rebmix: Finite Mixture Modeling, Clustering & Classification

Marko Nagode, Branislav Panić, Jernej Klemenc & Simon Oman

June 8, 2020

Abstract

The **rebmix** package provides R functions for random univariate and multivariate finite mixture model generation, estimation, clustering, latent class analysis and classification. Variables can be continuous, discrete, independent or dependent and may follow normal, lognormal, Weibull, gamma, binomial, Poisson, Dirac or von Mises parametric families.

1 Introduction

To cite the REBMIX algorithm please refer to (Nagode and Fajdiga, 2011a,b; Nagode, 2015, 2018). For theoretical backgrounds please upload `http://doi.org/10.5963/JAO0302001`.

2 What's new in version `2.12.0`

Version `2.12.0` introduces the Knuth algorithm (Knuth, 2019) as an effective way of optimal number of bins search. This affects the time efficiency of the `REBMIX` method considerably. The user can now enter different numbers of bins for different random variables. Two accompanied methods are added `optbins` and `bins`. The C++ code is optimized regarding memory allocation and efficiency. The R code is debugged and further improved regarding wrong user input messaging. Further debugging has been done. Outlier detection has been simplified. The `REBMIX` method now delivers some more components, but identifies also components with very low probability of occurrence, which is important regarding our future plans. The `plot` method is improved. The `RCLRMIX` method has been debugged and improved. The same holds for the `REBMIX` method.

3 Previous versions

Version `2.11.0` introduces the Expectation-Maximization (EM) algorithm for the improved estimation of Gaussian mixture model parameters (with diagonal and unrestriced covariance matrices). Here the REBMIX algorithm is used to assess the initial parameters of the EM algorithm. Two different variants of the EM algorithm are implemented, namely the original EM algorithm from (Dempster et al., 1977) and a k-means like variant of the EM algorithm (Classification EM) as described in (Celeux and Govaert, 1992). As the REBMIX algorithm estimates a wide range of parameters for the Gaussian mixture model for different numbers of components, three different strategies, named **exhaustive**, **best** and **single**, have been implemented. The **exhaustive** strategy is used to run the EM algorithm (or variant) on each solution of Gaussian mixture model parameters provided by the REBMIX algorithm. The **best** strategy utilizes a voting scheme for the estimated parameters from the REBMIX algorithm and runs the EM algorithm only on selected optimal parameters. The best candidates are chosen based on the value of the likelihood function (the highest one) for each number of components c from a minimum specified `cmin` to a maximum specified `cmax`. The **single** strategy is useful when the single value of, for example, the number of bins in histogram preprocessing is supplied as input for the REBMIX algorithm. Otherwise, when multiple numbers of bins k are supplied, this strategy is the same as the **exhaustive** strategy. To tackle the slow linear convergence of the EM algorithm, simple acceleration methods are implemented, which can be controlled with parameter `acceleration` and `acceleration.multiplier`. The increment of the EM algorithm in each iteration can be written as

$$\varDelta\mathbf{\Theta} = \mathbf{\Theta}^{(i+1)} - \mathbf{\Theta}^{(i)} \tag{1}$$

Instead of using a standard EM increment $\varDelta\mathbf{\Theta}$ to reduce the number of iterations needed for the EM algorithm, this increment can be multiplied with some multiplier a_{EM}, which is referred to as `acceleration.multiplier`. Therefore the update in each EM iteration now becomes

$$\mathbf{\Theta}^{i+1} = \mathbf{\Theta}^{(i)} + a_{\mathrm{EM}}\varDelta\mathbf{\Theta} \tag{2}$$

The safe range for the a_{EM} multiplier lies between 1.0 and 2.0, where 1.0 gives a standard EM increment and 2.0 doubles the EM increment. However, this does not necessarily mean that multiplication by a value of 2.0 will double the speed of the EM algorithm (i.e. by reducing the required number of iterations by 2). Here, 1.5 is a safe value which mostly speeds up the EM algorithm whilst retaining good results for the estimated parameters. A value of 1.9 can significantly speed up the estimation process, yet it can also deteriorate the quality of the resulting estimated parameters. Therefore, the value of the multiplicator needs to be set carefully. This value is set with `acceleration.multiplier` parameter. The other parameter `acceleration` controls how the a_{EM} multiplier is handled and can be one of **fixed**, **line** and **golden**. Selecting the **fixed** option means that the a_{EM} multiplier is specified via the `acceleration.multiplier` parameter and for each iteration of the EM algorithm the increment is increased by a specified value of a_{EM}. The **line** and **golden** options perform a, line and golden search (respectively) for the optimal value of a_{EM} for which the highest increase in the likelihood function of each EM iteration is achieved.

EM handling is carried out using the newly introduced class `"EM.Control"`. Classes `"REBMIX"` and `"REBMVNORM"` and its signature method `REBMIX` now accept the `"EM.Control"` object via the argument called `"EMcontrol"`. The class `EM.Control` has the same name convection for slots as the input argument `EMcontrol` (`strategy`, `variant`, `acceleration`, `tolerance`, `acceleration.multiplier` and `maximum.iterations`) as well as all accessor functions with the same name convention as `a.`*`slot name`* and setter function `a.`*`slot name`*`<-`.

Methods `Zp` and `coef` have been replaced by `a.Zp`, `a.theta1.all` and `a.theta2.all` getters. All slots can be accessed via accessors. Their names are generally composed of `a.` followed by the slot name and are used to read the slots. Class `"RNGMIX.Theta"` has been added to simplify random finite mixture model generation. Method `show` has been added for `"RCLS.chunk"` class. The minimum number of components `cmin` was added to `REBMIX` arguments and to the `"REBMIX"` class. The `"Parzen window"` preprocessing has been renamed to more commonly known `"kernel density estimation"`. Rough parameter estimation for binomial and Poisson parametric families has also been improved and the package is now broadened to latent class analysis in version 2.10.3. Method `split` has been improved and examples for its proper use are added.

GCC 8.1 notes and warnings in C++ functions have been eliminated in version 2.10.2. Cholesky decomposition is now used to calculate the logarithm of the determinant and inverse of variance-covariance matrices instead of LU decomposition. Special attention has been paid to resolving numerical problems related to high dimensional datasets.

Version 2.10.1 is the further debugged version of 2.10.0. Large `K` in combination with large dimension d can lead to histograms with numerous nonempty bins v. In order to restrain v, the well known RootN rule (Velleman, 1976) may intuitively be extended to multidimensions

$$v_{\max} = \frac{1+d}{d} n^{\frac{d}{1+d}}. \tag{3}$$

If $d = \infty$, then $v_{\max} = n$. If $d = 1$, then $v_{\max} = 2\sqrt{n}$. Minor debugging and function improvements have also been carried out in version 2.10.0. The acceleration rate is now progressively increasing. Each time the inner loop starts, the counter I_2 (see Nagode, 2015, for details) is initiated and constant

$$A = \left.\frac{1 - a_{\mathrm{r}}}{a_{\mathrm{r}}(D_l w_l - D_{\min})}\right|_{I_2=1} \tag{4}$$

is calculated. The acceleration rate a_{r} at $I_2 = 1$ always equals the value stored in the input argument `ar`. Otherwise

$$a_{\mathrm{r}} = \left. \frac{1}{A(D_l w_l - D_{\min}) + 1} \right|_{I_2 > 1}. \tag{5}$$

The Newton-Raphson root finding in C++ functions was improved in version 2.9.3. This affects only Weibull, gamma and von Mises parametric families. A circular von Mises parametric family has been added and further debugging carried out in version 2.9.2. Version 2.9.1 is a further debugged version 2.8.4. The R code has been extended and rewritten in S4 class system. The background C code has also been extended and rewritten as object-oriented C++ code. The package can now more easily be extended to other parametric families. Multivariate normal mixtures with unrestricted variance-covariance matrices have been added. Clustering has also been added and classification improved.

