio test statistics $w_1^{(1)}, \ldots, w_1^{(B)}$.
3. Calculate the empirical p-value as

$$p = \frac{1}{B}\sum_{i=1}^{B} I(w_1^{(i)} > w_0)$$

where $I(\cdot)$ is the indicator function.

1.4 Fitting mixture models to the binned data

The binned data is present instead of the raw data in some situations, often for the reason of storage convenience or necessity. The binned data is recorded in the form of (a_i, b_i, n_i) where a_i is the left bin value of the i^{th} bin, b_i is the right bin value of the i^{th} bin, and n_i is the number of observations that fall in the i^{th} bin for $i = 1, \ldots, r$, where r is the total number of bins.

The MLE for finite mixture models fitted to binned data can also be obtained via EM algorithm by introducing an additional latent variable x that represents the unknown value of the raw data, besides the usually latent variable z that represents the component an observation belongs to. To apply the EM algorithm we first write the complete-data log-likelihood as

$$Q(\Phi; \Phi^{(p)}) = \sum_{j=1}^{g}\sum_{i=1}^{r} n_i z^{(p)} [\log f(x^{(p)}; \theta_j) + \log \pi_j]$$

where $z^{(p)}$ is the expected value of z given $\Phi^{(p)}$ and $x^{(p)}$, the estimated value of Φ and expected value of x at p^{th} iteration. The estimate of Φ can be updated alternatively via an E-step, in which we estimate Φ by

maximizing $Q(\Phi; \Phi^{(p)})$, and an M-step, in which we compute $x^{(p)}$ and $z^{(p)}$, until the convergence of the EM algorithm. The M-step may not have a closed-form solution, e.g. in the Weibull mixture model or Gamma mixture model, which if is the case, an iterative approach like Newton's algorithm or bisection method may be used.

1.5 Beyond normality

The normal distribution is mostly used in a mixture model for continuous data, but there are also circumstances when other distributions fit the data better. McLachlan and Peel (2004) explained that a limitation of the normal distribution is that when the shapes of the components are skewed, there may not be a one-to-one correspondence between the number of components in the mixture model and that in the data. More than one normal component is needed to model a skewed component, which may cause overestimation of g. For skewed or asymmetric components, other distributions such as Gamma, Log-normal or Weibull might provide better model fitting than the normal distribution in a mixture model. As an example, Yu and Harvill (2019) demonstrated two examples where Weibull mixture models are preferred.

2 mixR package

We present the functions in **mixR** package for (a) fitting finite mixture models for continuous data for families including Normal, Weibull, Gamma and Log-normal via EM algorithm; (b) selecting the optimal number of components for a mixture model using BIC or bootstrap LRT. We also discuss how to fit mixture models with binned data.

2.1 Model fitting

The function `mixfit()` can be used to fit mixture models for four different families – Normal, Weibull, Gamma, and Log-normal. For Normal distribution, the variances of each component are the same by setting `ev = TRUE`.

```
# generate data from a Normal mixture model
library(mixR)
set.seed(102)
x1 = rmixnormal(1000, c(0.3, 0.7), c(-2, 3), c(2, 1))

# fit a Normal mixture model (unequal variances)
mod1 = mixfit(x1, ncomp = 2); mod1
#> Normal mixture model with 2 components
#>         comp1     comp2
#> pi  0.2882564 0.7117436
#> mu -2.2742464 2.9863952
#> sd  1.8172431 0.9636397
#>
#> EM iterations: 25 AIC: 4213.68 BIC: 4238.22 log-likelihood: -2101.84

# fit a Normal mixture model (equal variance)
mod1_ev = mixfit(x1, ncomp = 2, ev = TRUE); mod1_ev
#> Normal mixture model with 2 components
#>         comp1     comp2
#> pi  0.2470491 0.7529509
#> mu -2.7354208 2.8498073
#> sd  1.2198083 1.2198083
#>
#> EM iterations: 10 AIC: 4289.59 BIC: 4309.22 log-likelihood: -2140.8
```

```
plot(mod1, title = 'Normal Mixture Model (unequal variances)')
plot(mod1_ev, title = 'Normal Mixture Model (equal variance)')
```

