Model-Based Cluster and Discriminant Analysis with the MIXMOD Software

Christophe Biernacki [a,*] Gilles Celeux [b] Gérard Govaert [c]

Florent Langrognet [a]

[a] *UMR 6623, CNRS & Université de Franche-Comté*

[b] *INRIA*

[c] *UMR 6599, CNRS & Université de Technologie de Compiègne*

Abstract

The MIXMOD (MIXture MODelling) software fits mixture models to a given data set with a density estimation, a clustering or a discriminant analysis purpose. A large variety of algorithms to estimate the mixture parameters are proposed (EM, Classification EM, Stochastic EM) and it is possible to combine them to lead to different strategies in order to get a sensible maximum of the likelihood (or complete-data likelihood) function. MIXMOD is currently focused on multivariate Gaussian mixtures and fourteen different Gaussian models can be considered according to different assumptions on the component variance matrix eigenvalue decomposition. Moreover, different information criteria for choosing a parsimonious model (the number of mixture components, for instance), some of them favoring either a cluster analysis or a discriminant analysis view point, are included. Written in C++, MIXMOD is interfaced with SCILAB and MATLAB. The software, the statistical documentation and also the user guide are available on the internet at the following address:

`http://www-math.univ-fcomte.fr/mixmod/index.php`

Preprint submitted to Elsevier Science *7 January 2005*

Key words: Gaussian models, EM-like algorithms, model selection.

1 Introduction

Because of their high flexibility, finite mixture distributions are become a very popular approach to model a wide variety of random phenomena. In particular, it is recognized as being a powerful tool for density estimation, clustering and discriminant analysis. Consequently, fields which are potentially concerned by the mixture modelling approach are extremely varied, including astronomy, biology, genetics, economics, etc. Thus, softwares implementing the most recent evolutions in model-based cluster and discriminant analysis are welcome for various categories of people as researchers, engineers and teachers. MIXMOD is a software having for goal to meet these particular needs.

MIXMOD is publicly available under the GPL license and is distributed for different platforms (Linux, Unix, Windows). It is an object-oriented package built around C++ language but it is interfaced with the widely used mathematical softwares MATLAB and SCILAB. It was developed jointly by INRIA, the laboratory of mathematics of Besançon and the Heudiasyc laboratory of Compiègne.

In its present version, MIXMOD proposes multivariate Gaussian mixture models

* Corresponding author
Email addresses: `christophe.biernacki@math.univ-fcomte.fr` (Christophe Biernacki), `gilles.celeux@inria.fr` (Gilles Celeux), `gerard.govaert@utc.fr` (Gérard Govaert), `florent.langrognet@math.univ-fcomte.fr` (Florent Langrognet).

but generalization to other types of mixture distributions is planned for the next versions. The main features of the present version of the software are the following:

- three levels of use from the beginner to the expert;
- fourteen geometrically meaningful Gaussian mixture models from different variance matrices parameterizations;
- estimation of mixture parameters with EM and EM-like algorithms, provided with different initialization strategies and possibility of combining such algorithms;
- criteria to select a model which depends on the cluster or the discriminant analysis purpose;
- numerous displays including densities, iso-densities, discriminant rules, observations, labels, etc. in canonical or PCA (Principal Component Analysis) axes and for several dimensions (1D, 2D and 3D).

This paper does not intend to replace neither the user guide nor the statistical documentation that the reader can find on the web. The aim is rather to draw a synthetic overview of MIXMOD functionalities by mixing a short presentation of statistical tools with some selected examples.

2 Some technical features of MIXMOD

The development of the software started in 2001 and the last release of MIXMOD (MIXMOD 1.6) is composed of 40 C++ classes and 20000 C++ lines and is interfaced with SCILAB and MATLAB.

The web site (`http://www-math.univ-fcomte.fr/mixmod/index.php`) has been recently improved and allows to reach the following items: Download, Documentation, FAQ, Bugs, News, ...

2.1 Three different ways of using MIXMOD

The user can choose one of the following ways for using MIXMOD:

- MIXMOD as a GUI: the `mixmodGraph` function, available in SCILAB and MATLAB environment, allows to run MIXMOD Graphical User Interface. This function is the easiest way to discover MIXMOD, but some MIXMOD particular features are not available in this mode.
- MIXMOD as a SCILAB or a MATLAB function: in SCILAB and MATLAB environment, users can run the `mixmod` function. In this context, the user can use MIXMOD as well as any other SCILAB or MATLAB function. This function provides a lot of optional inputs and allows to fix in a more precise way some parameters than with the `mixmodGraph` function. Moreover, the `mixmodView` function can be called to visualize outputs of this function.
- MIXMOD as a command line: this last way to run MIXMOD is not available in SCILAB and MATLAB environments. This use of MIXMOD is reserved to experts (need for using files of inputs and outputs) who run MIXMOD in a shell environment (Linux, Unix or Windows).

The second way (`mixmod` function in SCILAB environment) will be used throughout this document to illustrate examples.

2.2 Data representation in MIXMOD

Three formats of data are used in MIXMOD:

- Individuals are represented in a standard way: a line for each individual and a column for each variable (see files with extension `.dat` in the directory DATA of MIXMOD package).
- Partition: each line corresponds to an individual and each column indicates the group membership (`0` if the individual does not belong to the group associated to this column and `1` otherwise). A line with only zeros (none `1`) indicates that the membership is unknown (see files with extension `.lab` in the directory DATA of MIXMOD package).
- Weight: each line corresponds to the weight of each individual (see files with extension `.wgt` in the directory DATA of MIXMOD package).

In the following, the Fisher's Iris data set (Fisher, 1936), will be used to illustrate MIXMOD functionalities. As previously described, individuals are in the file `iris.dat`, complete labels in `iris.lab` and weights in `iris.wgt`.

2.3 Performance of MIXMOD *throughout the versions*

One of the goals of MIXMOD is to provide a software which could run fast. In this way, a specific work was undertaken to achieve this goal. For example MIXMOD 1.6 is approximately 3 times faster than MIXMOD 1.1. This work will be continued in the next releases and versions.