4 Examples

To illustrate the use of the REBMIX algorithm, univariate and multivariate datasets are considered. The **rebmix** is loaded and the prompt before starting new page is set to `TRUE`.

```
R> library("rebmix")
R> devAskNewPage(ask = TRUE)
```

4.1 Gamma datasets

Three gamma mixtures are considered (Wiper et al., 2001). The first has four well-separated components with means 2, 4, 6 and 8, respectively

$$\begin{array}{lll} \theta_1 = 1/100 & \beta_1 = 200 & n_1 = 100 \\ \theta_2 = 1/100 & \beta_2 = 400 & n_2 = 100 \\ \theta_3 = 1/100 & \beta_3 = 600 & n_3 = 100 \\ \theta_4 = 1/100 & \beta_4 = 800 & n_4 = 100. \end{array}$$

The second has equal means but different variances and weights

$$\begin{array}{lll} \theta_1 = 1/27 & \beta_1 = 9 & n_1 = 40 \\ \theta_2 = 1/270 & \beta_2 = 90 & n_2 = 360. \end{array}$$

The third is a mixture of a rather diffuse component with mean 6 and two lower weighted components with smaller variances and means of 2 and 10, respectively

$$\begin{array}{lll} \theta_1 = 1/20 & \beta_1 = 40 & n_1 = 80 \\ \theta_2 = 1 & \beta_2 = 6 & n_2 = 240 \\ \theta_3 = 1/20 & \beta_3 = 200 & n_3 = 80. \end{array}$$

4.1.1 Finite mixture generation

```
R> n <- c(100, 100, 100, 100)
R> Theta <- new("RNGMIX.Theta", c = 4, pdf = "gamma")
R> a.theta1(Theta) <- rep(1/100, 4)
R> a.theta2(Theta) <- c(200, 400, 600, 800)
R> gamma1 <- RNGMIX(Dataset.name = "gamma1", n = n, Theta = a.Theta(Theta))
R> n <- c(40, 360)
R> Theta <- new("RNGMIX.Theta", c = 2, pdf = "gamma")
R> a.theta1(Theta) <- c(1/27, 1/270)
R> a.theta2(Theta) <- c(9, 90)
R> gamma2 <- RNGMIX(Dataset.name = "gamma2", n = n, Theta = a.Theta(Theta))
R> n <- c(80, 240, 80)
R> Theta <- new("RNGMIX.Theta", c = 3, pdf = "gamma")
```

```
R> a.theta1(Theta) <- c(1/20, 1, 1/20)
R> a.theta2(Theta) <- c(40, 6, 200)
R> gamma3 <- RNGMIX(Dataset.name = "gamma3", rseed = -4, n = n,
+      Theta = a.Theta(Theta))
```

4.1.2 Finite mixture estimation

```
R> gamma1est <- REBMIX(Dataset = a.Dataset(gamma1), Preprocessing = "kernel density estimation
+      cmax = 8, Criterion = "BIC", pdf = "gamma")
R> gamma2est <- REBMIX(Dataset = a.Dataset(gamma2), Preprocessing = "histogram",
+      cmax = 8, Criterion = "BIC", pdf = "gamma")
R> gamma3est <- REBMIX(Dataset = a.Dataset(gamma3), Preprocessing = "histogram",
+      cmax = 8, Criterion = "BIC", pdf = "gamma", K = 23:27)
```

4.1.3 Plot method

```
R> plot(gamma3est, pos = 1, what = c("pdf", "marginal cdf"), ncol = 2,
+      npts = 1000)
```

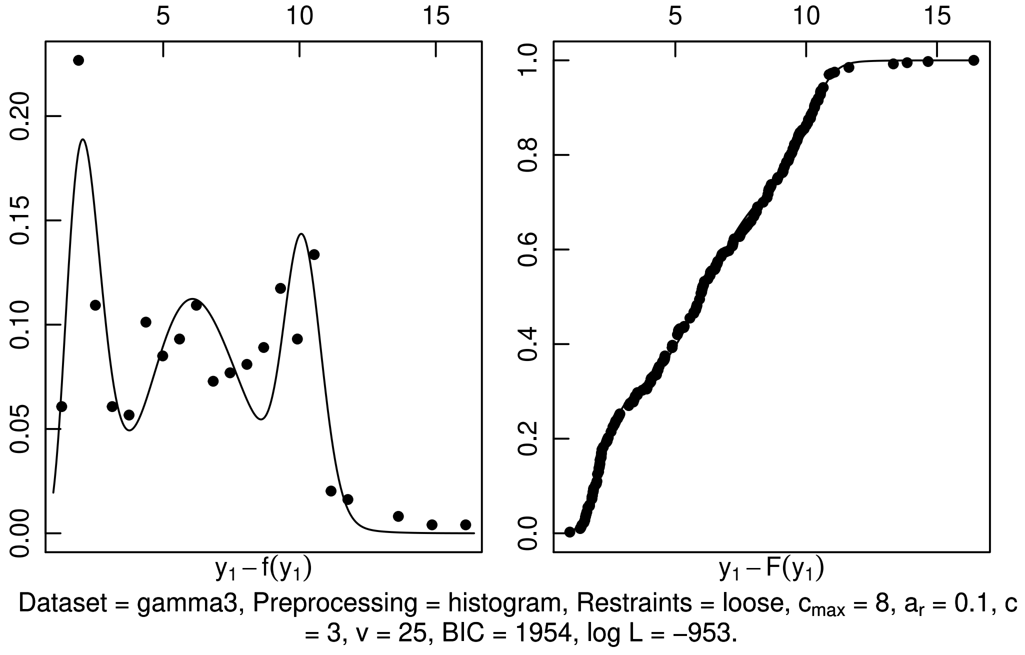


Figure 1: Gamma 3 dataset. Empirical density (circles) and predictive gamma mixture density in black solid line.

4.1.4 Summary, a.theta1.all and a.theta2.all methods

```
R> summary(gamma2est)

  Dataset Preprocessing Criterion c v/k    IC logL M
1  gamma2     histogram       BIC 2  37 -1349  690 5
Maximum logL = 690 at pos = 1.

R> a.theta1.all(gamma1est, pos = 1)

            [,1]
theta2.1 0.01027
theta2.2 0.00921
```

```
theta2.3 0.00870
theta2.4 0.01118
```

```
R> a.theta2.all(gamma1est, pos = 1)
```

```
         [,1]
theta2.1  195
theta2.2  437
theta2.3  918
theta2.4  535
```

4.1.5 Bootstrap methods

```
R> gamma3boot <- boot(x = gamma3est, pos = 1, Bootstrap = "p", B = 10)
R> gamma3boot
```

```
An object of class "REBMIX.boot"
Slot "c":
 [1] 3 3 3 4 3 3 3 3 3 3
Slot "c.se":
[1] 0.316
Slot "c.cv":
[1] 0.102
Slot "c.mode":
[1] 3
Slot "c.prob":
[1] 0.9
```

```
R> summary(gamma3boot)
```

```
w.cv
[1] 0.139 0.115 0.160
             [,1]
theta1.1.cv 0.497
theta1.2.cv 0.708
theta1.3.cv 0.479
            [,1]
theta2.1.cv 1.94
theta2.2.cv 0.67
theta2.3.cv 1.04
Mode probability = 0.9 at c = 3 components.
```

4.2 Poisson dataset

Dataset consists of $n = 600$ two dimensional observations obtained by generating data points separately from each of three Poisson distributions. The component dataset sizes and parameters, which are those studied in Ma et al. (2009), are displayed below

$$\begin{array}{ll} \boldsymbol{\theta}_1 = (3, 2)^\top & n_1 = 200 \\ \boldsymbol{\theta}_2 = (9, 10)^\top & n_2 = 200 \\ \boldsymbol{\theta}_3 = (15, 16)^\top & n_3 = 200 \end{array}$$

For the dataset Ma et al. (2009) conduct 100 experiments by selecting different initial values of the mixing proportions. In all the cases, the adaptive gradient BYY learning algorithm leads to the correct model selection, i.e., finally allocating the correct number of Poissons for the dataset. In the meantime, it also results in an estimate for each parameter in the original or true Poisson mixture which generated the dataset. As the dataset of Ma et al. (2009) can not exactly be reproduced, 10 datasets are generated with random seeds r_{seed} ranging from -1 to -10.