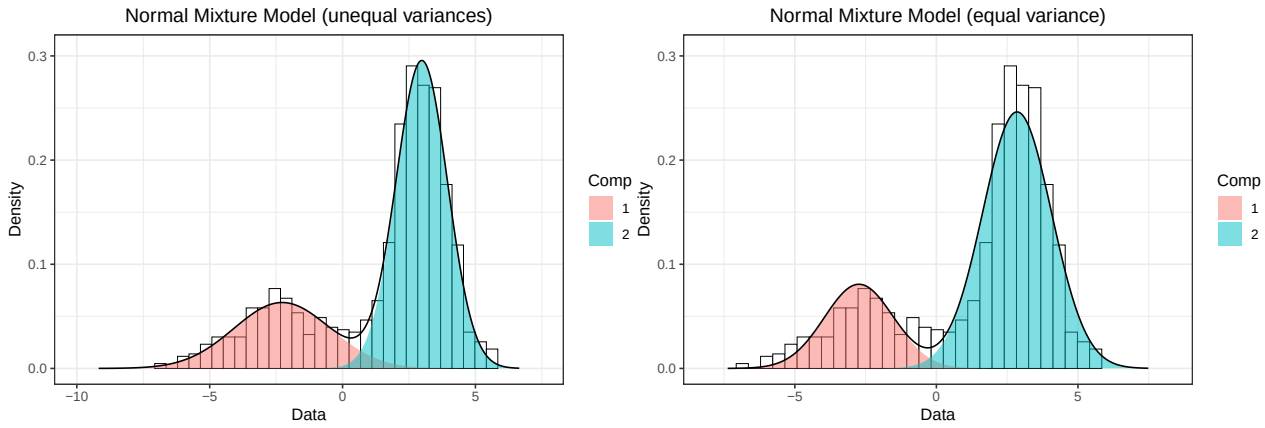

Figure 1: The fitted normal mixture model with unequal variances (left) and equal variance(right)

The initial values for Φ are estimated by k-means or hierarchical clustering method if they are not provided by the users. In situations when EM algorithm is stuck in a local minimum and leads to unsatisfactory fitting results, which happens more likely when the number of components g and/or data size n are large, initial values can be provided manually to get a better fitting.

To illustrate the idea that g tends to be over-estimated when using a normal mixture model to fit data with asymmetric or skewed components, we simulate data from a Weibull mixture model with $g = 2$, then fit both Normal and Weibull mixture models to the data. First we fit both models with $g = 2$. Weibull distribution provides better fitting than Normal by either visually checking the plots of the fitted results in Figure 2, or the fact that the log-likelihood of the fitted Weibull mixture model (244) is much higher than that of the fitted Normal mixture model (200). Figure 3 shows that the best value of g for Weibull mixture model is two and for Normal mixture model is four, higher than the actual value of g.

```
x2 = rmixweibull(1000, c(0.4, 0.6), c(0.6, 1.3), c(0.1, 0.1))
mod2_weibull = mixfit(x2, family = 'weibull', ncomp = 2); mod2_weibull
#> Weibull mixture model with 2 components
#>            comp1      comp2
#> pi     0.3637416  0.6362584
#> mu     0.6072458  1.3017602
#> sd     0.1046977  0.0970958
#> shape 6.8080322 16.5082764
#> scale 0.6501023  1.3441343
#>
#> EM iterations: 4 AIC: -519.07 BIC: -494.53 log-likelihood: 264.53
mod2_normal = mixfit(x2, ncomp = 2); mod2_normal
#> Normal mixture model with 2 components
#>         comp1     comp2
#> pi 0.3685599 0.6314401
#> mu 0.6112804 1.3037704
#> sd 0.1109374 0.0957956
#>
#> EM iterations: 10 AIC: -429.92 BIC: -405.39 log-likelihood: 219.96
plot(mod2_weibull)
plot(mod2_normal)
```

