Journal of Statistical Software

January 2012, Volume 46, Issue 6. http://www.jstatsoft.org/

HDclassif: An R Package for Model-Based Clustering and Discriminant Analysis of High-Dimensional Data

Laurent Bergé
Université Bordeaux IV

Charles Bouveyron
Université Paris 1

Stéphane Girard
INRIA Rhône-Alpes

Abstract

This paper presents the R package **HDclassif** which is devoted to the clustering and the discriminant analysis of high-dimensional data. The classification methods proposed in the package result from a new parametrization of the Gaussian mixture model which combines the idea of dimension reduction and model constraints on the covariance matrices. The supervised classification method using this parametrization is called high dimensional discriminant analysis (HDDA). In a similar manner, the associated clustering method is called high dimensional data clustering (HDDC) and uses the expectation-maximization algorithm for inference. In order to correctly fit the data, both methods estimate the specific subspace and the intrinsic dimension of the groups. Due to the constraints on the covariance matrices, the number of parameters to estimate is significantly lower than other model-based methods and this allows the methods to be stable and efficient in high dimensions. Two introductory examples illustrated with R codes allow the user to discover the `hdda` and `hddc` functions. Experiments on simulated and real datasets also compare HDDC and HDDA with existing classification methods on high-dimensional datasets. **HDclassif** is a free software and distributed under the general public license, as part of the R software project.

Keywords: model-based classification, high-dimensional data, discriminant analysis, clustering, Gaussian mixture models, parsimonious models, class-specific subspaces, R package.

1. Introduction

Classification in high-dimensional spaces is a recurrent problem in many fields of science, for instance in image analysis or in spectrometry. Indeed, the data used in these fields are often high-dimensional and this penalizes most of the classification methods. In this paper, we

focus on model-based approaches. We refer to Bock (1996) for a review on this topic. In this context, popular classification methods are based on the Gaussian mixture model (McLachlan and Peel 2000) and show a disappointing behavior when the size of the dataset is too small compared to the number of parameters to estimate. This well-known phenomenon is called the *curse of dimensionality* and was first identified by Bellman (1957). We refer to Pavlenko (2003) and Pavlenko and Rosen (2001) for a theoretical study of the effect of dimension in the model-based classification. To avoid over-fitting, it is necessary to find a balance between the number of parameters to estimate and the generality of the model. Recently, Bouveyron, Girard, and Schmid (2007a,b) have proposed a new parametrization of the Gaussian mixture model which takes into account the specific subspace around which each group is located. This parametrization therefore limits the number of parameters to estimate while proposing a flexible modeling of the data. The use of this re-parametrization in discriminant analysis yielded a new method called high dimensional discriminant analysis (HDDA, Bouveyron *et al.* 2007b) and the associated clustering method has been named high dimensional data clustering (HDDC, Bouveyron *et al.* 2007a).

The R (R Development Core Team 2011) package **HDclassif** (currently in version 1.2.1) implements these two classification methods for the clustering and the discriminant analysis of high-dimensional data. This paper briefly reviews in Section 2 the methodology of the HDDA and HDDC methods. Section 3 focuses on technical details of the learning and predicting routines. The practical use of the package is illustrated and compared to well-established classification packages in Section 4 on introductory and real-world datasets. Section 5 presents applications of the package to optical character recognition and to mass-spectrometry. Finally, some concluding remarks are provided in Section 6. The package is available from the Comprehensive R Archive Network at `http://CRAN.R-projects.org/package=HDclassif`.

2. Gaussian models for high-dimensional data classification

Classification is a statistical field which includes two techniques: supervised and unsupervised classifications. Supervised classification, also called discriminant analysis, aims to associate a new observation x with one of K known classes through a learning set of labeled observations. Conversely, unsupervised classification aims to segment a set of unlabeled observations into K homogeneous groups. Unsupervised classification is also known as clustering. We refer to McLachlan (1992) for more details on the general classification framework.

In both contexts, a popular approach is the use of the Gaussian mixture model which relies on the assumption that each class can be represented by a Gaussian density. This approach assumes that the observations $\{x_1, \ldots, x_n\}$ are independent realizations of a random vector $X \in \mathbb{R}^p$ with density:

$$f(x, \theta) = \sum_{k=1}^{K} \pi_k \phi(x; \mu_k, \Sigma_k), \tag{1}$$

where π_k is the mixture proportion of the kth component and ϕ is the Gaussian density parametrized by the mean μ_k and the covariance matrix Σ_k. This model gives rise in the supervised context to the well-known quadratic discriminant analysis (QDA). Unfortunately, this method requires the estimation of a very large number of parameters (proportional to p^2) and therefore faces numerical problems in high-dimensional spaces. Hopefully, due to the *empty space* phenomenon (Scott and Thompson 1983), it can be assumed that high-

dimensional data live around subspaces with a dimension lower than p. Recently, Bouveyron *et al.* (2007a,b) have introduced a new parametrization of the Gaussian mixture model which takes into account the specific subspace around which each cluster is located and therefore limits the number of parameters to estimate.