4.2.1 Finite mixture generation

```
R> n <- c(200, 200, 200)
R> Theta <- new("RNGMIX.Theta", c = 3, pdf = rep("Poisson", 2))
R> a.theta1(Theta, 1) <- c(3, 2)
R> a.theta1(Theta, 2) <- c(9, 10)
R> a.theta1(Theta, 3) <- c(15, 16)
R> poisson <- RNGMIX(Dataset.name = paste("Poisson_", 1:10, sep = ""),
+      n = n, Theta = a.Theta(Theta))
```

4.2.2 Finite mixture estimation

```
R> poissonest <- REBMIX(Dataset = a.Dataset(poisson), Preprocessing = "histogram",
+      cmax = 10, Criterion = "MDL5", pdf = rep("Poisson", 2), K = 1)
```

4.2.3 Plot method

4.2.4 Clustering

4.2.5 Summary, a.theta1.all and a.theta2.all methods

```
R> summary(poissonest)
```

```
      Dataset Preprocessing Criterion c v/k   IC  logL  M
1   Poisson_1     histogram      MDL5 3   1 7040 -3392  8
2   Poisson_2     histogram      MDL5 2   1 7267 -3554  5
3   Poisson_3     histogram      MDL5 2   1 7173 -3506  5
4   Poisson_4     histogram      MDL5 3   1 7004 -3374  8
5   Poisson_5     histogram      MDL5 3   1 7074 -3409  8
6   Poisson_6     histogram      MDL5 3   1 7140 -3442  8
7   Poisson_7     histogram      MDL5 2   1 7303 -3572  5
8   Poisson_8     histogram      MDL5 4   1 7145 -3397 11
9   Poisson_9     histogram      MDL5 2   1 7198 -3519  5
10 Poisson_10     histogram      MDL5 3   1 7004 -3374  8
Maximum logL = -3374 at pos = 10.
```

```
R> a.theta1.all(poissonest, pos = 1)
```

```
          [,1]  [,2]
theta2.1  3.55  2.59
theta2.2 14.62 15.96
theta2.3  9.28 10.19
```

```
R> a.theta2.all(poissonest, pos = 1)
```

```
         [,1] [,2]
theta2.1    0    0
theta2.2    0    0
theta2.3    0    0
```

4.3 Multivariate normal `wreath` dataset

A `wreath` dataset (Fraley et al., 2005) consist of 1000 observations drawn from a 14-component normal mixture in which the covariances of the components have the same size and shape but differ in orientation.

```
R> data("wreath", package = "mclust")
```

4.3.1 Finite mixture estimation

```
R> wreathest <- REBMIX(model = "REBMVNORM", Dataset = list(as.data.frame(wreath)),
+      Preprocessing = "histogram", cmax = 20, Criterion = "BIC")
```

4.3.2 Plot method

4.3.3 Clustering

4.3.4 Summary method

```
R> summary(wreathest)
```

```
   Dataset Preprocessing Criterion  c v/k    IC  logL  M
1 dataset1     histogram       BIC 14  21 11174 -5300 83
Maximum logL = -5300 at pos = 1.
```

4.3.5 Summary method

```
R> summary(wreathclu)
```

```
Number of clusters  1         2         3         4         5
From cluster        2         3         4         11        14
To cluster          1         2         3         1         1
Entropy             1.49e-14  9.88e-04  2.23e-03  4.09e-03  6.43e-03
Entropy decrease    0.000988  0.001239  0.001868  0.002335  0.005271

Number of clusters  6         7         8         9         10
From cluster        10        5         9         13        12
To cluster          2         1         5         4         10
Entropy             1.17e-02  2.13e-02  3.24e-02  4.96e-02  7.39e-02
Entropy decrease    0.009594  0.011100  0.017251  0.024270  0.027832

Number of clusters  11        12        13
From cluster        6         7         8
To cluster          4         4         2
Entropy             1.02e-01  1.31e-01  2.82e-01
Entropy decrease    0.029363  0.150608  0.607494
```

4.4 Multivariate normal `ex4.1` dataset

A `ex4.1` dataset (Baudry et al., 2010; Fraley et al., 2016) consist of 600 two dimensional observations.

```
R> data("Baudry_etal_2010_JCGS_examples", package = "mclust")
```

4.4.1 Finite mixture estimation using exhaustive REBMIX&EM strategy

```
R> EM <- new("EM.Control", strategy = "exhaustive", variant = "EM",
+      acceleration = "fixed", acceleration.multiplier = 1, tolerance = 1e-04,
+      maximum.iterations = 1000)
R> ex4.1est.dens <- REBMIX(model = "REBMVNORM", Dataset = list(as.data.frame(ex4.1)),
+      Preprocessing = "histogram", cmax = 10, Criterion = "BIC",
+      EMcontrol = EM)
```

4.4.2 Plot method

4.4.3 Clustering

4.4.4 Summary method

```
R> summary(ex4.1est.dens)
```

```
   Dataset Preprocessing Criterion c v/k   IC  logL  M
1 dataset1     histogram       BIC 6  11 4126 -1951 35
Maximum logL = -1951 at pos = 1.
```

4.4.5 Summary method

```
R> summary(ex4.1clu.dens)
```

```
Number of clusters  1          2          3          4          5
From cluster        3          4          2          6          5
To cluster          1          3          1          4          1
Entropy             1.15e-14   1.04e-06   1.05e+00   4.77e+00   4.09e+01
Entropy decrease    1.04e-06   1.05e+00   3.72e+00   3.62e+01   8.11e+01
```

4.4.6 Clustering with exhaustive REBMIX&ECM strategy and ICL criterion

```
R> CEM <- new("EM.Control", strategy = "exhaustive", variant = "ECM",
+      acceleration = "fixed", acceleration.multiplier = 1, tolerance = 1e-04,
+      maximum.iterations = 1000)
R> ex4.1est <- REBMIX(model = "REBMVNORM", Dataset = list(as.data.frame(ex4.1)),
+      Preprocessing = "histogram", cmax = 10, Criterion = "ICL",
+      EMcontrol = CEM)
```

```
R> summary(ex4.1est)
```

```
   Dataset Preprocessing Criterion c v/k   IC  logL  M
1 dataset1     histogram       ICL 4  11 4234 -2041 23
Maximum logL = -2041 at pos = 1.
```

```
R> summary(ex4.1clu)
```

```
Number of clusters  1          2          3
From cluster        3          4          2
To cluster          1          3          1
Entropy             8.10e-15   3.18e-10   2.58e-01
Entropy decrease    3.18e-10   2.58e-01   2.99e+00
```

4.4.7 Acceleration of EM algorithm

Standard EM algorithm with fixed acceleration.multiplier of $a_{\mathrm{EM}} = 1.0$:

```
R> EM.normal <- new("EM.Control", strategy = "exhaustive", variant = "EM",
+      acceleration = "fixed", acceleration.multiplier = 1, tolerance = 1e-04,
+      maximum.iterations = 1000)
R> ex4.1est.em.normal <- REBMIX(model = "REBMVNORM", Dataset = list(as.data.frame(ex4.1)),
+      Preprocessing = "histogram", cmax = 15, Criterion = "BIC",
+      EMcontrol = EM.normal)
R> cat("Total number of EM algorithm iterations: ", a.summary.EM(ex4.1est.em.normal,
+      pos = 1, col.name = "total.iterations.nbr"), ". Value of BIC: ",
+      a.summary(ex4.1est.em.normal, pos = 1, col.name = "IC"))
```

```
Total number of EM algorithm iterations:  1970 . Value of BIC:  4126
```

Standard EM algorithm with fixed acceleration.multiplier of $a_{\mathtt{EM}} = 1.5$:

```
R> EM.fixed1.5 <- new("EM.Control", strategy = "exhaustive", variant = "EM",
+      acceleration = "fixed", acceleration.multiplier = 1.5, tolerance = 1e-04,
+      maximum.iterations = 1000)
R> ex4.1est.em.fixed1.5 <- REBMIX(model = "REBMVNORM", Dataset = list(as.data.frame(ex4.1)),
+      Preprocessing = "histogram", cmax = 15, Criterion = "BIC",
+      EMcontrol = EM.fixed1.5)
R> cat("Total number of EM algorithm iterations: ", a.summary.EM(ex4.1est.em.fixed1.5,
+      pos = 1, col.name = "total.iterations.nbr"), ". Value of BIC: ",
+      a.summary(ex4.1est.em.fixed1.5, pos = 1, col.name = "IC"))
```

```
Total number of EM algorithm iterations:  1380 . Value of BIC:  4126
```

Standard EM algorithm with line search for optimal increment $a_{\mathtt{EM}}$ in each iteration:

```
R> EM.line <- new("EM.Control", strategy = "exhaustive", variant = "EM",
+      acceleration = "line", acceleration.multiplier = 1, tolerance = 1e-04,
+      maximum.iterations = 1000)
R> ex4.1est.em.line <- REBMIX(model = "REBMVNORM", Dataset = list(as.data.frame(ex4.1)),
+      Preprocessing = "histogram", cmax = 15, Criterion = "BIC",
+      EMcontrol = EM.line)
R> cat("Total number of EM algorithm iterations: ", a.summary.EM(ex4.1est.em.line,
+      pos = 1, col.name = "total.iterations.nbr"), ". Value of BIC: ",
+      a.summary(ex4.1est.em.line, pos = 1, col.name = "IC"))
```

```
Total number of EM algorithm iterations:  1477 . Value of BIC:  4126
```