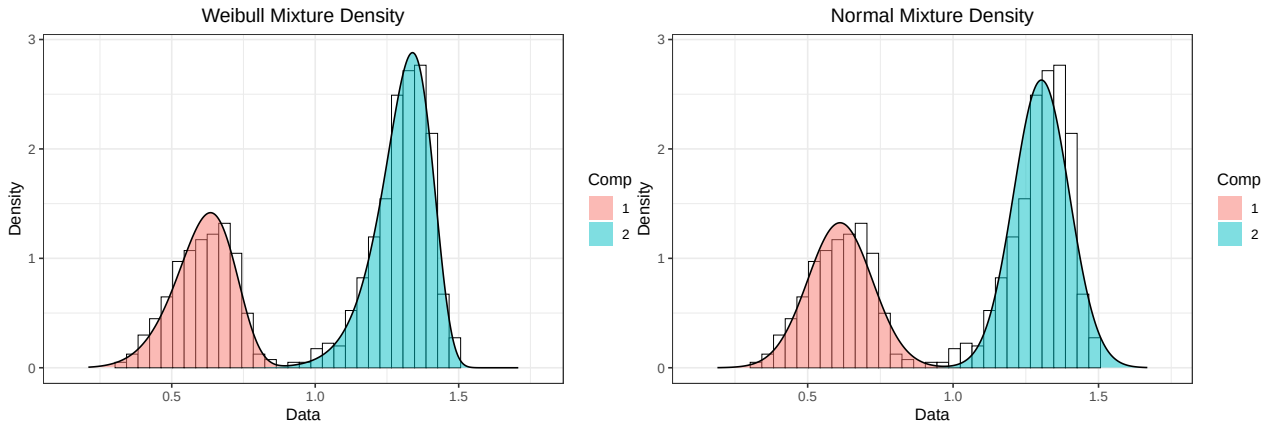

Figure 2: The fitted Weibull mixture model (left) and Normal mixture model (right) to the same data

2.2 Model selection by BIC

The function `select()` is used to fit a series of finite mixture models with values of g specified in `ncomp`, and then select the best g by BIC. For normal mixture models, both equal and unequal variances are considered. Figure 3 shows the value of BIC for normal and Weibull mixture models with different g. For Weibull mixture models, BIC increases monotonically as g increases from two to six, therefore the best value of g is two. For normal mixture models, BIC decreases first when g goes from two to four, and then increases when g goes from four to six, which is true for both equal variance and unequal variances. The best model is $g = 4$ with equal variance as its BIC is the lowest. Figure 4 shows the fitted Weibull mixture models and normal mixture model with the best values of g.

```
# Selecting the best g for Weibull mixture model
s_weibull = select(x2, ncomp = 2:6, family = 'weibull')

# Selecting the best g for Normal mixture model
s_normal = select(x2, ncomp = 2:6)
plot(s_weibull)
plot(s_normal)
```

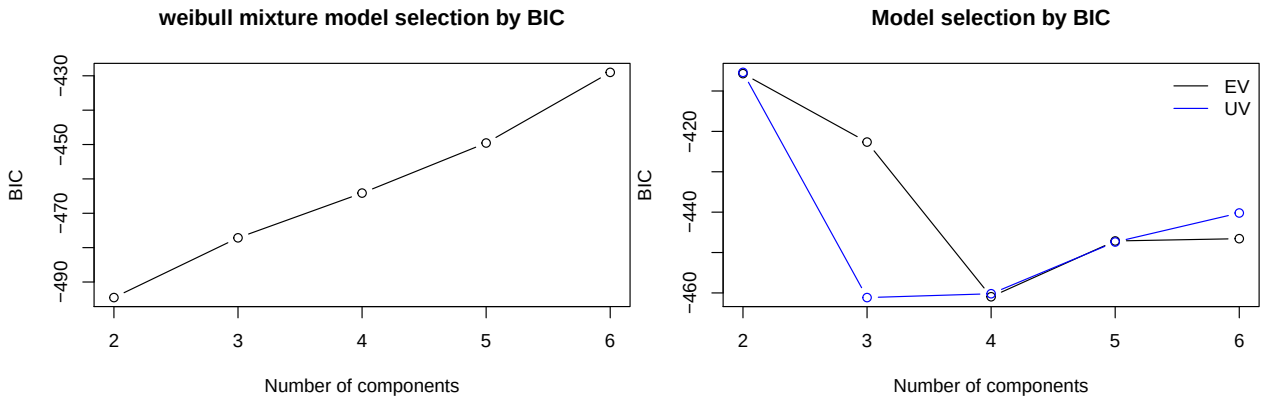

Figure 3: The value of BIC for Weibull mixture models (left) and Normal mixture models (right) with different values of g.

```
plot(mod2_weibull)
plot(mixfit(x2, ncomp = 4, ev = TRUE))
```