2.1. The Gaussian model $[a_{kj}b_kQ_kd_k]$ and its submodels

As in the classical Gaussian mixture model framework (McLachlan 1992), we assume that class conditional densities are Gaussian $\mathcal{N}_p(\mu_k, \Sigma_k)$ with means μ_k and covariance matrices Σ_k , for $k = 1, \ldots, K$. Let Q_k be the orthogonal matrix with the eigenvectors of Σ_k as columns and Δ_k be the diagonal matrix which contains the eigenvalues of Σ_k such that:

$$\Delta_k = Q_k^t \Sigma_k Q_k. \tag{2}$$

The matrix Δ_k is therefore the covariance matrix of the kth class in its eigenspace. It is further assumed that Δ_k can be divided into two blocks:

$$\Delta_k = \begin{pmatrix} \boxed{\begin{matrix} a_{k1} & & 0 \\ & \ddots & \\ 0 & & a_{kd_k} \end{matrix}} & \mathbf{0} \\ \mathbf{0} & \boxed{\begin{matrix} b_k & & 0 \\ & \ddots & \\ & & \ddots \\ 0 & & b_k \end{matrix}} \end{pmatrix} \begin{matrix} \left.\vphantom{\begin{matrix} a \\ a \\ a \end{matrix}}\right\} d_k \\ \left.\vphantom{\begin{matrix} a \\ a \\ a \end{matrix}}\right\} (p - d_k) \end{matrix} \tag{3}$$

with $a_{kj} > b_k$, $j = 1, \ldots, d_k$, and where $d_k \in \{1, \ldots, p-1\}$ is unknown. This Gaussian model will be denoted to by $[a_{kj}b_kQ_kd_k]$ in the sequel. With these notations and from a practical point of view, one can say that the parameters $a_{k1}, \ldots, a_{kd_k}$ model the variance of the actual data of the kth class and the unique parameter b_k can be viewed as modeling the variance of the noise. The dimension d_k can be considered as well as the intrinsic dimension of the latent subspace of the kth group which is spanned by the d_k first column vectors of Q_k. Let us remark that if we constrain d_k to be equal to $(p-1)$ for all $k = 1, \ldots, K$, the model $[a_{kj}b_kQ_kd_k]$ then reduces to the classical Gaussian mixture model with full covariance matrices for each mixture component which yields QDA in the supervised framework.

By fixing some parameters to be common within or between classes, it is possible to obtain particular models which correspond to different regularizations. Fixing the dimensions d_k to be common between the classes yields the model $[a_{kj}b_kQ_kd]$ which is the model proposed in Tipping and Bishop (1999) in the unsupervised classification framework. As a consequence, the modeling presented above encompasses the mixture of probabilistic principal component analyzers introduced in Tipping and Bishop (1999) and extended in McLachlan, Peel, and Bean (2003). It is also important to notice that this modeling is closely related with the recent works of McNicholas and Murphy (2008a,b); Baek, McLachlan, and Flack (2009) and Bouveyron and Brunet (2012). Moreover, our approach can be combined with a "parsimonious model" strategy to further limit the number of parameters to estimate. It is indeed possible to add constraints on the different parameters to obtain more regularized models. Fixing the first d_k eigenvalues to be common within each class, we obtain the more restricted model $[a_kb_kQ_kd_k]$. The model $[a_kb_kQ_kd_k]$ often gives satisfying results, i.e., the assumption that

Model	Number of parameters	Asymptotic order	Number of parameters $K = 4$, $d = 10$, $p = 100$
$[a_{kj}b_kQ_kd_k]$	$\rho + \bar{\tau} + 2K + D$	Kpd	4231
$[a_{kj}bQ_kd_k]$	$\rho + \bar{\tau} + K + D + 1$	Kpd	4228
$[a_kb_kQ_kd_k]$	$\rho + \bar{\tau} + 3K$	Kpd	4195
$[ab_kQ_kd_k]$	$\rho + \bar{\tau} + 2K + 1$	Kpd	4192
$[a_kbQ_kd_k]$	$\rho + \bar{\tau} + 2K + 1$	Kpd	4192
$[abQ_kd_k]$	$\rho + \bar{\tau} + K + 2$	Kpd	4189
$[a_{kj}b_kQ_kd]$	$\rho + K(\tau + d + 1) + 1$	Kpd	4228
$[a_{kj}bQ_kd]$	$\rho + K(\tau + d) + 2$	Kpd	4225
$[a_kb_kQ_kd]$	$\rho + K(\tau + 2) + 1$	Kpd	4192
$[ab_kQ_kd]$	$\rho + K(\tau + 1) + 2$	Kpd	4189
$[a_kbQ_kd]$	$\rho + K(\tau + 1) + 2$	Kpd	4189
$[abQ_kd]$	$\rho + K\tau + 3$	Kpd	4186
$[a_jbQd]$	$\rho + \tau + d + 2$	pd	1360
$[abQd]$	$\rho + \tau + 3$	pd	1351
Full-GMM	$\rho + Kp(p + 1)/2$	$Kp^2/2$	20603
Com-GMM	$\rho + p(p + 1)/2$	$p^2/2$	5453
Diag-GMM	$\rho + Kp$	$2Kp$	803
Sphe-GMM	$\rho + K$	Kp	407