3 Twenty-eight Gaussian models

3.1 Eigenvalue parameterization of variance matrices

In MIXMOD, it is assumed that a data $\boldsymbol{x}_i \in \mathbb{R}^d$ belongs to the kth cluster if it is a realization of the Gaussian density

$$\varphi(\boldsymbol{x}_i; \boldsymbol{\mu}_k, \Sigma_k) = (2\pi)^{-d/2} |\Sigma_k|^{-1/2} \exp\left\{ -\frac{1}{2} (\boldsymbol{x}_i - \boldsymbol{\mu}_k)' \Sigma_k^{-1} (\boldsymbol{x}_i - \boldsymbol{\mu}_k) \right\}. \quad (1)$$

Thus, this cluster is ellipsoidal, centered at the mean $\boldsymbol{\mu}_k$ and the variance matrice Σ_k determine its other geometric characteristics.

Following Banfield and Raftery (1993) and Celeux and Govaert (1995), each variance matrix is parameterized by its eigenvalue decomposition

$$\Sigma_k = \lambda_k D_k A_k D_k' \quad (2)$$

where $\lambda_k = |\Sigma_k|^{1/d}$, D_k is the matrix of eigenvectors of Σ_k and A_k is a diagonal matrix, such that $|A_k| = 1$, with the normalized eigenvalues of Σ_k on the diagonal in a decreasing order. The parameter λ_k determines the *volume* of the kth cluster, D_k its *orientation* and A_k its *shape*. By allowing some but not all of these quantities to vary between clusters, we obtain parsimonious and easily interpreted models which are appropriate to describe various clustering situations.

First, the volumes, the shapes and the orientations of clusters can be allowed to vary or to be equal between clusters. Variations on assumptions on the parameters λ_k, D_k and A_k lead to eight general models of interest. For instance, we can assume different volumes and keep the shapes and orientations equal by requiring that $A_k = A$ (A unknown) and $D_k = D$ (D unknown) for each cluster. We denote this model $[\lambda_k DAD']$. With this convention, writing $[\lambda D_k AD'_k]$ means that we consider clusters with equal volumes, equal shapes and different orientations. Another family of interest consists of assuming that the variance matrices Σ_k are diagonal. In the parameterization (2), it means that the orientation matrices D_k are permutation matrices. We write $\Sigma_k = \lambda_k B_k$ where B_k is a diagonal matrix with $|B_k| = 1$. This particular parameterization gives rise to four models: $[\lambda B]$, $[\lambda_k B]$, $[\lambda B_k]$ and $[\lambda_k B_k]$. The last family of models consists of assuming spherical shapes, namely $A_k = I$, I denoting the identity matrix. In such a case, two parsimonious models are in competition: $[\lambda I]$ and $[\lambda_k I]$. Finally, fourteen Gaussian models are obtained in this way.

Note that, in the following, models $[\lambda DAD']$ and $[\lambda D_k A_k D'_k]$ may be also expressed respectively in the more compact form $[\lambda C]$ and $[\lambda C_k]$. Models $[\lambda_k C]$ and $[\lambda_k C_k]$ are implicitly defined by analogy.

3.2 Constraints on proportions

Beyond these geometrical features, another important property of the kth cluster is the proportion of data that belongs *a priori* to it. Such a proportion is denoted by p_k. Thus, two typical models on proportions are considered: the case where clusters have the same proportion is denoted by $[p]$, and the case where they may have different proportions is denoted by $[p_k]$. Combining these two models on proportions with the fourteen previous models on geometrical features leads to twenty-eight different models noted $[p\lambda I]$, $[p_k\lambda I]$, $[p\lambda_k DAD']$, etc. These twenty-eight models are available in the M-step of the EM algorithm and its variants (SEM, CEM), and the MIXMOD identifier for each model is expressed in Table 1.

3.3 Links with some standard criteria

Those different clustering situations have not only a simple geometric interpretation, but they allow also to retrieve some standard criteria that were proposed without any reference to a statistical model. For instance, in clustering, the K-means criterion of Ward (1963) is obtained with the simplest model $[p\lambda I]$, the model $[p\lambda DAD']$ corresponds to the criterion suggested by Friedman and Rubin (1967) and models $[p\lambda_k DAD']$, $[p\lambda_k DA_k D']$ and $[p\lambda_k D_k A_k D_k']$ correspond to other classical clustering criteria (see for instance Scott and Symons, 1971; Diday and Govaert, 1974; Maronna and Jacovkis, 1974; Schroeder, 1976). In discriminant analysis, models $[p\lambda C]$ and $[p\lambda_k C_k]$ respectively correspond to classical linear and quadratic allocation rules (see for instance McLachlan, 1992).

4 Model-based clustering

4.1 Data and aim

Data managed in MIXMOD for clustering are typically composed by n vectors $\mathbf{x} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ in $\mathbb{R}^d$ and the aim is to estimate an unknown partition $\mathbf{z}$ of $\mathbf{x}$ into K clusters, where $\mathbf{z} = \{\mathbf{z}_1, \ldots, \mathbf{z}_n\}$ denotes the labels with $\mathbf{z}_i = (z_{i1}, \ldots, z_{iK})$, $z_{ik} = 1$ if $\boldsymbol{x}_i$ belongs to the kth cluster and 0 if not. Nevertheless, partial labelling of data is possible, so MIXMOD allows to consider situations where the set of individuals $\mathbf{x}$ is divided into two sets $\mathbf{x} = \{\mathbf{x}^\ell, \mathbf{x}^u\}$ where $\mathbf{x}^\ell = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m\}$ $(1 \leq m \leq n)$ are units with known labels $\mathbf{z}^\ell = \{\mathbf{z}_1, \ldots, \mathbf{z}_m\}$ whereas $\mathbf{x}^u = \{\boldsymbol{x}_{m+1}, \ldots, \boldsymbol{x}_n\}$ are units with unknown