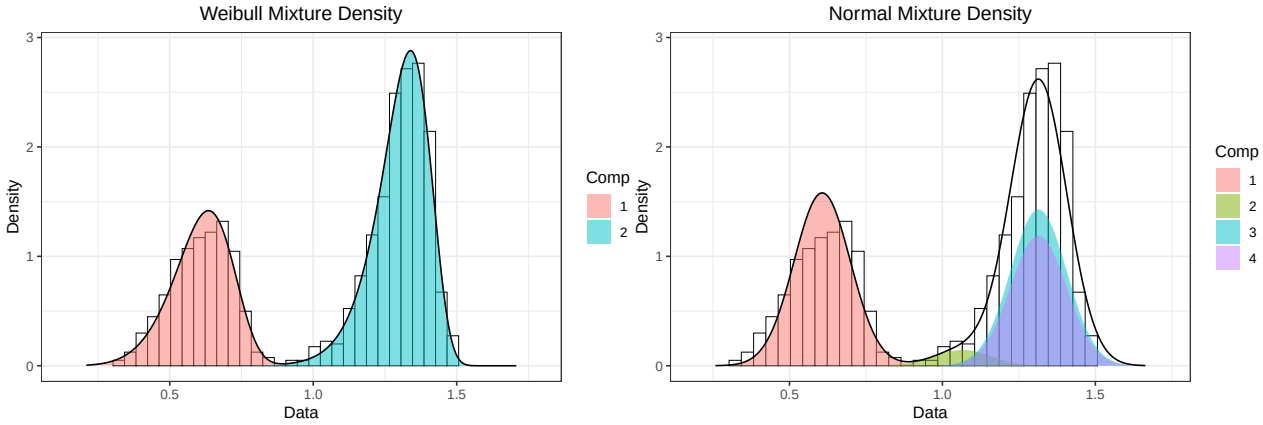

Figure 4: The fitted Weibull mixture model with $g = 2$ (left) and the Normal mixture model with $g = 4$ and equal variance (right)

2.3 Model selection by bootstrap LRT

The function `bs.test()` performs bootstrap LRT and returns the p-value as well as the test statistics w_0 and w_1. As an example, the data set `x1` above are generated from a Normal mixture model with $g = 2$. If we conduct bootstrap LRT for $g = 2$ against $g = 3$ for `x1` and set the number of bootstrap iterations `B=100`, we get p-value of 0.48, showing that the Normal mixture model with three components is not any better than the one with two components for data `x1`.

As another example, the data set `x2` above are generated from a Weibull mixture model with $g = 2$. We discussed previously that if we use Normal distribution to fit the mixture model, the best value of g selected by BIC is four. A bootstrap LRT of $g = 2$ against $g = 4$ returns zero p-value, indicating that if we fit a Normal mixture model to `x2`, $g = 4$ is a much better fit than $g = 2$, though visually the data shows two modes rather than four. Figure 5 shows the histogram of w_1 and the location of w_0 (red vertical line) for the above two examples.

```
b1 = bs.test(x1, ncomp = c(2, 3))
plot(b1, main = 'Bootstrap LRT for Normal Mixture Models (g = 2 vs g = 3)')
b1$pvalue
#> [1] 0.47
b2 = bs.test(x2, ncomp = c(2, 4))
plot(b2, main = 'Bootstrap LRT for Normal Mixture Models (g = 2 vs g = 4)')
b2$pvalue
#> [1] 0
```