Standard EM algorithm with golden search for optimal increment $a_{\mathtt{EM}}$ in each iteration:

```
R> EM.golden <- new("EM.Control", strategy = "exhaustive", variant = "EM",
+      acceleration = "golden", acceleration.multiplier = 1, tolerance = 1e-04,
+      maximum.iterations = 1000)
R> ex4.1est.em.golden <- REBMIX(model = "REBMVNORM", Dataset = list(as.data.frame(ex4.1)),
+      Preprocessing = "histogram", cmax = 15, Criterion = "BIC",
+      EMcontrol = EM.golden)
R> cat("Total number of EM algorithm iterations: ", a.summary.EM(ex4.1est.em.golden,
+      pos = 1, col.name = "total.iterations.nbr"), ". Value of BIC: ",
+      a.summary(ex4.1est.em.golden, pos = 1, col.name = "IC"))
```

```
Total number of EM algorithm iterations:  2098 . Value of BIC:  4127
```

4.5 Multivariate iris dataset

The well known set of iris data as collected originally by Anderson (1936) and first analysed by Fisher (1936) is considered here. It is available at Asuncion and Newman (2007) consisting of the measurements of the length and width of both sepals and petals of 50 plants for each of the three types of iris species setosa, versicolor and virginica. The iris dataset is loaded, split into three subsets for the three classes and the Class column is removed.

```
R> data("iris")
R> levels(iris[["Class"]])
```

```
[1] "iris-setosa"     "iris-versicolor" "iris-virginica"
```

```
R> set.seed(5)
R> Iris <- split(p = 0.75, Dataset = iris, class = 5)
```

4.5.1 Finite mixture estimation

```
R> irisest <- REBMIX(model = "REBMVNORM", Dataset = a.train(Iris),
+      Preprocessing = "kernel density estimation", cmax = 10, Criterion = "ICL-BIC")
```

4.5.2 Classification

```
R> iriscla <- RCLSMIX(model = "RCLSMVNORM", x = list(irisest), Dataset = a.test(Iris),
+      Zt = a.Zt(Iris))
```

4.5.3 Show and summary methods

```
R> iriscla
```

```
An object of class "RCLSMVNORM"
Slot "CM":

                  iris-setosa iris-versicolor iris-virginica
  iris-setosa              13               0              0
  iris-versicolor           0              13              0
  iris-virginica            0               0             13
Slot "Error":
[1] 0
Slot "Precision":
[1] 1 1 1
Slot "Sensitivity":
[1] 1 1 1
Slot "Specificity":
[1] 1 1 1
Slot "Chunks":
[1] 1
```

```
R> summary(iriscla)
```

```
             Test      Predictive Frequency
1     iris-setosa     iris-setosa        13
2 iris-versicolor     iris-setosa         0
3  iris-virginica     iris-setosa         0
4     iris-setosa iris-versicolor         0
5 iris-versicolor iris-versicolor        13
6  iris-virginica iris-versicolor         0
7     iris-setosa  iris-virginica         0
8 iris-versicolor  iris-virginica         0
9  iris-virginica  iris-virginica        13
Error = 0.
```

4.5.4 Plot method

4.6 Multivariate adult dataset

The adult dataset containing 48842 instances with 16 continuous, binary and discrete variables was extracted from the census bureau database Asuncion and Newman (2007). Extraction was done by Barry Becker from the 1994 census bureau database. The adult dataset is loaded, complete cases are extracted and levels are replaced with numbers.

```
R> data("adult")
R> adult <- adult[complete.cases(adult), ]
R> adult <- as.data.frame(data.matrix(adult))
```

Numbers of unique values for variables are determined and displayed.

```
R> cmax <- unlist(lapply(apply(adult[, c(-1, -16)], 2, unique),
+      length))
R> cmax
```

```
           Age      Workclass         Fnlwgt      Education  Education.Num
            74              7          26741             16             16
Marital.Status     Occupation   Relationship           Race            Sex
             7             14              6              5              2
  Capital.Gain   Capital.Loss Hours.Per.Week Native.Country
           121             97             96             41
```

The dataset is split into train and test subsets for the two incomes and the `Type` and `Income` columns are removed.

```
R> Adult <- split(p = list(type = 1, train = 2, test = 1), Dataset = adult,
+      class = 16)
```

4.6.1 Finite mixture estimation

Number of components, component weights and component parameters are estimated assuming that the variables are independent for the set of chunks $y_{1j}, y_{2j}, \ldots, y_{14j}$.

```
R> adultest <- list()
R> for (i in 1:14) {
+      adultest[[i]] <- REBMIX(Dataset = a.train(chunk(Adult, i)),
+          Preprocessing = "histogram", cmax = min(120, cmax[i]),
+          Criterion = "BIC", pdf = "Dirac", K = 1)
+  }
```

4.6.2 Classification

The class membership prediction is based upon the best first search algorithm.

```
R> adultcla <- BFSMIX(x = adultest, Dataset = a.test(Adult), Zt = a.Zt(Adult))
```

4.6.3 Show and summary methods

```
R> adultcla
```

```
An object of class "RCLSMIX"
Slot "CM":

        1     2
  1 10649   711
  2  1397  2303
Slot "Error":
[1] 0.14
Slot "Precision":
[1] 0.937 0.622
Slot "Sensitivity":
[1] 0.884 0.764
```

```
Slot "Specificity":
[1] 1.228 0.943
Slot "Chunks":
[1] 11 12  4  8  1
```

```
R> summary(adultcla)
```

```
  Test Predictive Frequency
1    1          1     10649
2    2          1      1397
3    1          2       711
4    2          2      2303
Error = 0.14.
```

4.6.4 Plot method

5 Summary

The users of the rebmix package are kindly encouraged to inform the author about bugs and wishes.

References

E. Anderson. The species problem in iris. *Annals of the Missouri Botanical Garden*, 23(3):457–509, 1936. doi: 10.2307/2394164.

A. Asuncion and D. J. Newman. Uci machine learning repository, 2007. URL http://archive.ics.uci.edu/ml.

J. P. Baudry, A. E. Raftery, G. Celeux, K. Lo, and R. Gottardo. Combining mixture components for clustering. *Journal of Computational and Graphical Statistics*, 19(2):332–353, 2010. doi: 10.1198/jcgs.2010.08111.

G. Celeux and G. Govaert. A classification em algorithm for clustering and two stochastic versions. *Computational Statistics & Data Analysis*, 14(3):315–332, 1992. doi: 10.1016/0167-9473(92)90042-E.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society B*, 39(1):1–38, 1977. URL https://www.jstor.org/stable/2984875.

R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7(3):179–188, 1936. doi: 10.1111/j.1469-1809.1936.tb02137.x.

C. Fraley, A. Raftery, and R. Wehrens. Incremental model-based clustering for large datasets with small clusters. *Journal of Computational and Graphical Statistics*, 14(3):529–546, 2005. doi: 10.1198/106186005X59603.

C. Fraley, A. E. Raftery, L. Scrucca, T. B. Murphy, and M. Fop. *mclust: Normal Mixture Modelling for Model-Based Clustering, Classification, and Density Estimation*, 2016. URL http://CRAN.R-project.org/package=mclust. R package version 5.1.

K. H. Knuth. Optimal data-based binning for histograms and histogram-based probability density models. *Digital Signal Processing*, 95:102581, 2019. doi: 10.1016/j.dsp.2019.102581.

J. Ma, J. Liu, and Z. Ren. Parameter estimation of poisson mixture with automated model selection through byy harmony learning. *Pattern Recognition*, 42(11):2659–2670, 2009. doi: 10.1016/j.patcog.2009.03.029.

M. Nagode. Finite mixture modeling via rebmix. *Journal of Algorithms and Optimization*, 3(2):14–28, 2015. doi: 10.5963/JAO0302001.

M. Nagode. Multivariate normal mixture modeling, clustering and classification with the rebmix package. *ArXiv e-prints*, Jan. 2018.

M. Nagode and M. Fajdiga. The rebmix algorithm for the univariate finite mixture estimation. *Communications in Statistics - Theory and Methods*, 40(5):876–892, 2011a. doi: 10.1080/03610920903480890.

M. Nagode and M. Fajdiga. The rebmix algorithm for the multivariate finite mixture estimation. *Communications in Statistics - Theory and Methods*, 40(11):2022–2034, 2011b. doi: 10.1080/03610921003725788.

P. F. Velleman. Interactive computing for exploratory data analysis i: Display algorithms. In *Proceedings of the Statistical Computing Section*, Washington, D.C., 1976. American Statistical Association.