MIXMOD identifier	Model	Family	Prop.	Volume	Shape	Orient.
`Gaussian_p_L_C`	$[p\lambda DAD']$	General	Equal	Equal	Equal	Equal
`Gaussian_p_Lk_C`	$[p\lambda_k DAD']$			Free	Equal	Equal
`Gaussian_p_L_D_Ak_D`	$[p\lambda DA_kD']$			Equal	Free	Equal
`Gaussian_p_Lk_D_Ak_D`	$[p\lambda_k DA_kD']$			Free	Free	Equal
`Gaussian_p_L_Dk_A_Dk`	$[p\lambda D_kAD'_k]$			Equal	Equal	Free
`Gaussian_p_Lk_Dk_A_Dk`	$[p\lambda_k D_kAD'_k]$			Free	Equal	Free
`Gaussian_p_L_Ck`	$[p\lambda D_kA_kD'_k]$			Equal	Free	Free
`Gaussian_p_Lk_Ck`	$[p\lambda_k D_kA_kD'_k]$			Free	Free	Free
`Gaussian_p_L_B`	$[p\lambda B]$	Diagonal	Equal	Equal	Equal	Axes
`Gaussian_p_Lk_B`	$[p\lambda_k B]$			Free	Equal	Axes
`Gaussian_p_L_Bk`	$[p\lambda B_k]$			Equal	Free	Axes
`Gaussian_p_Lk_Bk`	$[p\lambda_k B_k]$			Free	Free	Axes
`Gaussian_p_L_I`	$[p\lambda I]$	Spherical	Equal	Equal	Equal	NA
`Gaussian_p_Lk_I`	$[p\lambda_k I]$			Free	Equal	NA
`Gaussian_pk_L_C`	$[p_k\lambda DAD']$	General	Free	Equal	Equal	Equal
`Gaussian_pk_Lk_C`	$[p_k\lambda_k DAD']$			Free	Equal	Equal
`Gaussian_pk_L_D_Ak_D`	$[p_k\lambda DA_kD']$			Equal	Free	Equal
`Gaussian_pk_Lk_D_Ak_D`	$[p_k\lambda_k DA_kD']$			Free	Free	Equal
`Gaussian_pk_L_Dk_A_Dk`	$[p_k\lambda D_kAD'_k]$			Equal	Equal	Free
`Gaussian_pk_Lk_Dk_A_Dk`	$[p_k\lambda_k D_kAD'_k]$			Free	Equal	Free
`Gaussian_pk_L_Ck`	$[p_k\lambda D_kA_kD'_k]$			Equal	Free	Free
`Gaussian_pk_Lk_Ck`	$[p_k\lambda_k D_kA_kD'_k]$			Free	Free	Free
`Gaussian_pk_L_B`	$[p_k\lambda B]$	Diagonal	Free	Equal	Equal	Axes
`Gaussian_pk_Lk_B`	$[p_k\lambda_k B]$			Free	Equal	Axes
`Gaussian_pk_L_Bk`	$[p_k\lambda B_k]$			Equal	Free	Axes
`Gaussian_pk_Lk_Bk`	$[p_k\lambda_k B_k]$			Free	Free	Axes
`Gaussian_pk_L_I`	$[p_k\lambda I]$	Spherical	Free	Equal	Equal	NA
`Gaussian_pk_Lk_I`	$[p_k\lambda_k I]$			Free	Equal	NA

Table 1

Characteristics and identifiers of the twenty-eight Gaussian models available in MIXMOD.

labels $\mathbf{z}^u = \{\mathbf{z}_{m+1}, \ldots, \mathbf{z}_n\}$. Moreover, MIXMOD allows to specify a weight for each unit. This option is useful, for instance, when different units take exactly the same vector value.

Example 1 *The following instructions load 150 individuals of Iris data set (in* MIXMOD *package) and a partial partition of Iris data set.*

```
-> iris_individuals=read('DATA/iris.dat',150,4);
-> iris_partial_partition=read('DATA/iris.partial.lab',150,3);
```

Example 2 *As in Iris data set, individuals 102 and 143 are identical, weights can be used; the followings instructions load 149 individuals without repetition, 148 with weight 1 and one (102) with weight 2.*

```
-> iris_individuals_norepeat=read('DATA/iris149.dat',149,4);
-> iris_weight=ones(149,1);
-> iris_weight(102)=2;
```

4.2 Distributional hypotheses

In the Gaussian model-based clustering considered by MIXMOD, all data $(\boldsymbol{x}_i, \mathbf{z}_i)$ $(i = 1, \ldots, n)$, $\mathbf{z}_i$ being often unavailable as previously described, are assumed to arise independently from the joint probability distribution $\prod_{k=1}^{K} (p_k \varphi(\boldsymbol{x}_i; \boldsymbol{\mu}_k, \Sigma_k))^{z_{ik}}$. In this statistical context, two commonly used maximum likelihood (m.l.) approaches have been retained in MIXMOD: the mixture approach and the classification approach.

The mixture approach begins with a density estimation problem since it aims to maximize the log-likelihood

$$L(\theta; \mathbf{x}, \mathbf{z}^\ell) = \sum_{i=1}^{m}\sum_{k=1}^{K} z_{ik} \ln\left(p_k \varphi(\boldsymbol{x}_i; \boldsymbol{\mu}_k, \Sigma_k)\right) + \sum_{i=m+1}^{n} \ln\left(\sum_{k=1}^{K} p_k \varphi(\boldsymbol{x}_i; \boldsymbol{\mu}_k, \Sigma_k)\right) \tag{3}$$

over the mixture parameter $\theta = (p_1, \ldots, p_K, \boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K, \Sigma_1, \ldots, \Sigma_K)$. Then, a partition $\hat{\mathbf{z}}^u$ is derived from the m.l. estimator $\hat{\theta}$ by assigning each $\boldsymbol{x}_i$ of $\mathbf{x}^u$ to the component k providing the largest conditional probability

$$t_k(\boldsymbol{x}_i; \hat{\theta}) = \frac{\hat{p}_k \varphi(\boldsymbol{x}_i; \hat{\boldsymbol{\mu}}_k, \hat{\Sigma}_k)}{\sum_{k'=1}^{K} \hat{p}_{k'} \varphi(\mathbf{x}_i; \hat{\boldsymbol{\mu}}_{k'}, \hat{\Sigma}_{k'})} \tag{4}$$

that such $\boldsymbol{x}_i$ arises from it: it corresponds to the MAP (*Maximum A Posteriori*) principle. Maximizing $L(\theta; \mathbf{x}, \mathbf{z}^\ell)$ can be performed in MIXMOD via the EM algorithm of Dempster et al. (1977) or by a stochastic version of EM called SEM (see for instance Celeux and Diebolt, 1985; McLachlan and Krishnan, 1997). We will describe the different ways to use these algorithms in a further section. Obviously, the estimator $\hat{\theta}$, and consequently $\hat{\mathbf{z}}^u$, will depend on both the Gaussian model (see the previous section) and the number of clusters at hand.