2.4 Mixture model fitting with binned data

The function `mixfit()` can also fit mixture models with binned data, in the form of a three-column matrix each row of which represents a bin with the left bin value, the right bin value, and the total number of data points that fall in each bin (analogous to the data used to create a histogram). The function `bin()` is used to create binned data from raw data, and the function `reinstate()` can simulate the raw data from binned data. Figure 6 shows the mixture models fitted with data binned from raw data `x1` and `x2`, with 30 bins for each data set.

```
x1_binned = bin(x1, seq(min(x1), max(x1), length = 30))
head(x1_binned, 3)
#>              a         b freq
#> [1,] -7.070279 -6.625030    2
#> [2,] -6.179782 -5.734534    7
```

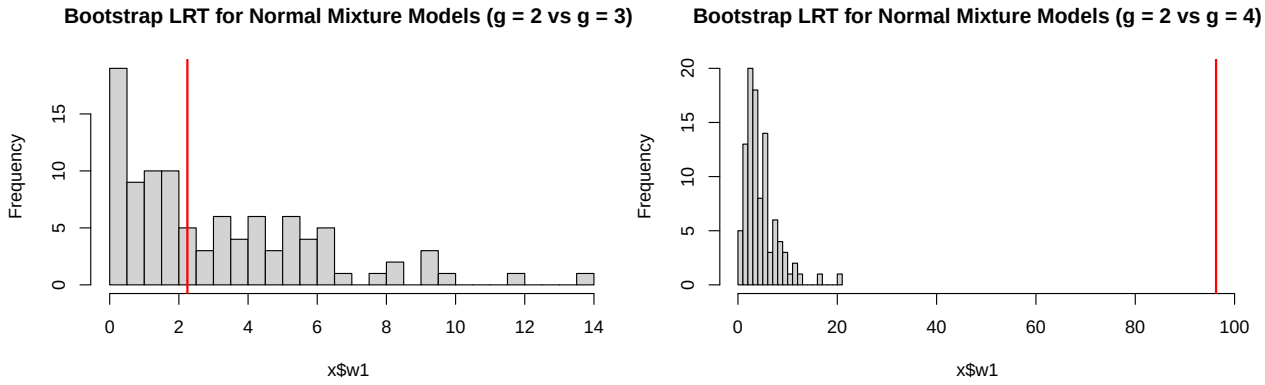

Figure 5: (left) The bootstrap LRT of $H_0 : g = 2$ against $H_1 : g = 3$ for fitting Normal mixture models for data 'x1'; (right) The bootstrap LRT of $H_0 : g = 2$ against $H_1 : g = 4$ for fitting Normal mixture models for data 'x2'. In each plot the histogram shows the distribution of w_1 and the red line shows the value of w_0.

```
#> [3,] -5.734534 -5.289286    6

mod1_binned = mixfit(x1_binned, ncomp = 2)
plot(mod1_binned, xlab = 'x1_binned',
     title = 'The Normal Mixture Model Fitted With Binned Data')
mod1_binned
#> Normal mixture model with 2 components
#>         comp1     comp2
#> pi  0.2872466 0.7127534
#> mu -2.2952167 2.9887117
#> sd  1.8010692 0.9593141
#>
#> EM iterations: 34 AIC: 5848.1 BIC: 5872.64 log-likelihood: -2919.05

x2_binned = bin(x2, seq(min(x2), max(x2), length = 30))
head(x2_binned, 3)
#>              a         b freq
#> [1,] 0.3024235 0.3439529    2
#> [2,] 0.3439529 0.3854824    5
#> [3,] 0.3854824 0.4270118   12

mod2_binned = mixfit(x2_binned, ncomp = 2, family = 'weibull')
plot(mod2_binned, xlab = 'x2_binned',
     title = 'The Weibull Mixture Model Fitted With Binned Data')
mod2_binned
#> Weibull mixture model with 2 components
#>           comp1      comp2
#> pi    0.3638721  0.6361279
#> mu    0.6079659  1.3020818
#> sd    0.1036559  0.0963527
#> shape 6.8906250 16.6448242
#> scale 0.6504611  1.3441444
#>
#> EM iterations: 6 AIC: 5863.35 BIC: 5887.89 log-likelihood: -2926.67
```

As binning can be considered a way to compress data, binned data can accelerate the fitting of mixture