M. Wiper, D. R. Insua, and F. Ruggeri. Mixtures of gamma distributions with applications. *Journal of Computational and Graphical Statistics*, 10(3):440–454, 2001. URL `http://www.jstor.org/stable/1391098`.

Marko Nagode
University of Ljubljana
Faculty of Mechanical Engineering
Aškerčeva 6
1000 Ljubljana
Slovenia
Marko.Nagode@fs.uni-lj.si.

```
R> plot(poissonest, pos = 1, what = c("pdf", "marginal pdf", "IC",
+      "D", "logL"), nrow = 2, ncol = 3, npts = 1000)
```

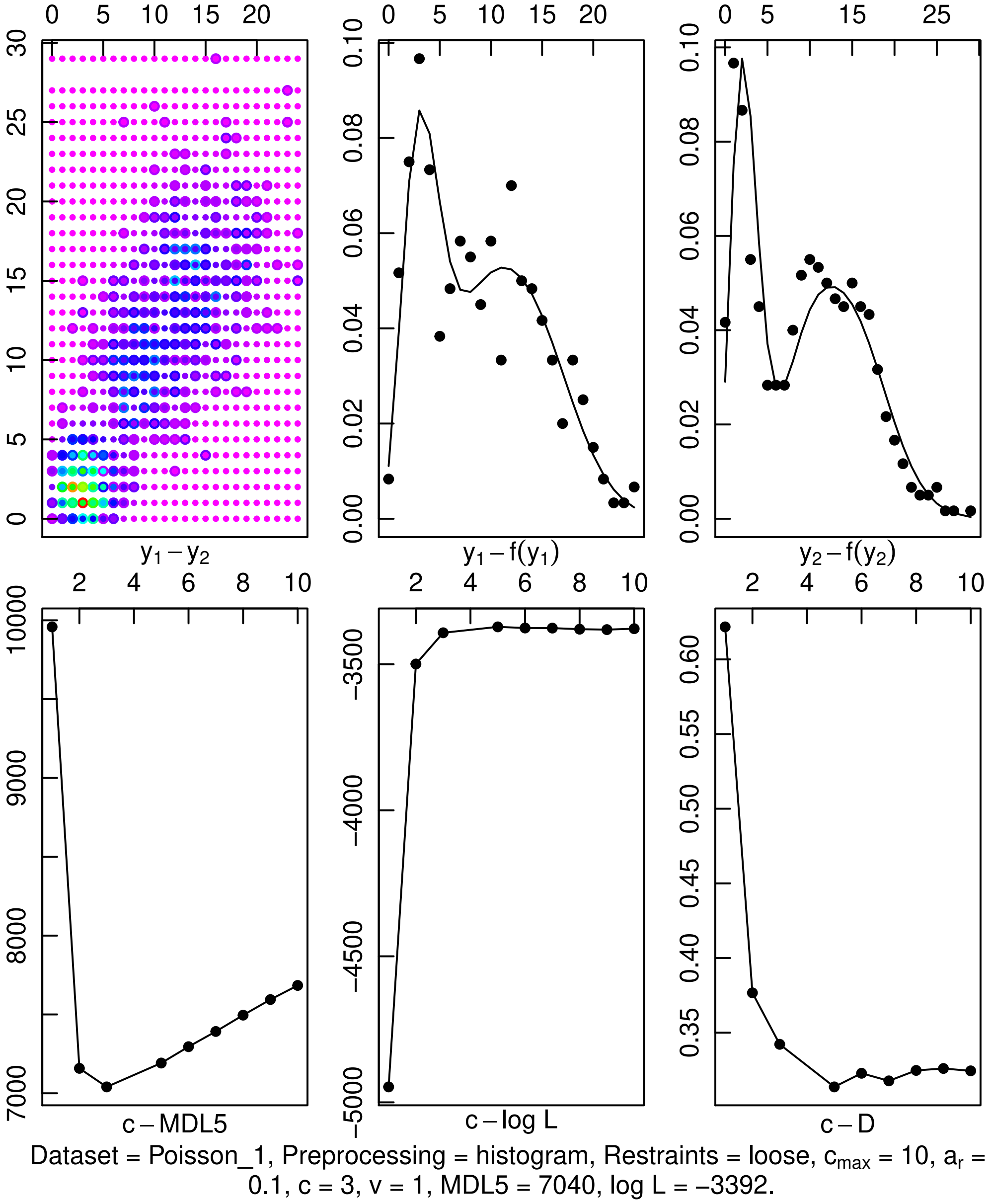


Figure 2: Poisson dataset. Empirical densities (coloured large circles), predictive multivariate Poisson-Poisson mixture density (coloured small circles), empirical densities (circles), predictive univariate marginal Poisson mixture densities and progress charts (solid line).

```
R> poissonclu <- RCLRMIX(x = poissonest, pos = 1, Zt = a.Zt(poisson))
R> plot(poissonclu)
```

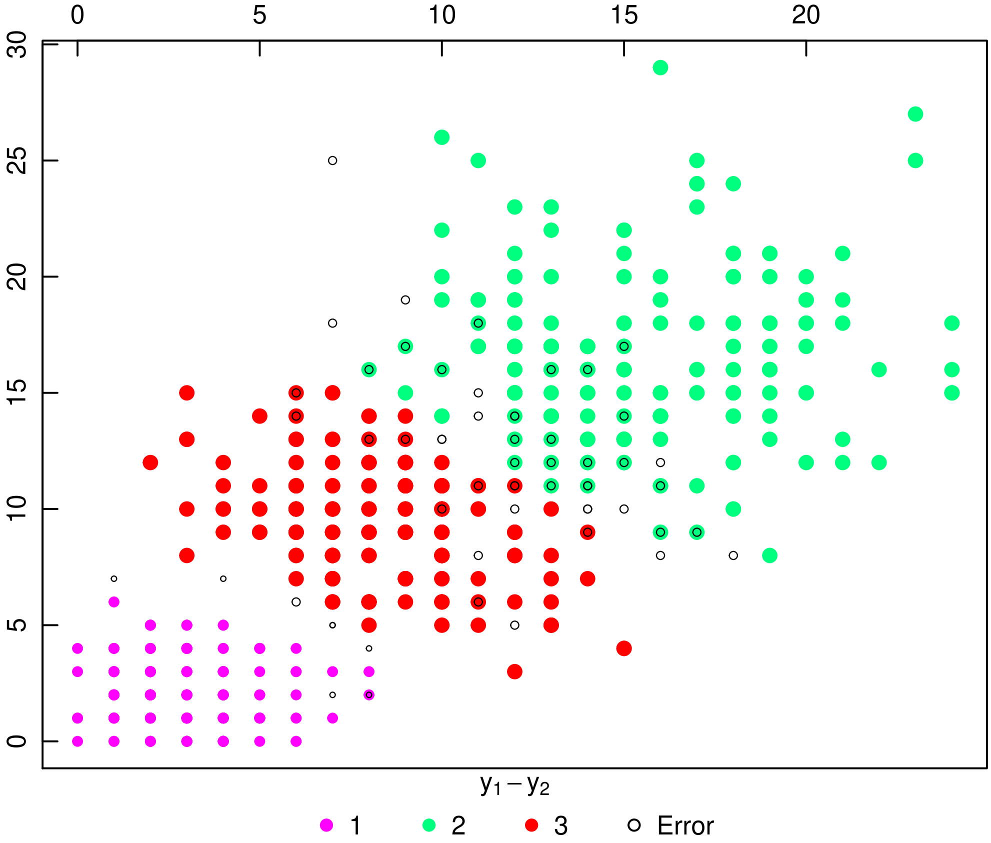


Figure 3: Poisson dataset. Predictive cluster membership (coloured circles), error (black circles).

```
R> plot(wreathest)
```