Example 3 *The following instructions run EM algorithm (default) on Iris data with three components and* $[p_k\lambda_k C]$ *Gaussian model (default) and display estimated labels.*

```
-> out=mixmod(iris_individuals,3);
-> out.modelOutput.proba.label
```

Example 4 *The following instructions run SEM algorithm on Iris data (with-*

out repetition) with weight information, 3 components, and $[p_k\lambda_k C_k]$ Gaussian model and display estimated labels. `mixmodInput` *is a useful initialization function described in Section 9.4.*

```
-> [criterion,model,strategy]=mixmodInput();
-> strategy.tabAlgo.name='SEM';
-> model=['Gaussian_pk_Lk_Ck'];
-> out=mixmod(iris_individuals_norepeat,3,iris_weight,model,strategy);
-> out.modelOutput.proba.label
```

4.4 Estimation by classification approach

The second approach available in MIXMOD is the classification approach where the indicator vectors $\mathbf{z}^u$, identifying the mixture component origin, are treated as unknown parameters. It aims to maximize the complete-data log-likelihood

$$CL(\theta, \mathbf{z}^u; \mathbf{x}, \mathbf{z}^l) = \sum_{i=1}^{n} \sum_{k=1}^{K} z_{ik} \ln \left(p_k \varphi(\boldsymbol{x}_i; \boldsymbol{\mu}_k, \Sigma_k) \right) \tag{5}$$

over both the parameter θ and the labels $\mathbf{z}^u$. The CL criterion can be maximized by making use of a classification version of the EM algorithm, the so-called CEM algorithm (Celeux and Govaert, 1992) which includes a classification step (C-step) between the E and M steps. In Section 7, different strategies making use of this algorithm will be consider to derive the m.l. estimate of θ.

Example 5 *The following instructions run CEM algorithm on Iris data with a partially known partition, $[p_k\lambda B]$ Gaussian model and three components.*

```
-> [criterion,model,strategy]=mixmodInput();
```

```
-> strategy.tabAlgo.name='CEM';
-> model=['Gaussian_pk_L_B'];
-> known_partition=list(iris_partial_partition);
-> out=mixmod(iris_individuals,3,known_partition,model,strategy);
-> label=out.modelOutput.proba.label
```

5 Model-based discriminant analysis

5.1 Data and aim

Data managed in MIXMOD for discriminant analysis are typically composed by n vectors $(\mathbf{x},\mathbf{z}) = \{(\boldsymbol{x}_1,\boldsymbol{z}_1),\ldots,(\boldsymbol{x}_n,\boldsymbol{z}_n)\}$, where $\boldsymbol{x}_i$ belongs to $\mathbb{R}^d$ and $\mathbf{z}$ defines a known partition of the observations into K classes. The aim is to estimate the group $\mathbf{z}_{n+1}$ of any new observation with vector $\boldsymbol{x}_{n+1}$ of $\mathbb{R}^d$ and with unknown label. Nevertheless, as in the clustering context, partial labelling of data is possible, so MIXMOD allows to consider situations where some labels $\mathbf{z}^u$ of $\mathbf{z}$ are unknown. Weighting the data is also available in discriminant analysis.

Example 6 *The following instruction read the complete partition of Iris data set.*

```
-> iris_partition=read('DATA/iris.lab',150,3);
```

5.2 Estimation

The same statistical assumptions than in the clustering situation are made and the mixture parameter θ is estimated by maximizing the log-likelihood (3) with the EM or the SEM algorithms. Note that, when $\mathbf{z}$ is completely known, the estimation of the model parameter reduces to a single run of the M-step of EM. Then, the estimate $\hat{\theta}$ allows to assign any new point $\mathbf{z}_{n+1}$ by the MAP procedure.

Example 7 *In Mixmod, discriminant analysis has two steps:*

- *Step 1: (M-step: estimation of the mixture parameter): the following instructions create strategy for this step, set the partition to initialize the algorithm and display the estimated parameter.*

```
-> [criterion,model,strategy]=mixmodInput("DAstep1");
-> strategy.initType.tabLabel=iris_partition;
-> out=mixmod(iris_individuals,3,strategy);
-> out.modelOutput.param
```

- *Step 2: (Map-step: assignment of a new point): the following instructions create strategy for this step, set the parameter to initialize the algorithm and display the assignment label of the new point.*

```
-> newPoint=read('DATA/iris_new_point.dat',1,4);
-> [criterion,model,strategy]=mixmodInput("DAstep2");
-> strategy.initType.tabParam=out.modelOutput.param;
-> out=mixmod(newPoint,3,strategy);
-> out.modelOutput.proba.label
```

6 An overview of MIXMOD's algorithms

6.1 EM algorithm

The EM algorithm aims to maximize the likelihood. Starting from an initial arbitrary parameter θ^0, the qth iteration of the EM algorithm consists of repeating the following E and M steps.

- **E-step:** Compute the conditional probabilities $t_{ik}^q = t_k(\boldsymbol{x}_i; \theta^{q-1})$ that $\boldsymbol{x}_i$ belongs to the kth cluster ($i = m+1, \ldots, n$) by using the current value θ^{q-1} of the parameter.
- **M-step:** The m.l. estimate θ^q of θ is updated using the conditional probabilities t_{ik}^q as conditional mixing weights. This step highly depends on the Gaussian model at hand (Celeux and Govaert, 1995).

6.2 SEM algorithm

In SEM, a S-step is incorporated between the E- and M-step of EM. It corresponds to a restoration of the unknown labels by drawing them at random from their current conditional distribution. In the M-step, the conditional mixing weights are no longer the conditional probabilities but they now correspond to the labels obtained in the new stochastic step. SEM does not converge pointwise. It generates a Markov chain whose stationary distribution is more or less concentrated around the m.l. parameter estimator. A natural parameter estimate from a SEM sequence $(\theta^q)_{q=r,\ldots,Q}$ is the mean $\sum_{q=r+1}^{Q} \theta^q/(Q-r)$ of the iterates values where the first r burn-in iterates have been discarded when computing this mean. An alternative estimate is to consider the parameter value leading to the highest likelihood in a SEM sequence.