Table 1: Properties of the HD models and some classical Gaussian models: K is the number of components, d and d_k are the intrinsic dimensions of the classes, p is the dimension of the observation space, $\rho = Kp + K - 1$ is the number of parameters required for the estimation of means and proportions, $\bar{\tau} = \sum_{k=1}^{K} d_k[p - (d_k + 1)/2]$ and $\tau = d[p - (d + 1)/2]$ are the number of parameters required for the estimation of orientation matrices Q_k, and $D = \sum_{k=1}^{K} d_k$. For asymptotic orders, the assumption that $K \ll d \ll p$ is made.

each matrix Δ_k contains only two different eigenvalues, a_k and b_k, seems to be an efficient way to regularize the estimation of Δ_k. The good practical behavior of this specific model can be explained by the fact that the variance in the estimation of a_k, which is the mean of the $a_{k1}, \ldots, a_{kd_k}$, is less than the variance in the separate estimations of the $a_{k1}, \ldots, a_{kd_k}$. Therefore, the bias introduced by this assumption on the model seems to be balanced by the limited variance in the estimation of model parameters. Another type of regularization is to fix the parameters b_k to be common between the classes. This yields the models $[a_{kj}bQ_kd_k]$ and $[a_kbQ_kd_k]$ which assume that the variance outside the class specific subspaces is common. This can be viewed as modeling the noise outside the latent subspace of the group by a single parameter b which could be appropriate when the data are obtained in a common acquisition process. Among the 28 models proposed in the original articles (Bouveyron *et al.* 2007a,b), 14 models have been selected to be included in the package for their good behaviors in practice. Table 1 lists the 14 models available in the package and their corresponding complexity (i.e., the number of parameters to estimate). The complexity of classical Gaussian models is also provided in a comparison purpose. The Full-GMM model refers to the classical Gaussian mixture model with full covariance matrices, the Com-GMM model refers to the Gaussian mixture model for which the covariance matrices are assumed to be equal to a common covariance matrix ($S_k = S$, $\forall k$), Diag-GMM refers to the Gaussian mixture model for which

$\Sigma_k = \text{diag}(s_{k1}^2, \ldots, s_{kp}^2)$ with $s_k^2 \in \mathbb{R}_+^p$ and Sphe-GMM refers to the Gaussian mixture model for which $\Sigma_k = s_k^2 I_p$ with $s_k^2 \in \mathbb{R}_+$. Remark that the Com-GMM model is the model of the popular linear discriminant analysis (LDA) in the supervised context.

2.2. High dimensional discriminant analysis

The use of the models presented in the previous paragraph has given birth to a method for high-dimensional discriminant analysis called HDDA (Bouveyron *et al.* 2007b). HDDA is made of a learning step, in which model parameters are estimated from a set of learning observations, and a classification step which aims to predict the class belonging of new unlabeled observations. In the context of supervised classification, the learning data are complete, i.e., a label z_i indicating the class belonging is available for each observation x_i of the learning dataset. The estimation of model parameters is therefore direct through the maximum likelihood method and parameter estimators are closed-form. Estimators for model parameters can be found in Bouveyron *et al.* (2007b). Once the model parameters learned, it is possible to use HDDA for predicting the class of a new observation x using the classical *maximum a posteriori* (MAP) rule which assigns the observation to the class with the largest posterior probability. Therefore, the classification step mainly consists in computing, for each class $k = 1, \ldots, K$, $\mathbb{P}(Z = k|X = x) = 1 \Big/ \sum_{\ell=1}^{K} \exp\left(\frac{1}{2}(\Gamma_k(x) - \Gamma_\ell(x))\right)$ where the cost function $\Gamma_k(x) = -2\log(\pi_k \phi(x; \mu_k, \Sigma_k))$ has the following form in the case of the model $[a_k b_k Q_k d_k]$:

$$\Gamma_k(x) = \frac{1}{a_k}\|\mu_k - P_k(x)\|^2 + \frac{1}{b_k}\|x - P_k(x)\|^2 + d_k \log(a_k) + (p - d_k)\log(b_k) - 2\log(\pi_k), \quad (4)$$

where P_k is the projection operator on the latent subspace of the kth class. Let us notice that $\Gamma_k(x)$ is mainly based on two distances: the distance between the projection of x on the latent subspace and the mean of the class and the distance between the observation and the latent subspace. This function favors the assignment of a new observation to the class for which it is close to the subspace and for which its projection on the class subspace is close to the mean of the class. The variance terms a_k and b_k balance the importance of both distances.

2.3. High dimensional data clustering

In the unsupervised classification context, the use of the models presented above yielded a model-based clustering method called HDDC (Bouveyron *et al.* 2007b). Conversely to the supervised case, the data at hand in the clustering context are not complete (i.e., the labels are not observed for the observations of the dataset to cluster). In such a situation, the direct maximization of the likelihood is an intractable problem and the expectation-maximization (EM) algorithm (Dempster, Laird, and Rubin 1977) can be used to estimate the mixture parameters by iteratively maximizing the likelihood. The EM algorithm alternates between the following E and M steps at each iteration q:

- The E step computes the posterior probabilities $t_{ik}^{(q)} = \mathbb{P}(Z = k|X = x_i)$ through Equation (4) using the model parameters estimated at iteration $q - 1$,
- The M step updates the estimates of model parameters by maximizing the expectation of the complete likelihood conditionally to the posterior probabilities $t_{ik}^{(q)}$. Update formulas for model parameters can be found in Bouveyron *et al.* (2007b).