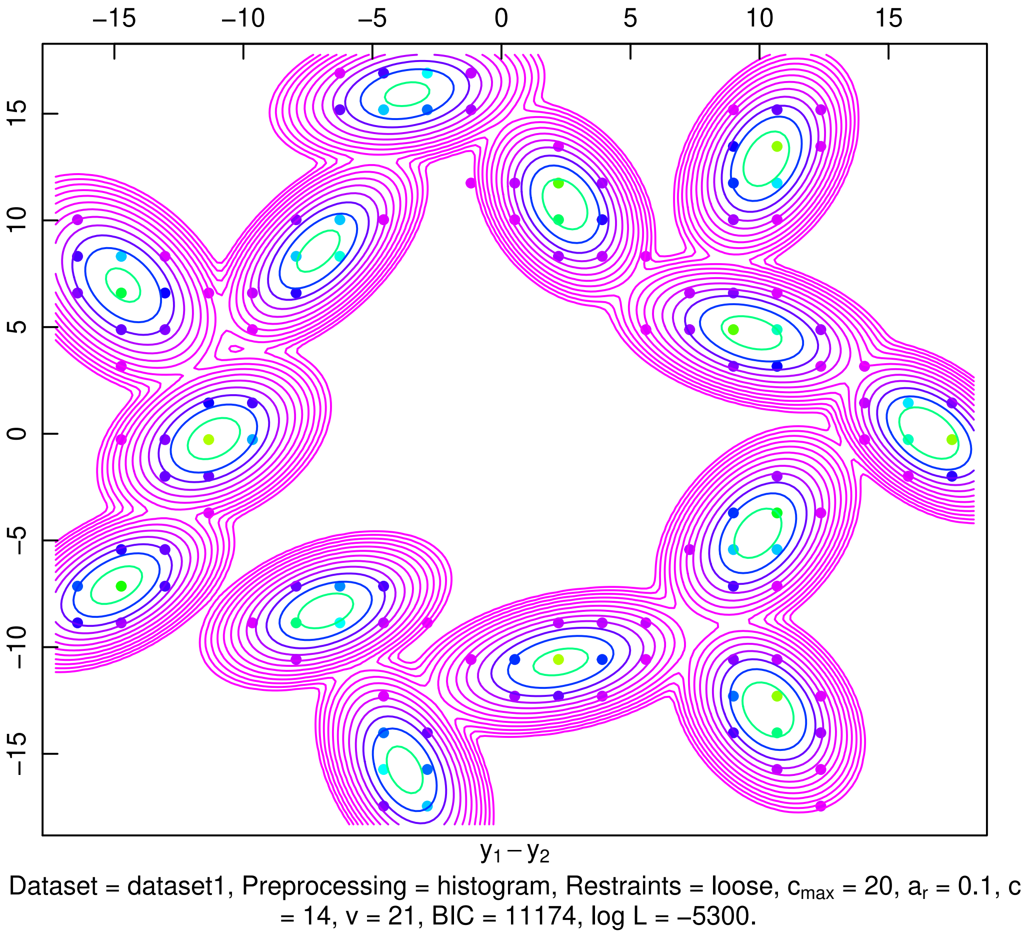


Figure 4: Dataset `wreath`. Empirical densities (coloured circles), predictive multivariate normal mixture density (coloured lines).

```
R> wreathclu <- RCLRMIX(model = "RCLRMVNORM", x = wreathest)
R> plot(wreathclu, s = 14)
```

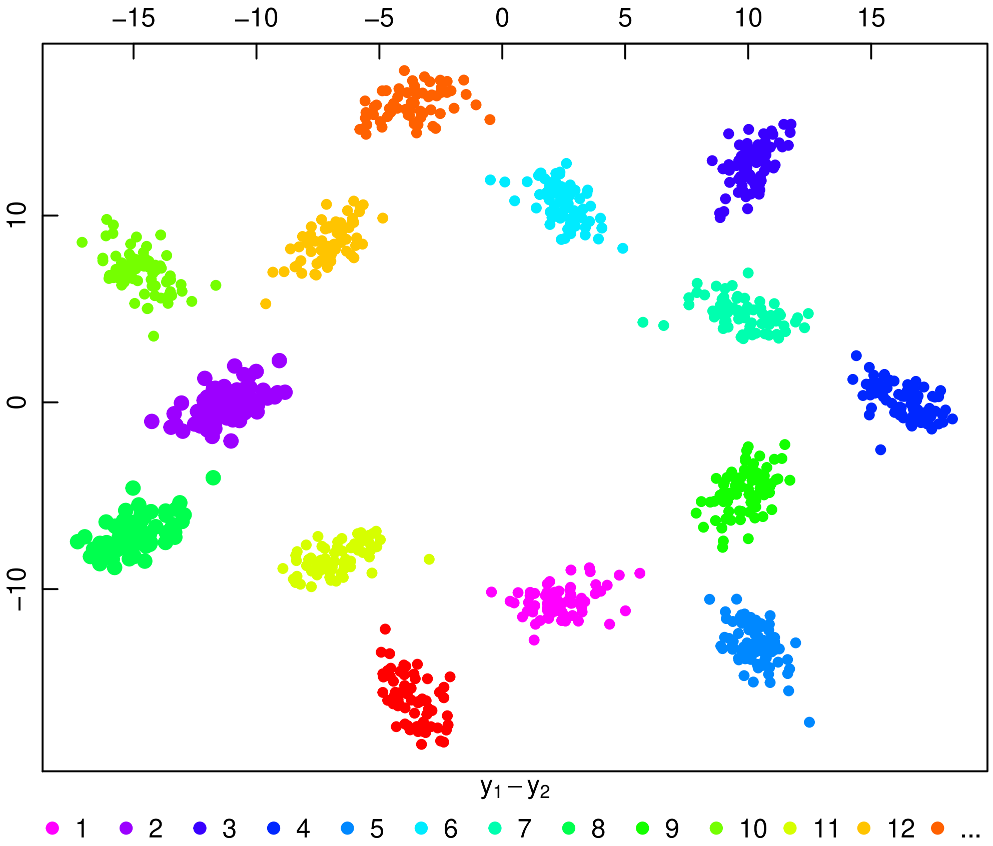
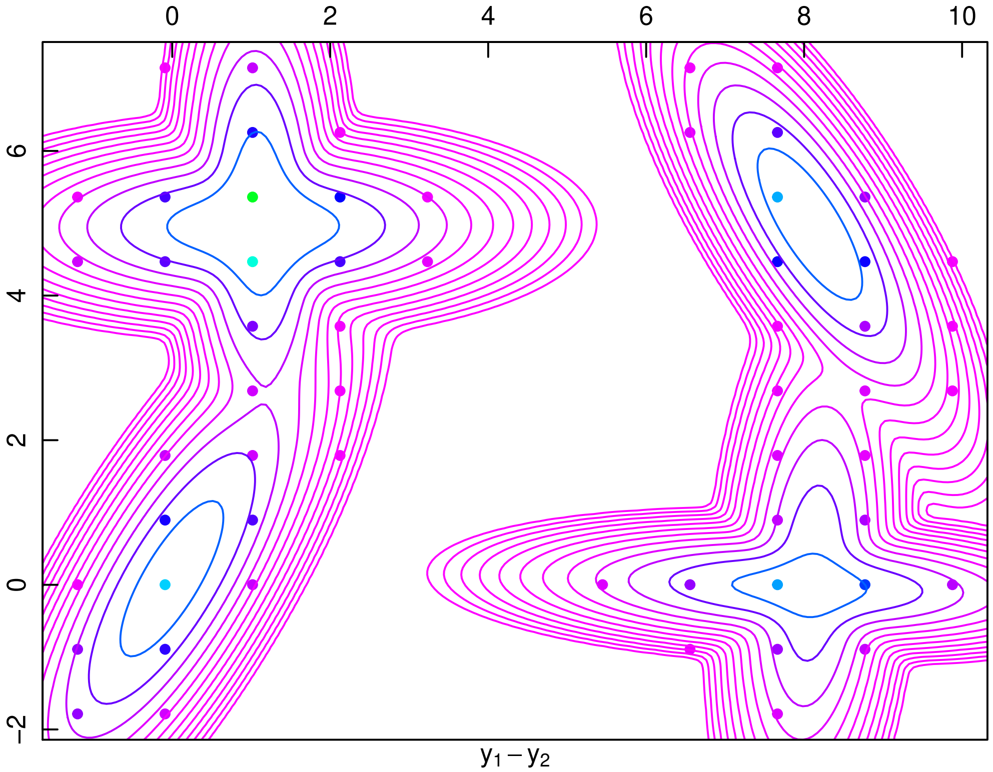


Figure 5: Dataset `wreath`. Predictive cluster membership (coloured circles).

```
R> plot(ex4.1est.dens, pos = 1, what = c("pdf"), nrow = 1, ncol = 1)
```



Dataset = dataset1, Preprocessing = histogram, Restraints = loose, c_{max} = 10, a_r = 0.1, c = 6, v = 11, BIC = 4126, log L = −1951.