6.3 CEM algorithm

The CEM algorithm incorporates a classification step between the E- and M-steps of EM. Contrary to SEM, this new step is deterministic since labels are obtained from a MAP procedure for the current parameter value. Like SEM, in the M-step, the conditional mixing weights are these new labels. CEM is a *K-means*-like algorithm and contrary to EM, it converges in a finite number of iterations. CEM is not maximizing the observed log-likelihood L (3) but is maximizing in θ and $\mathbf{z}^u$ the complete data log-likelihood CL (4). As a consequence, CEM is not expected to converge to the m.l. estimate of θ and yields inconsistent estimates of the parameters especially when the mixture components are overlapping or are in disparate proportions (McLachlan and Peel, 2000, Section 2.21).

6.4 M-step and Map functions

The M-step is simply deduced from the definition of the EM algorithm and the MAP procedure has already been described (see Section 4.3).

7 Strategies for using EM and other related algorithms

7.1 Initialization strategies

There are five different possible ways to start an algorithm in MIXMOD. Except the first one which is deterministic, it is recommended to repeat several times the set {starting stragegy/running algorithm} in order to select the solution providing the best criterion value. This criterion corresponds to the likelihood when the running algorithm is EM or SEM, and to the complete-data likelihood when it is CEM.

- An algorithm can be started from some user specifications as a particular partition $\mathbf{z}^{u0}$ or a particular mixture parameter θ^0. This possibility is available for EM, SEM and CEM.
- An algorithm can be started from a random mixture parameter θ^0. In MIXMOD, this random initial position is obtained by drawing at random component means in the data set, by fixing proportions to equality and by choosing a diagonal common variance matrix where the diagonal is equal to the empirical variance of each variable. Since this is probably the most employed way of initiating EM, CEM or SEM, it can be regarded as a reference strategy.
- The EM algorithm can be started from the position providing the highest complete-data likelihood after many runs of CEM started with random positions and stopped with stability of the CL criterion. The number of restarts of CEM is *a priori* unknown but will depend on the allotment of iterations chosen by the user (see Biernacki et al., 2003).
- The EM algorithm can be started from the position providing the highest

likelihood after many short runs of EM started with random positions. By a short run of EM, we mean that we do not wait for convergence and that we stop the algorithm as soon as $(L^q - L^{q-1})/(L^q - L^0) \leq 10^{-2}$, L^q denoting the observed log-likelihood at the qth iteration. Here 10^{-2} represents a threshold value which has to be chosen on a pragmatic ground. The number of restarts of short runs of EM is *a priori* unknown but will depend on the allotment of iterations chosen by the user (see Biernacki et al., 2003).

Example 8

```
-> [criterion,model,strategy]=mixmodInput();
-> strategy.initType.name='SMALL_EM';
-> out=mixmod(iris_individuals,3,strategy);
```

- The EM algorithm can be started from the position providing the highest likelihood in a sequence of SEM started with random positions and with an allotment of iterations chosen by the user (see Biernacki et al., 2003).

7.2 Stopping rules

In MIXMOD, there are three ways to stop an algorithm.

- The EM, SEM and CEM algorithms can be stopped after a pre-defined number of iterations.
- An algorithm can be stopped using a threshold for the relative change of the criterion at hand (the likelihood L or the classification likelihood CL). This possibility available with EM is not recommended since EM can encounter slow convergence situations. For CEM which is converging in a finite number of iterations, it is recommended to stop it at stationarity.

Example 9 *The following instructions run EM algorithm with a threshold* *of 0.0001.*

```
-> [criterion,model,strategy]=mixmodInput();
-> strategy.tabAlgo.stopRule='EPSILON';
-> strategy.tabAlgo.stopRuleValue=0.0001;
-> out=mixmod(iris_individuals,3,strategy);
```

- An algorithm can be stopped by combination of both previous criteria, i.e. as soon as one of the criteria is verified.

  Example 10 *The following instructions run EM algorithm with a maximum of 200 iterations and a threshold of 0.0001.*

```
-> [criterion,model,strategy]=mixmodInput();
-> strategy.tabAlgo.stopRule ='NBITERATION_EPSILON';
-> strategy.tabAlgo.stopRuleValue(1)=200;
-> strategy.tabAlgo.stopRuleValue(2)=0.0001;
-> out=mixmod(iris_individuals, 3, strategy);
```

7.3 Chained algorithms

In MIXMOD, it is possible to easily link the algorithms EM, SEM and CEM in all imaginable ways. For instance, this property is useful to propose some original initialization strategies that we now present.

Example 11 *In this example, the SEM algorithm (200 iterations) is followed by the EM algorithm (100 iterations). Algo1 is the default algorithm (EM) with the default number of iterations (100). Strategy is composed of algo1 and algo2.*

```
-> [criterion, model, strategy]=mixmodInput();
```

```
-> algo1 = strategy.tabAlgo;
-> algo2=algo1;
-> algo2.name='SEM';
-> algo2.stopRuleValue=200;
-> strategy.tabAlgo=list(algo2, algo1);
-> out=mixmod(iris_individuals,3,strategy);
```

8 Model selection

It is of high interest to automatically select a Gaussian model M and the number K of mixture components. However, choosing a sensible mixture model is highly dependent of the modelling purpose; consequently, a difference is made now between the cluster and the discriminant analysis point of view.

8.1 Cluster analysis purpose

In MIXMOD, three criteria are available in an unsupervised setting: BIC, ICL and NEC. It can be noted that if no information on K is available, it is recommended to vary it between 1 and the smallest integer larger than $n^{0.3}$ (see Bozdogan, 1993).

In a density estimation perspective, BIC (for Bayesian Information Criterion) must be preferred. Denoting by $\nu_{M,K}$ the number of free parameters in the Gaussian model M with K clusters, BIC is expressed by the following penalization of the maximum log-likelihood $L_{M,K}$:

$$\mathrm{BIC}_{M,K} = -2L_{M,K} + \nu_{M,K} \ln(n). \tag{6}$$

The couple (M, K) leading to the lowest value of BIC has to be retained. Despite the fact that regularity conditions necessary for deriving BIC (Schwarz, 1978) are not fulfilled for mixtures, it has been proved, for a large family of mixtures, that the criterion BIC is consistent (Kéribin, 2000) and has been proved to be efficient on a practical ground (see for instance Fraley and Raftery, 1998).