The EM algorithm stops when the difference between the estimated values of the likelihood at two consecutive iterations is smaller than a given threshold.

3. Learning and predicting routines

This section first focuses on technical issues related to the inference in HDDA and HDDC. In a second part, details are given about the inputs and outputs of both methods.

3.1. Implementation issues

We discuss here the implementation issues related to the determination of the hyper-parameters and to the case where the number of observations n is smaller than the space dimension p.

Estimation of the hyper-parameters

The use of maximum likelihood or the EM algorithm for parameter estimation makes the methods HDDA and HDDC almost automatic, except for the estimation of the hyper-parameters $d_1, \ldots, d_k$ and K. Indeed, these parameters cannot be determined by maximizing the likelihood since they both control the model complexity. The estimation of the intrinsic dimensions $d_1, \ldots, d_k$ is a difficult problem with no unique technique to use. In Bouveyron *et al.* (2007b), the authors proposed a strategy based on the eigenvalues of the class conditional covariance matrix Σ_k of the kth class. The j-th eigenvalue of Σ_k corresponds to the fraction of the full variance carried by the j-th eigenvector of Σ_k. The class specific dimension d_k, $k = 1, \ldots, K$, is estimated through Cattell's scree-test (Cattell 1966) which looks for a break in the eigenvalues scree. The selected dimension is the one for which the subsequent eigenvalue differences are smaller than a threshold. The threshold can be provided by the user, selected through cross-validation in the supervised case or using BIC (Schwarz 1978) in the unsupervised case. In the clustering case, the number of clusters K may have to be estimated as well and can be chosen thanks to the BIC criterion. In the specific case of the models $[a_k b_k Q_k d_k]$, $[a_k b_k Q_k d]$, $[ab Q_k d_k]$ and $[ab Q_k d]$, it has been recently proved by Bouveyron, Celeux, and Girard (2011) that the maximum likelihood estimate of the intrinsic dimensions d_k is asymptotically consistent.

Case $n_k < p$

Furthermore, in the special case where the number of observations of a class, n_k, is smaller than the dimension p, the parametrization presented in the previous section allows to use a linear algebra trick. Since the data do not live in a subspace larger than the number of observations it contains, the intrinsic dimension d_k cannot be larger than the number of observations of the class. Then, there is no need to compute all the eigenvalues and eigenvectors of the empirical covariance matrix $W_k = \mathcal{X}_k^t \mathcal{X}_k$, where $\mathcal{X}_k$ is the $n_k \times p$ matrix containing the centered observations of the k-th class. Indeed, it is faster and more numerically stable to calculate, when $n_k < p$, the eigenvalues and eigenvectors of the inner product matrix $\mathcal{X}_k \mathcal{X}_k^t$ which is a $n_k \times n_k$ matrix. Let ν_{kj} be the eigenvector associated to the j-th eigenvalue λ_{kj} of the matrix $\mathcal{X}_k \mathcal{X}_k^t$, then for $j = 1, \ldots, d_k$:

$$\mathcal{X}_k \mathcal{X}_k^t v_{kj} = \lambda_{kj} \nu_{kj}.$$

Therefore, the eigenvector of W_k associated to the eigenvalue λ_{kj} can be obtained by multi-

	Free dimensions		Common dimensions	
Variances	Class specific noise	Common noise	Class specific noise	Common noise
Free	$[a_{kj}b_kQ_kd_k]$	$[a_{kj}bQd_k]$	$[a_{kj}b_kQ_kd]$	$[a_{kj}bQ_kd]$
Isotropic	$[a_kb_kQ_kd_k]$	$[a_kbQ_kd_k]$	$[a_kb_kQ_kd]$	$[a_kbQ_kd]$
Homosced.	$[ab_kQ_kd_k]$	$[abQ_kd_k]$	$[ab_kQ_kd]$	$[abQ_kd]$

Table 2: Models with class specific orientation matrix.

plying ν_{kj} by $\mathcal{X}_k^t$. Using this computational trick, it has been possible to classify a dataset of 10 classes with 13 observations in each class and described in a 1024-dimensional space. Furthermore, it has been noticed in this case a reduction by a factor 500 of the computing time compared to the classical approach.

3.2. Input options

The main routines `hdda` and `hddc` have the following common options:

- `model`: 14 models can be used in those functions: 12 models with class specific orientation matrix (summarized in Table 2) and two models with common covariance matrix: the models $[a_jbQd]$ and $[abQd]$. The list of all available models is given on Table 1. The most general model is $[a_{kj}b_kQ_kd_k]$, all the parameters are class-specific and each class subspace has as many parameters as its intrinsic dimension, it is the default model.

  Both `hdda` and `hddc` can select, among all possible models, the most appropriate one for the data at hand. The model with the largest BIC value is kept. It is possible to run all 14 models (see Table 1) using the option `model = "ALL"`.