Figure 6: Dataset `ex4.1`. Empirical densities (coloured circles), predictive multivariate normal mixture density (coloured lines).

```
R> ex4.1clu.dens <- RCLRMIX(model = "RCLRMVNORM", x = ex4.1est.dens)
R> plot(ex4.1clu.dens)
```

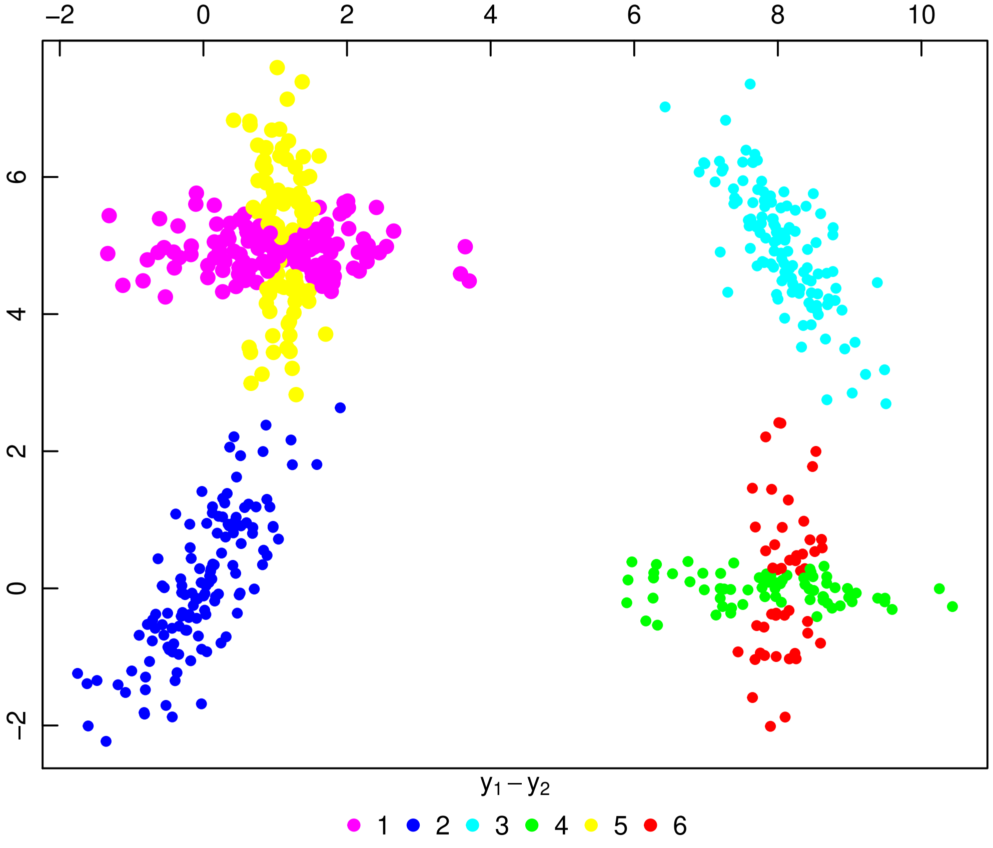


Figure 7: Dataset `ex4.1`. Predictive cluster membership (coloured circles).

```
R> plot(ex4.1est, pos = 1, what = c("pdf"), nrow = 1, ncol = 1)
```

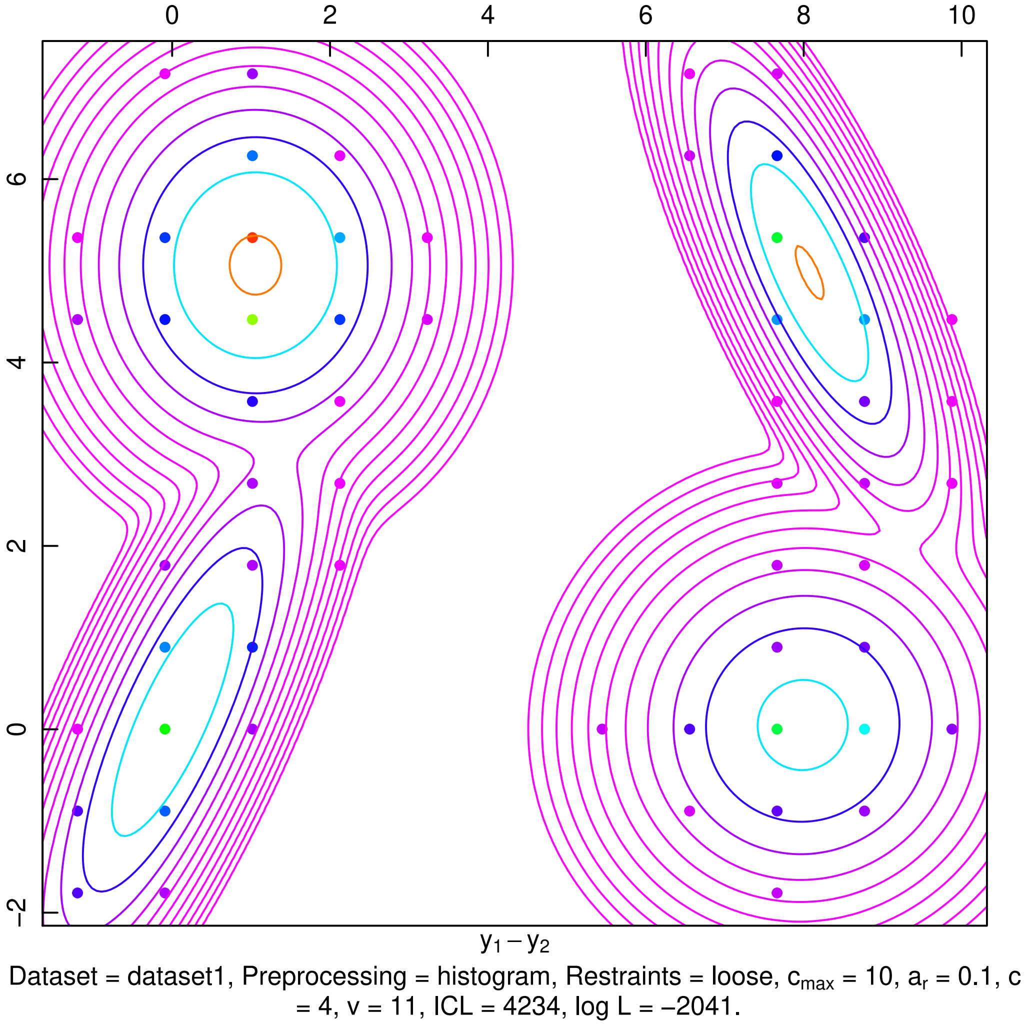


Figure 8: Dataset `ex4.1`. Empirical densities (coloured circles), predictive multivariate normal mixture density (coloured lines).

```
R> ex4.1clu <- RCLRMIX(model = "RCLRMVNORM", x = ex4.1est)
R> plot(ex4.1clu)
```