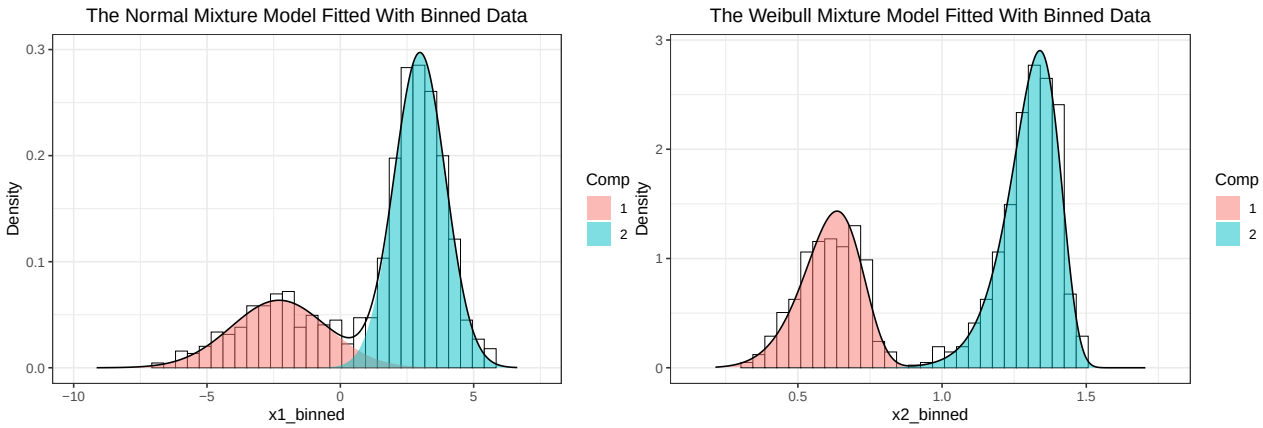

Figure 6: (left) The Normal mixture model fitted with binned data; (right) The Weibull mixture model fitted with binned data.

models, especially when the original data set is large. To illustrate, we simulate 100,000 data points from a Normal mixture model with five components, and bin the data with 100 bins. Normal mixture models are fitted on both the simulated raw data and binned data. The results show that model fitting takes 27 seconds on raw data, and less than one second on binned data. Another example shows that fitting a Weibull mixture model with data binned from a data set with one million observations takes just over two seconds [1].

```
# a function to generate parameters for a mixture model
generate_params = function(ncomp = 2) {
  pi = runif(ncomp)
  low = runif(1, 0, 0)
  upp = low + runif(1, 0, 10)
  mu = runif(ncomp, low, upp)
  sd = runif(ncomp, (max(mu) - min(mu))/ncomp/10, (max(mu) - min(mu))/ncomp/2)
  list(pi = pi / sum(pi), mu = sort(mu), sd = sd)
}

# simulate data from a Normal mixture model
set.seed(988)
n = 100000
ncomp = 5
params = generate_params(ncomp)
x_large = rmixnormal(n, pi = params$pi, mu = params$mu, sd = params$sd)

# fitting a Normal mixture model with raw data
t1 = Sys.time()
mod_large <- mixfit(x_large, ncomp = ncomp)
t2 = Sys.time()
t2 - t1
#> Time difference of 13.79882 secs

plot(mod_large, title = 'Normal Mixture Model Fitted With Raw Data')
mod_large
#> Normal mixture model with 5 components
#>         comp1     comp2     comp3     comp4     comp5
#> pi 0.2808553 0.1288440 0.1485099 0.2384039 0.2033869
```

[1] evaluated on iMac with processor: 3 GHz Quad-Core Intel Core i5, memory: 8 GB 2400 MHz DDR4

```
#> mu 0.0266805 0.6152623 1.1103667 1.5788525 3.4490338
#> sd 0.0865347 0.1714647 0.1256346 0.2206441 0.2436014
#>
#> EM iterations: 225 AIC: 198090.29 BIC: 198223.47 log-likelihood: -99031.14

# fitting a Normal mixture model with binned data
t3 = Sys.time()
x_binned = bin(x_large, seq(min(x_large), max(x_large), length = 100))
mod_binned <- mixfit(x_binned, ncomp = ncomp)
t4 = Sys.time()
t4 - t3
#> Time difference of 0.566231 secs

plot(mod_binned, title = 'Normal Mixture Model Fitted With Binned Data')
mod_binned
#> Normal mixture model with 5 components
#>         comp1     comp2     comp3     comp4     comp5
#> pi 0.2810244 0.1273384 0.1544025 0.2338445 0.2033902
#> mu 0.0268700 0.6129720 1.1125285 1.5852661 3.4490635
#> sd 0.0853756 0.1683686 0.1288791 0.2164886 0.2431581
#>
#> EM iterations: 364 AIC: 804626.59 BIC: 804759.77 log-likelihood: -402299.3
```