- `d`: This parameter specifies how the choice of the intrinsic dimensions is done:

  - `Cattell`: Cattell's scree-test is used to find the intrinsic dimension of each class. This is the default value of `d`. If the model is with common dimensions, the scree-test is done on the covariance matrix of the whole dataset.
  - `BIC`: The intrinsic dimensions are selected with the BIC criterion. See also Bouveyron *et al.* (2011) for a discussion of this topic.
  - `CV`: For `hdda` only. A V-fold cross-validation (CV) can be done in order to select the best threshold or the best common dimension (for models with common dimension only). The V-fold CV is done for each dimension (respectively threshold) in the argument `cv.dim` (resp. `cv.threshold`), then the dimension (resp. threshold) that gives the best good classification rate is kept. The dataset is split in `cv.vfold` (default is 10) random subsamples, then CV is done for each sample: each of them is used as validation data while the remaining data is used as training data. If `cv.vfold` is equal to the number of observations, then this CV is equivalent to a leave-one-out.

- `cv.dim, cv.threshold and cv.vfold`: These parameters are only used if `d = "CV"`. The first two are vectors that specify the different dimensions (default is $\{1, \ldots, 10\}$) or thresholds (default is $\{0.001, 0.005, 0.01, 0.05, 0.1, 0.2, \ldots, 0.9\}$) for which the cross-validations have to be done. The last one specifies the number of samples used in the V-fold CV, its default value is 10.

- `com_dim`: It is used only for common dimensions models. The user can specify the common dimension he wants. If used, it must be an integer. Its default value is set to `NULL`.
- `threshold`: The threshold used in Cattell's scree-test. The default value is 0.2, which corresponds to 0.2 times the highest difference between two successive eigenvalues.
- `scaling`: Logical: whether to center and standardize the variables of the dataset or not. By default, the data are not scaled (`scaling = FALSE`).
- `graph`: This option is only there for the sake of comparison, when several estimations are run at the same time (either when using several models, or when using cross-validation to select the best dimension/threshold). If `graph = TRUE`, the plot of the results of all estimations is displayed. Default value is `FALSE`.
- `LOO`: For `hdda` only. If `TRUE`, it returns results (classes and posterior probabilities) for leave-one-out cross-validation.

The routine `hddc` has the additional following options:

- `K`: It designates the number of clusters for which the classification has to be done. The algorithm selects the result with the maximum BIC value. Default is `1:10` which means that the clustering is done for one to ten classes and then the solution with the largest BIC value is kept.
- `itermax`: The maximum number of iterations, default is 60.
- `eps`: Defines the threshold value of the stopping criterion. The algorithm stops when the difference between two successive log-likelihoods is below this threshold, default is 10^{-3}.
- `algo`: Three algorithms can be used:
  - `EM`: The default value. The standard EM algorithm is used.
  - `CEM`: Classification EM (Celeux and Govaert 1992) is used to have a faster convergence: at each step, a cluster is allocated to each observation using the *maximum a posteriori* rule.
  - `SEM`: Stochastic EM (Celeux and Diebolt 1985) is used to avoid initialization problems and to try not to stop in a local maximum of the log-likelihood. At each iteration, it allocates each observation to a cluster using a multinomial distribution of probability t_{ik} (the posterior probability that the observation i belongs to the group k).
- `init`: There are five initialization:
  - `kmeans`: The initial class of each observation is provided by the k-means algorithm; it is done using the function `kmeans` with 50 maximum iterations, 4 starts and the default algorithm. This is the default initialization of `hddc`. Note that the user can parametrize `kmeans` using the `...` argument in `hddc`.

  - `param`: This is an initialization of the parameters. It was proposed by McLachlan and Peel (2000), they suggest to set the proportions π_k of the mixture to $1/K$ and generate the means μ_k accordingly to a multivariate Gaussian distribution $\mathcal{N}(m, S)$ where m and S are respectively the empirical mean and the covariance matrix of the whole dataset. The covariance matrices Σ_k are finally initialized to S.
  - `mini-em`: This is the initialization procedure proposed in Biernacki, Celeux, and Govaert (2003). The algorithm is run m times, doing each times nb iterations with a random initialization; the default is 5 times with 10 iterations. Then the result with the highest log-likelihood is kept as the initialization of the algorithm. The parameters m and nb can be set with the `mini.nb` argument.
  - `random`: The group memberships are randomly sampled using a multinomial distribution with equal prior probabilities.
  - *A prior class vector*: The user can also provide his own initialization by giving a vector of group memberships.

- `mini.nb`: This parameter settles the `mini-em` initialization, it is a vector of length 2, containing m and nb, its default value is `c(5, 10)`.

The function `predict.hdc`, which computes the class prediction of a dataset with the parameters previously obtained using either the function `hdda` or `hddc`, may also take another argument:

- `cls`: This argument takes the original class vector of the dataset, it is optional and only for comparison sake (see Section 3.3 for further explanations).