But in a cluster analysis perspective, ICL and NEC can provide more parsimonious and robust answers. In order to take into account the ability of the mixture model to give evidence for a clustering structure of the data, an alternative to the BIC criterion is to consider the ICL (for Integrated Complete-data Likelihood) (see Biernacki et al., 2000) expressed by

$$\mathrm{ICL}_{M,K} = \mathrm{BIC}_{M,K} - 2 \sum_{i=m+1}^{n} \sum_{k=1}^{K} \hat{z}_{ik} \ln(t_{ik}), \tag{7}$$

where $t_{ik} = t_k(\boldsymbol{x}_i; \hat{\theta}_{M,K})$ with $\hat{\theta}_{M,K}$ the m.l. estimate and where $\hat{\mathbf{z}} = \mathrm{MAP}(\hat{\theta}_{M,K})$. This criterion, to be minimized, is simply the BIC criterion penalized by an entropy term which measures the overlap of the clusters. The NEC (for Normalized Entropy Criterion) criterion of Celeux and Soromenho (1994) uses a similar entropy term $E_K = -\sum_{i=m+1}^{n} \sum_{k=1}^{K} t_{ik} \ln(t_{ik})$ but is essentially devoted to choose the number of mixture components K, rather that the model parameterization M (Biernacki and Govaert, 1999). This criterion, to be minimized, is expressed by

$$\mathrm{NEC}_K = \frac{E_K}{L_K - L_1}. \tag{8}$$

Note that NEC_1 is not defined. Biernacki et al. (1999) proposed the following efficient rule to deal with this problem: Let $K^\star$ be the value minimizing NEC_K, $(2 \leq K \leq K_{\mathrm{sup}})$, K_{sup} being an upper bound for the number of mixture components. We choose $K^\star$ clusters if $\mathrm{NEC}_{K^\star} \leq 1$, otherwise we declare no clustering structure in the data.

Example 12 *This exemple selects the best Gaussian model between* $[p_k\lambda_k C]$ *or* $[p_k\lambda_k C_k]$ *Gaussian models and the best number of mixture components between 2 and 3 with the ICL criterion. The best model and number of components are displayed.*

```
-> model=['Gaussian_pk_Lk_C','Gaussian_pk_Lk_Ck'];
-> K=[2,3];
-> criterion=['ICL'];
-> out=mixmod(iris_individuals,K,model,criterion);
-> out.modelOutput.type
-> out.modelOutput.nbCluster
```

Example 13 *The selection is done with ICL, NEC and BIC criteria. The best Gaussian model for BIC criterion (the 3rd criterion) and the best number of components for NEC criterion (the 2nd criterion) are displayed.*

```
-> model=['Gaussian_pk_Lk_C','Gaussian_pk_Lk_Ck'];
-> K=[2,3];
-> criterion=['ICL','NEC','BIC'];
-> out=mixmod(iris_individuals,K,model,criterion);
-> out.modelOutput(3).type
-> out.modelOutput(2).nbCluster
```

8.2 Discriminant analysis purpose

In this situation, note that only the model M has to be selected. In MIXMOD, two criteria are proposed in a supervised setting: BIC and the cross validated error rate (CV). The CV criterion, valid only in the discriminant analysis (supervised) context, is defined by

$$\mathrm{CV}_M = \frac{1}{m}\sum_{i=1}^{m}\delta(\hat{\mathbf{z}}_i^{(i)}, \mathbf{z}_i) \tag{9}$$

with δ the 0-1 cost and $\hat{\mathbf{z}}_i^{(i)}$ the group to which $\boldsymbol{x}_i$ is assigned when designing the assignment rule from the entire data set $(\mathbf{x}, \mathbf{z})$ without $(\boldsymbol{x}_i, \mathbf{z}_i)$. Fast estimation of the n discriminant rules is implemented in the Gaussian situation when $m = n$, i.e. when all labels are known (Biernacki and Govaert, 1999).

In MIXMOD, according to a line detailed in Bensmail and Celeux (1996), it is possible to select one of the Gaussian models by minimization of the cross validated error rate. But, it is important to stress that this cross validated error rate is an optimistic estimate of the actual error rate. Actually, this is a situation where the method includes the selection of one model among several ones. Thus, there is a need to assess the actual error rate from an independent sample. Roughly speaking, three samples are needed: a *training* sample to estimate the parameters of the model, a *validation* sample to choose one of the model and a *test* sample to assess the actual error rate of the whole method. It means that when using cross validation to assess the error rate, there is the need to perform a *double* cross validation to get an unbiased estimate. In practice, such a double cross validation will by painfully slow and it is not currently implemented in MIXMOD. To assess a decision rule involving the choice of a model in MIXMOD, it is necessary to discard at random a test sample from the whole data set. This test sample will be used to assess the actual error rate of the whole procedure.

Example 14 *In this discrimination example, the best Gaussian model between $[p_k\lambda_k C]$ and $[p_k\lambda_k C_k]$ is chosen with cross-validation.*

```
-> [criterion,model,strategy]=mixmodInput("DAstep1");
-> strategy.initType.tabLabel=iris_partition;
-> criterion=['CV'];
-> model=['Gaussian_pk_Lk_C','Gaussian_pk_Lk_Ck'];
-> out=mixmod(iris_individuals,3,criterion,model,strategy);
-> out.modelOutput.type
```

9 Companion functions

The MATLAB or SCILAB environment allows to obtain easily high levels functions, typically some graphical displays.

9.1 Graphical displays of criterion values

One of the optional outputs of `mixmod` function is a four dimension matrix with values of all requested criteria for all requested strategies, all requested numbers of mixture components and all requested Gaussian models.

Example 15 *This example selects with BIC and ICL criteria the best Gaussian model (among all the Gaussian models) and the best number of mixture components (1,2,3 or 4). The variable* `outTab` *contains 224 values composed by (1 strategy) $\times$ (4 numbers of components) $\times$ (28 Gaussian models) $\times$ (2 criteria). The last instruction plots (see Figure 1) the ICL criterion values (the 2nd criterion) for the first (and unique) strategy, the 4 numbers of components and the Gaussian model $[p\lambda B_k]$ (the 5th one).*

```
-> [criterion,model,strategy]=mixmodInput('allModel');
```

```
-> K=[1 2 3 4];
-> criterion=['BIC','ICL'];
-> [out, outTab]=mixmod(iris_individuals,K,model,criterion);
-> plot2d(outTab(1,:,5,2));
```

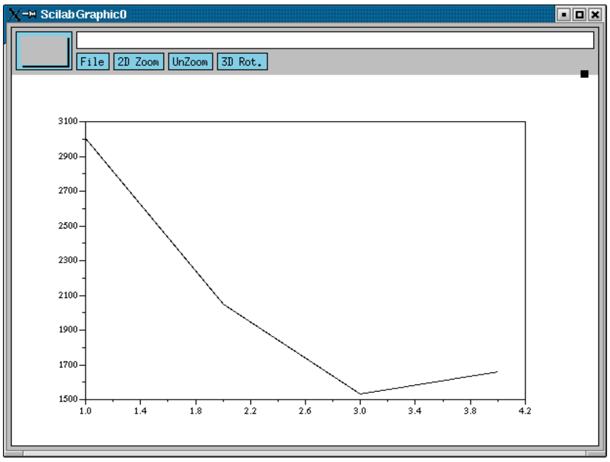

Fig. 1. ICL criterion values for different component numbers.