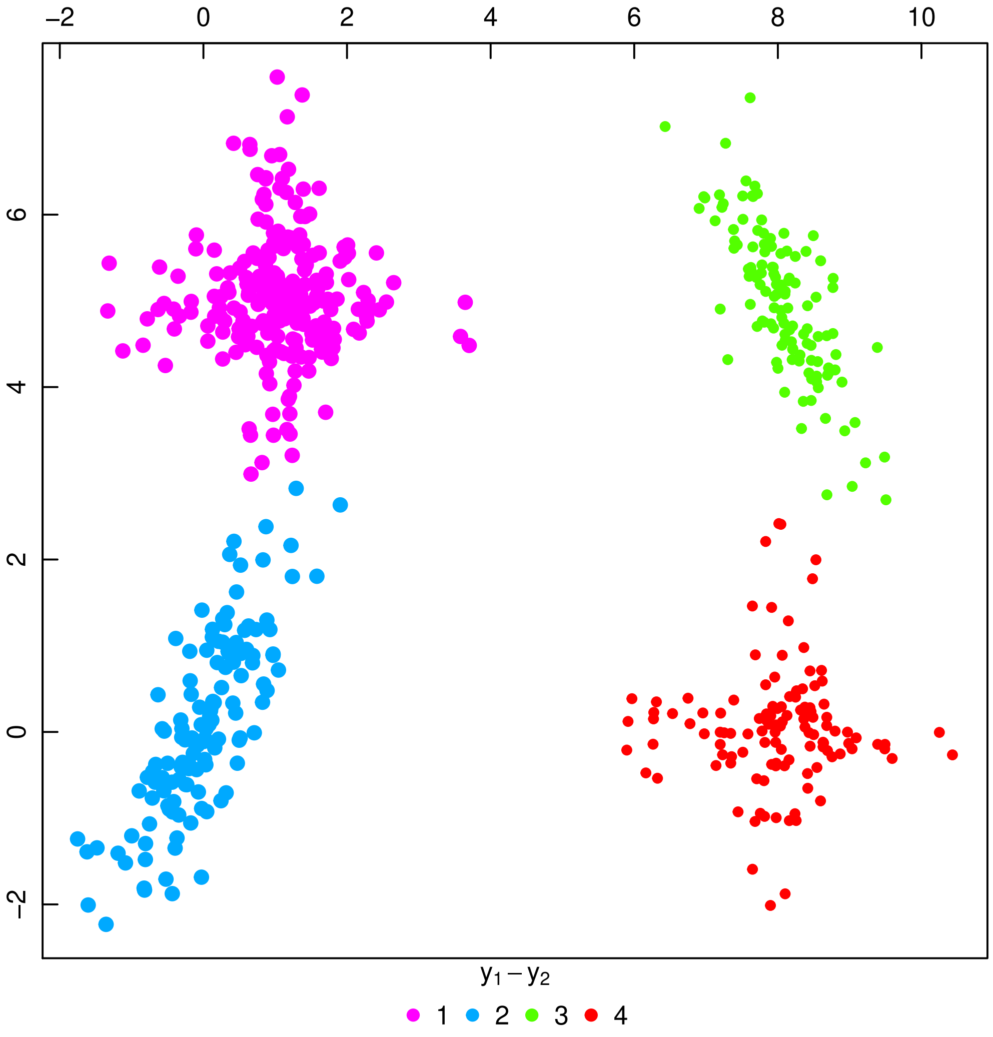


Figure 9: Dataset `ex4.1`. Predictive cluster membership (coloured circles).

```
R> plot(iriscla, nrow = 3, ncol = 2)
```

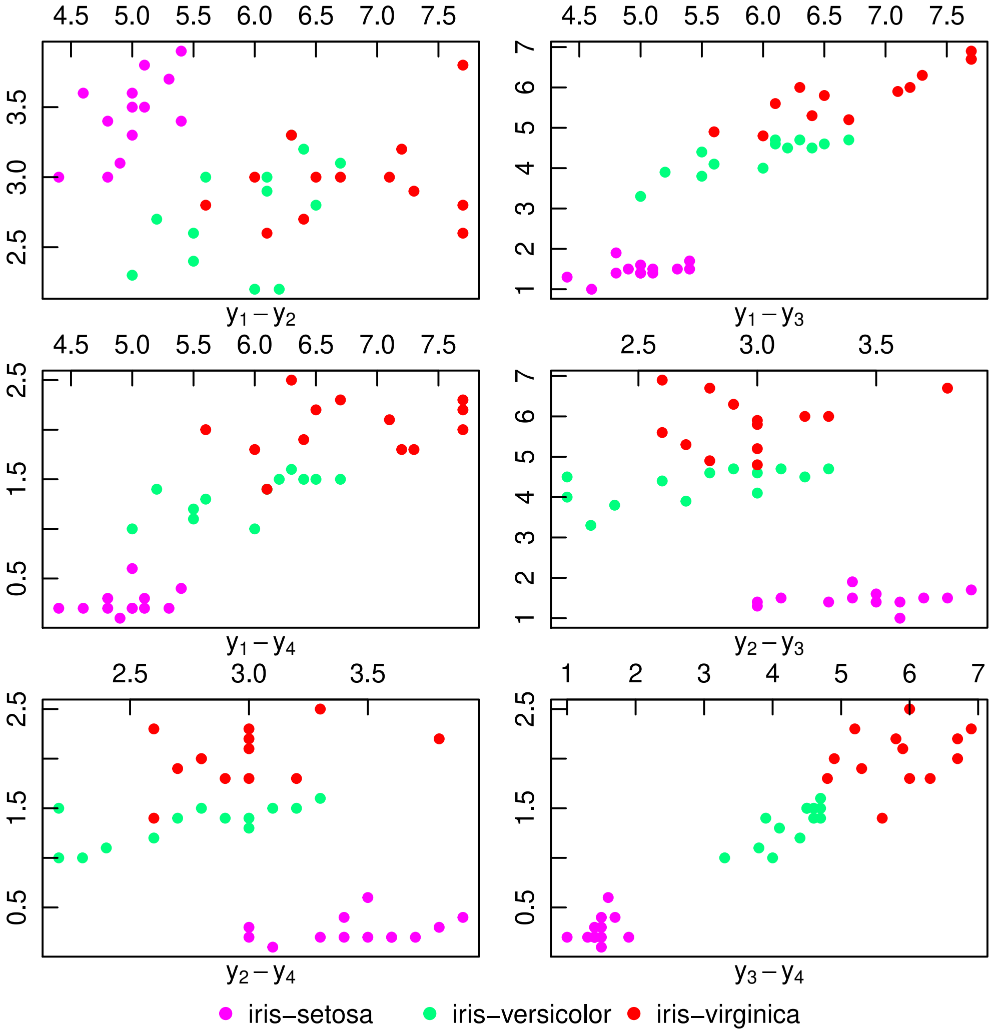


Figure 10: Dataset `iris`. Predictive class membership (coloured circles), error (black circles).

```
R> plot(adultcla, nrow = 5, ncol = 2)
```

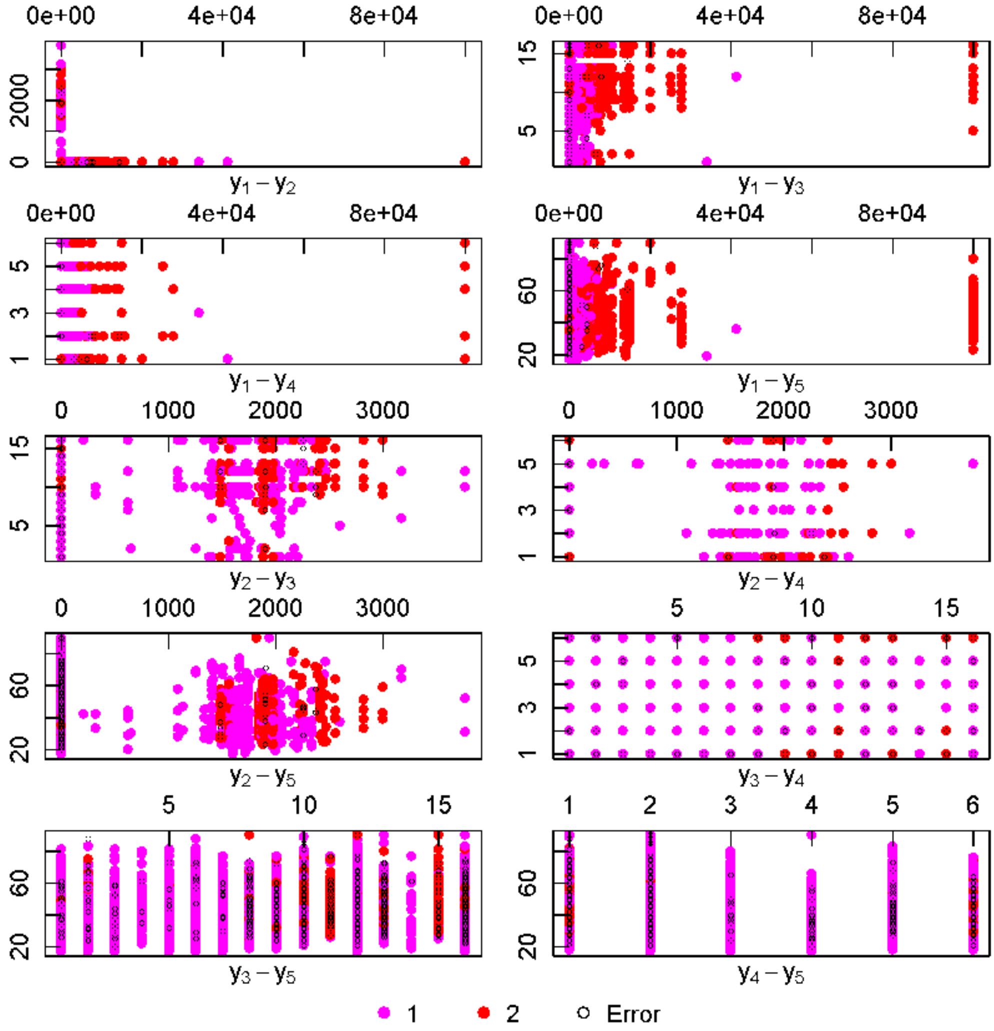


Figure 11: Dataset `adult`. Predictive class membership (coloured circles), error (black circles).