The function `plot.hdc` uses the parameters obtained using `hdda` or `hddc`. It may also take two arguments:

- `method`: The method used to select the intrinsic dimension. It can be `"BIC"` or `"Cattell"`. By default it takes the method used when obtaining the parameters using `hdda` or `hddc`.
- `threshold`: The threshold for Cattell's scree-test. The default is the one used when obtaining the parameters using `hdda` or `hddc`.

3.3. Output

The routines `hdda` and `hddc` have the following common outputs:

- All the estimated model parameters:
  - `a`: The variance parameters within the class-specific subspaces.
  - `b`: The variance parameters outside the class-specific subspaces.
  - `d`: The intrinsic dimensions of the classes.
  - `prop`: The proportions of the classes.
  - `mu`: The means of the classes.
  - `ev`: The eigenvalues of each Σ_k, the covariance matrix of the classes.
  - `Q`: The orthogonal matrices defining the orientation of the classes.

- `scaling`: Contains the mean and the standard deviation of the original dataset, if scaled.
- `BIC`: The BIC value of the estimation.

Also, `hddc` has the following specific outputs:

- `class`: The cluster vector obtained with HDDC.
- `posterior`: The $n \times K$ matrix giving the posterior probability t_{ik} that the observation i belongs to the group k.
- `loglik`: The vector of the log-likelihood at each iteration.

The routine `predict.hdc` gives the following results:

- `class`: The vector of the classification result.
- `posterior`: The $n \times K$ matrix giving the posterior probability t_{ik} that the observation i belongs to the class k.
- If the initial class vector is given to the argument `cls` then:
  - The correct classification rate and this confusion matrix are shown on the R console.
  - `confusion`: The confusion matrix of the classification is given in the output object.

The function `plot.hdc` shows either the graph of Cattell's scree-test or the graph of the dimensions selection using the BIC criterion. Also, a `print` method has been implemented to sum up the main parameters of the model.

4. Practical examples in R

This section aims to illustrate both the use and the main features of the methods HDDA and HDDC through the package **HDclassif**. Two introductory examples, which can be directly run from the package using the command `demo("HDclassif")`, are first presented. The last experiments of this section focus on the numerical advantages of both HDDA and HDDC.

4.1. HDDA: An introductory example

To introduce the supervised classification method HDDA, we first use the `wine` dataset that can be found in the package. This dataset is the result of a chemical analysis of wines from the same region in Italy but derived from $K = 3$ different crops. There are $n = 178$ observations and the $p = 13$ variables are constituents found in each of the three categories of wine. As the variables are from very different nature, some being much larger than others, we choose to center and scale the dataset, which consists to put the mean to 0 and the standard deviation to 1 for each variable, using the option `scaling = TRUE`. The following example can be run using the command `demo("hdda")`.

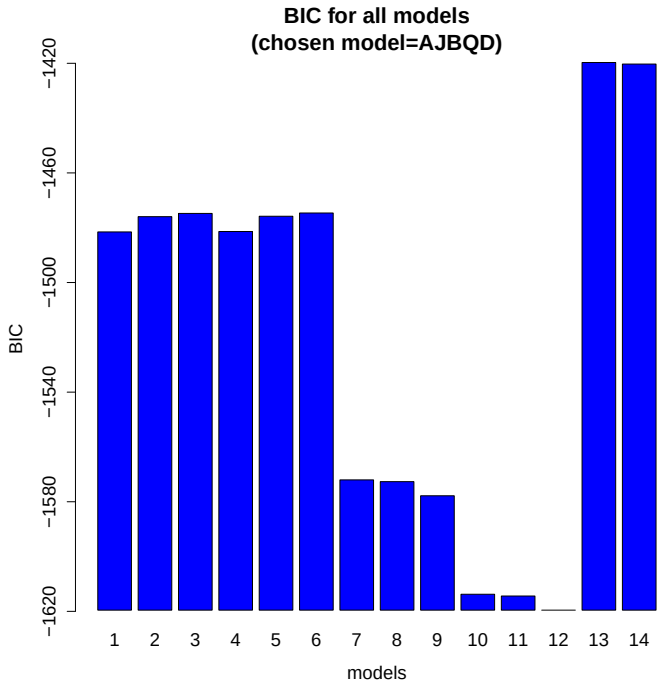

Figure 1: This graph summarizes the results of the estimation of the learning sample of the `wine` dataset. It gives the BIC values for all HDDA models. The match between the numbers and the models is given in the outputs of the subsection *First results* in Section 4.1.