9.2 MIXMODVIEW *function for graphics*

MIXMOD provides the `mixmodView` function to visualize the outputs. This function allows to display graphics (density, iso-density, ...) in 1-D, 2-D and 3-D spaces, from output coming from an execution of the `mixmod` function.

Example 16 *Consider, for instance, the selection of the best Gaussian model for K=3 with the BIC criterion*

```
-> [criterion,model,strategy]=mixmodInput('allModel');
-> out=mixmod(iris_individuals,3,model);
```

the following graphics can be made.

- *Figure 2 displays iso-density component and density mixture in the first*

PCA space with the following command:

```
-> mixmodView(iris_individuals,out,list('pca1D'),
                  'isoDensityComponent','densityMixture');
```

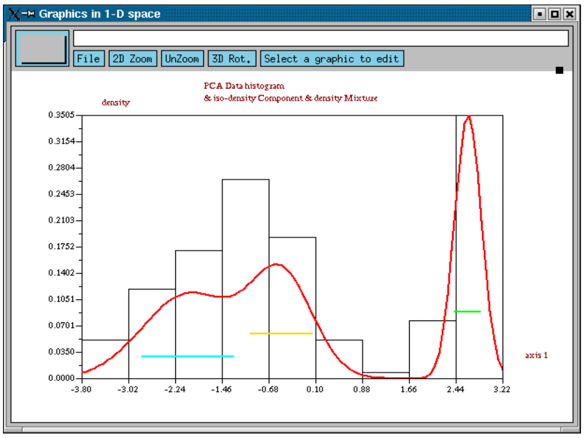

Fig. 2. Iso-density component and density mixture in the first PCA space.

- *Figure 3 displays class limit, iso-density component and individuals in the first PCA 2D space.*

```
-> mixmodView(iris_individuals,out,list('pca2D'),
                  'isoDensityComponent','bothLimit','point');
```

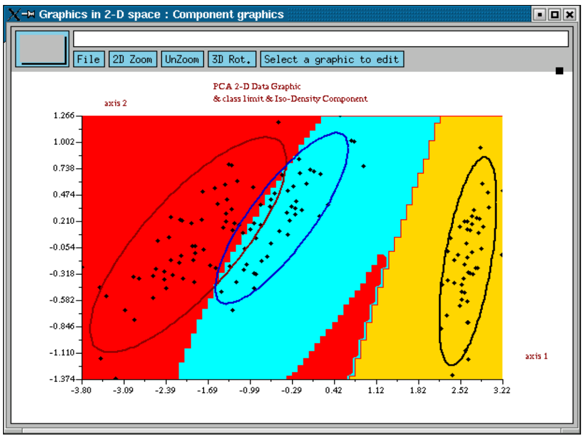

Fig. 3. Class limit, iso-density component and individuals in the first PCA 2D space.

- *Figure 4 displays density mixture in the first PCA 2D space.*

```
-> mixmodView(iris_individuals,out,list('pca2D'),
```

```
'densityMixture');
```

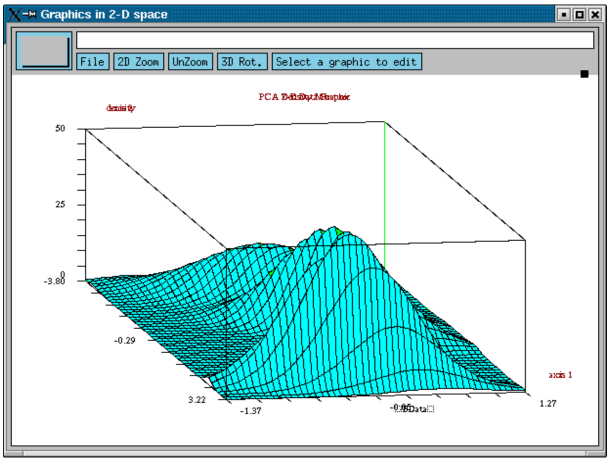

Fig. 4. Density mixture in the first PCA 2D space.

- *Figure 5 displays individuals and labels in the first PCA 3D space.*

  ```
  ->mixmodView(iris_individuals,out,list('pca3D'),'class');
  ```

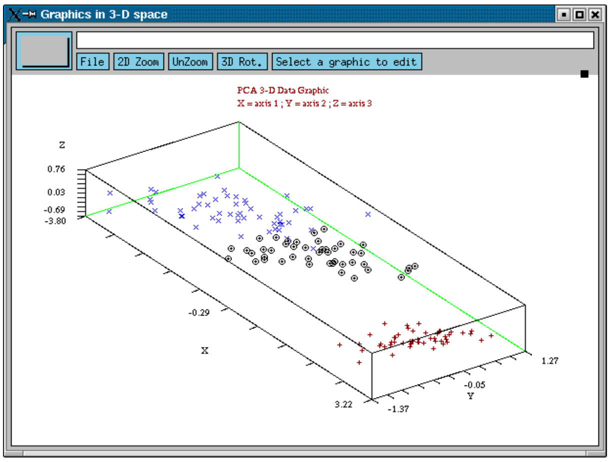

Fig. 5. Individuals and labels in the first PCA 3D space.

9.3 printMixmod *function for summary*

The `printMixmod` function can be used to summarize outputs of `mixmod` function.