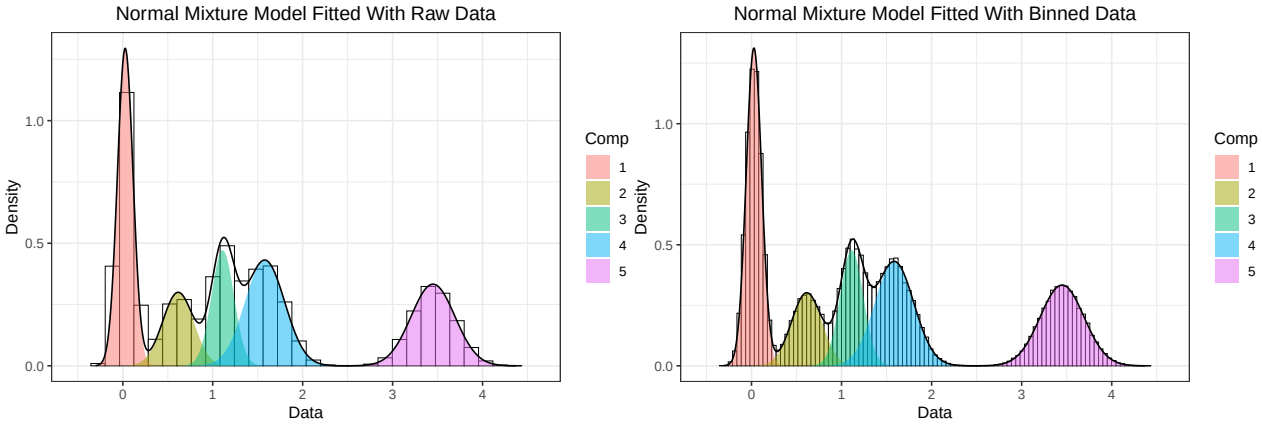

Figure 7: Normal mixture models fitted to the raw data (left) and binned data (right)

3 Citation

Run `citation(package = 'mixR')` to see how to cite package **mixR** in publications.

4 References

Dempster, Arthur P, Nan M Laird, and Donald B Rubin. 1977. "Maximum Likelihood from Incomplete Data via the EM Algorithm." *Journal of the Royal Statistical Society: Series B (Methodological)* 39 (1): 1–22.

Efron, Bradley, and Robert J Tibshirani. 1994. *An Introduction to the Bootstrap*. CRC press.

Feng, Ziding D, and Charles E McCulloch. 1996. "Using Bootstrap Likelihood Ratios in Finite Mixture Models." *Journal of the Royal Statistical Society: Series B (Methodological)* 58 (3): 609–17.

McLachlan, Geoffrey J. 1987. "On Bootstrapping the Likelihood Ratio Test Statistic for the Number of Components in a Normal Mixture." *Journal of the Royal Statistical Society: Series C (Applied Statistics)* 36 (3): 318–24.

McLachlan, Geoffrey J, and David Peel. 2004. *Finite Mixture Models*. John Wiley & Sons.

Steele, Russell J, and Adrian E Raftery. 2010. "Performance of Bayesian Model Selection Criteria for Gaussian Mixture Models." *Frontiers of Statistical Decision Making and Bayesian Analysis* 2: 113–30.

Yu, Youjiao, and Jane L Harvill. 2019. "Bootstrap Likelihood Ratio Test for Weibull Mixture Models Fitted to Grouped Data." *Communications in Statistics-Theory and Methods* 48 (18): 4550–68.