First results

The dataset is split into two different samples. The first one is used to learn the model while the second will be used to test the method performance. The learning dataset is made of 40 randomly selected observations while the test is made of the 138 remaining ones. The parameters are obtained using the function `hdda` on the learning dataset. We use the option `model = "all"` to select the best model with respect to the BIC criterion among all HDDA models. In this case, the intrinsic dimension is selected using the Cattell's scree-test with its default treshold. Notice that since the Cattell's scree-test operates on the empirical covariance matrices, it does not depend on the chosen model. Also, in order to see clearly the BIC differences between the different models, we use the option `graph = TRUE` to plot these results. They are shown in Figure 1. Then, the prediction is done on the testing dataset using the function `predict`. The R code used and the results of the classification are shown below:

```
R> data("wine")
R> w <- wine[, -1]
R> cls <- wine[, 1]
R> set.seed(1)
R> ind <- sample(178, 40)
R> prms <- hdda(w[ind, ], cls[ind], scaling = TRUE, model = "all",
+    graph = TRUE)

 # :          Model               BIC
```

```
 1 :          AKJBKQKDK          -1481.539
 2 :          AKBKQKDK           -1475.969
 3 :          ABKQKDK            -1474.783
 4 :          AKJBQKDK           -1481.384
 5 :          AKBQKDK            -1475.814
 6 :          ABQKDK             -1474.627
 7 :          AKJBKQKD           -1572.024
 8 :          AKBKQKD            -1572.666
 9 :          ABKQKD             -1577.823
10 :          AKJBQKD            -1613.758
11 :          AKBQKD             -1614.4
12 :          ABQKD              -1619.557
13 :          AJBQD              -1419.712
14 :          ABQD               -1420.275

SELECTED: Model AJBQD, BIC=-1419.712.
```

First of all, one can see that for this sample the model which best fits the data with respect to the BIC criterion is one of the most constrained, with a covariance matrix being common for all classes. Remark that another way to select the model would have been to use cross-validation. Let us see the results on the testing dataset:

```
R> res <- predict(prms, w[-ind, ], cls[-ind])

Correct classification rate: 0.9782609.
               Initial class
Predicted class  1  2  3
              1 44  0  0
              2  2 52  0
              3  0  1 39
```

It appears that the method performs well with a correct classification rate of 97%, even with a small learning dataset of 40 individuals. Moreover, the confusion matrix helps to see clearly where are the mismatches.

Intrinsic dimension selection

We now use the full dataset in order to discuss the selection of the intrinsic dimensions, since d_k can be estimated using either Cattell's scree-test or the BIC criterion. HDDA is first used with the BIC criterion to determine the intrinsic dimension of each class-specific subspace, using `d = "BIC"`. Once the parameters are obtained, we use the command `plot` to see the result of the dimension selection.

```
R> prms <- hdda(w, cls, scaling = TRUE, d = "BIC")
R> plot(prms)
```

Figure 2 is the result of the command `plot`, it shows the selection of the intrinsic dimensions of the classes using the BIC criterion. Here, 3 dimensions are selected for the first cluster, 4

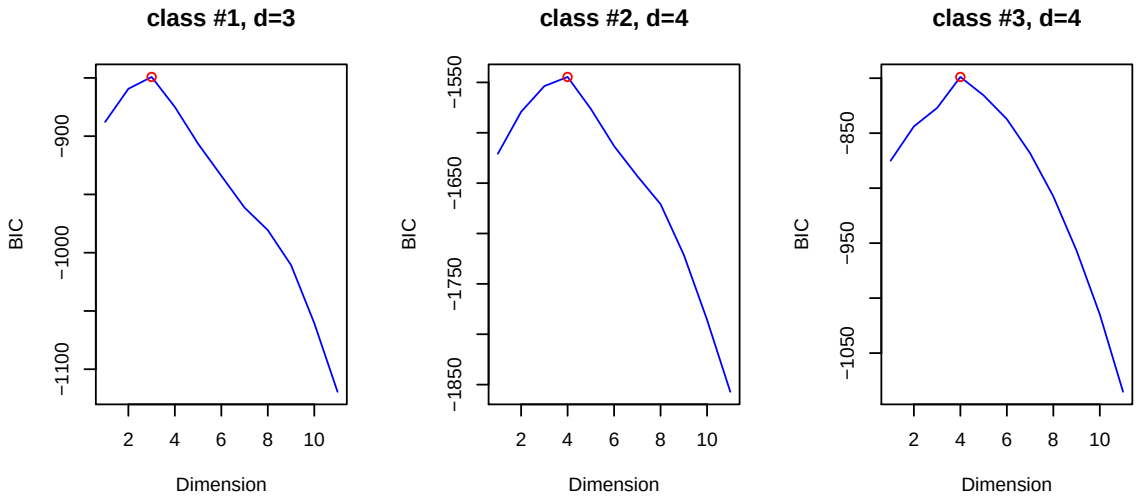

Figure 2: Selection of the intrinsic dimension of the classes in HDDA using the BIC criterion for the `wine` dataset.

for the second and 4 for the third one. Let us recall that the dimensions can also be selected using Cattell's scree-test. The same `plot` command can be used with the option `method = "Cattell"` to see the results of Cattell's scree-test. We use the option `threshold` to first use a threshold of 0.2 and then one of 0.3:

```
R> plot(prms, method = "Cattell", threshold = 0.2)
R> plot(prms, method = "Cattell", threshold = 0.3)
```

Figure 3 shows the results of these two plots. An increase of the scree-test threshold, from 0.2 to 0.3, leads to a selection of fewer intrinsic dimensions. With a higher threshold, the BIC criterion and the scree-test both select the same number of dimensions for each class. However, it is important to recall that the method HDDA always keeps all the dimensions for the modeling and the classification. Indeed, besides the main variance parameters (a_{kj}), there are also the noise variance parameters (b_k) which model the data outside the class-specific subspaces. Therefore, the method is robust to changes on the intrinsic dimensions: a slight change in the intrinsic dimension estimation does not imply a big modification of the classification results. We recommend to use a threshold between 0.1 and 0.3 to select a small number of dimensions (typically less than 10) and a threshold around 0.01 for more dimensions. Notice as well that, when using both intrinsic dimension and model selections, HDDA first selects the dimension, using cattell or BIC, then the dimension is kept to run the different models.