Example 17 *With the last example,* `printMixmod` *function displays a readable summary of the variable* `out` *(input conditions, criterion value, likelihood, complete-data likelihood, estimated parameter, ...).*

```
-> printMixmod(out);
```

9.4 INPUTMIXMOD *function for input facilities*

The `inputMixmod` function allows to create some SCILAB (or MATLAB) structures which can be used in `mixmod` function.

Example 18 *To create three variables initialized to the default criterion (BIC), the default Gaussian model ($[p_k \lambda_k DAD']$) and the default strategy (initialization at random, EM algorithm, algorithm stopped if the number of iterations is greater than 200), type:*

```
-> [criterion,model,strategy]=mixmodInput();
```

Example 19 *To create a model structure with all Gaussian models, or all diagonal ones, or all spherical ones, or all general ones, type:*

```
-> [criterion,model,strategy]=mixmodInput('allModel');
-> [criterion,model,strategy]=mixmodInput('diagonalModel');
-> [criterion,model,strategy]=mixmodInput('sphericalModel');
-> [criterion,model,strategy]=mixmodInput('generalModel');
```

Example 20 *To create a criterion structure with all criteria, type:*

```
-> [criterion,model,strategy]=mixmodInput('allCriteria');
```

Example 21 *To create a strategy for the first, the second or both steps of*

Discriminant Analysis, type:

```
-> [criterion,model,strategy]=mixmodInput('DAstep1');
-> [criterion,model,strategy]=mixmodInput('DAstep2');
-> [criterion,model,tabStrategy]=mixmodInput('DAallStep');
```

10 Further developments of MIXMOD

In the Gaussian framework, MIXMOD is now a relatively reliable and fast software. In one hand, remarks that users have deposited on the web site have contributed to correct some bugs, and, in the other hand, the speed of the code has been improved throughout the different versions. Beyond these two important features that will be continued in the future, the proposals of the users are welcome not only on the `mixmod` function but also on other functionalities as `mixmodView` (the web site is a natural way to collect and exchange such information).

In the next months, the version 2.0 of MIXMOD will be available and will add clustering and discriminant analysis for multivariate binary or qualitative data since using such data is common in some various and important fields (ecology, image analysis, etc.). In this context, mixture of Bernoulli or multinomial distributions will be employed and some parsimonious original models will be proposed. In further evolutions of MIXMOD 2.0, it will be also useful to propose a way to treat in the same exercise mixed data with both continuous and qualitative variables.

Acknowledgements: The authors are grateful to Grégory Noulin and Yann Vernaz for their contribution concerning the MIXMOD coding. They are also indebted with the support of their respective institutions. In particular, they are pleased to ackknowledge the support of a ODL grant from INRIA during two years.

References

Banfield, J. D., Raftery, A. E., 1993. Model-based Gaussian and non-Gaussian clustering. Biometrics 49, 803–821.

Bensmail, H., Celeux, G., 1996. Regularized Gaussian discriminant analysis through eigenvalue decomposition. Journal of the American Statistical Association 91 (2), 1743–17448.

Biernacki, C., Celeux, G., Govaert, G., 1999. An improvement of the NEC criterion for assessing the number of clusters in a mixture model. Pattern Recognition Letters 20, 267–272.

Biernacki, C., Celeux, G., Govaert, G., 2000. Assessing a mixture model for clustering with the integrated completed likelihood. IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (7), 719–725.

Biernacki, C., Celeux, G., Govaert, G., 2003. Choosing starting values for the EM algorithm for getting the highest likelihood in multivariate gaussian mixture models. Computational Statistics and Data Analysis 41, 561–575.

Biernacki, C., Govaert, G., 1999. Choosing models in model-based clustering and discriminant analysis. J. Statis. Comput. Simul. 64, 49–71.

Bozdogan, H., 1993. Choosing the number of component clusters in the mixture-model using a new informational complexity criterion of the inverse-fisher information matrix. In: Opitz, O., Lauritzen, B., Klar, R. (Eds.), Information and Classification. Springer-Verlag, Heidelberg, pp. 40–54.

Celeux, G., Diebolt, J., 1985. The SEM algorithm: A probabilistic teacher algorithm derived from the EM algorithm for the mixture problem. Computational Statistics Quarterly 2, 73–82.

Celeux, G., Govaert, G., 1992. A classification EM algorithm for clustering and two stochastic versions. Computational Statistics and Data Analysis

14 (3), 315–332.

Celeux, G., Govaert, G., 1995. Gaussian parsimonious clustering models. Pattern Recognition 28 (5), 781–793.

Celeux, G., Soromenho, G., 1994. An entropy criterion for assessing the number of clusters in a mixture model. Journal of Classification 13, 195–212.

Dempster, A. P., Laird, N. M., Rubin, D. B., 1977. Maximum likelihood from incomplete data via the EM algorithm (with discussion). Journal of the Royal Statistical Society B 39, 1–38.

Diday, E., Govaert, G., 1974. Classification avec distance adaptative. C. R. Acad. Sc. Paris, série A 278, 993–995.

Fisher, R. A., 1936. Multiple measurements in taxinomie problems. Annals of Eugenics 7 (2), 179–188.

Fraley, C., Raftery, A. E., 1998. How many clusters? Which clustering method? Answers via model-based cluster analysis. Computer Journal 41, 578–588.

Friedman, H. P., Rubin, J., 1967. On some invariant criteria for grouping data. Journal of American Statistical Association 62, 1159–1178.

Kéribin, C., 2000. Consistent estimation of the order of mixture models. Sankhyã, Series A 1, 49–66.

Maronna, R., Jacovkis, P., 1974. Multivariate clustering procedure with variable metrics. Biometrics 30, 499–505.

McLachlan, G. J., 1992. Discriminant Analysis and Statistical Pattern Recognition. Wiley, New York.

McLachlan, G. J., Krishnan, K., 1997. The EM Algorithm. Wiley, New York.

McLachlan, G. J., Peel, D., 2000. Finite Mixture Models. Wiley, New York.

Schroeder, A., 1976. Analyse d'un mélange de distributions de probabilité de même type. Revue de Statistique Appliquée 24 (1), 39–62.

Schwarz, G., 1978. Estimating the number of components in a finite mixture model. Annals of Statistics 6, 461–464.

Scott, A. J., Symons, M. J., 1971. Clustering methods based on likelihood ratio criteria. Biometrics 27, 387–397.

Ward, J., 1963. Hierarchical grouping to optimize an objective function. Journal of the American Statistical Association" 58, 236–244.