Using common dimensions models

We now use a common dimension model, the model $[a_{kj}b_kQ_kd]$, to illustrate the dimension selection. First we apply a cross-validation on the dataset using the option `d = "CV"`, with `cv.vfold = 178` in order to make a leave-one-out CV (LOO-CV), and for different common dimensions going from 1 to 10 (which is the default value). The graph which allows to compare

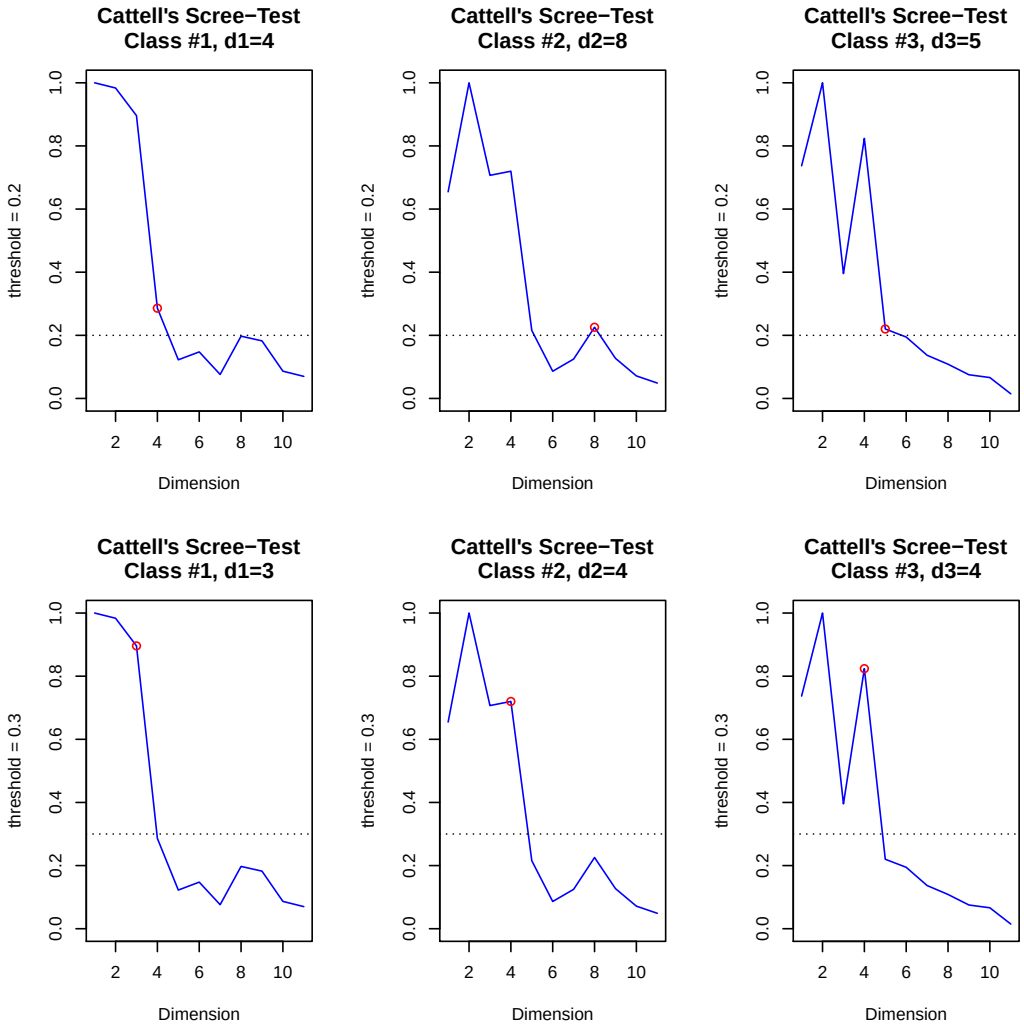

Figure 3: Effect of the Cattell's threshold on the dimension selection for the `wine` dataset with HDDA. These figures are the results of the command `plot`. Those at the top have a threshold of 0.2 while the threshold is of 0.3 for those at the bottom.

the results of the LOO-CV is displayed using `graph = TRUE`. We then compare the results of the CV with the dimensions selected by the two other criteria: BIC and Cattell's scree-test.

```
R> prms <- hdda(w, cls, scaling = TRUE, model = "AkjBkQkD", d = "CV",
+    cv.vfold = 178, graph = TRUE, show = FALSE)
R> plot(prms, method = "BIC")
R> plot(prms, method = "Cattell")
```

The results of dimension selection with the BIC criterion and Cattell's scree-test are displayed on the first two graphs of Figure 4. Both criteria select 5 dimensions. The result of the LOO-CV, displayed on the third graph, validates these results, as the best good classification rate is obtained for 5 dimensions.

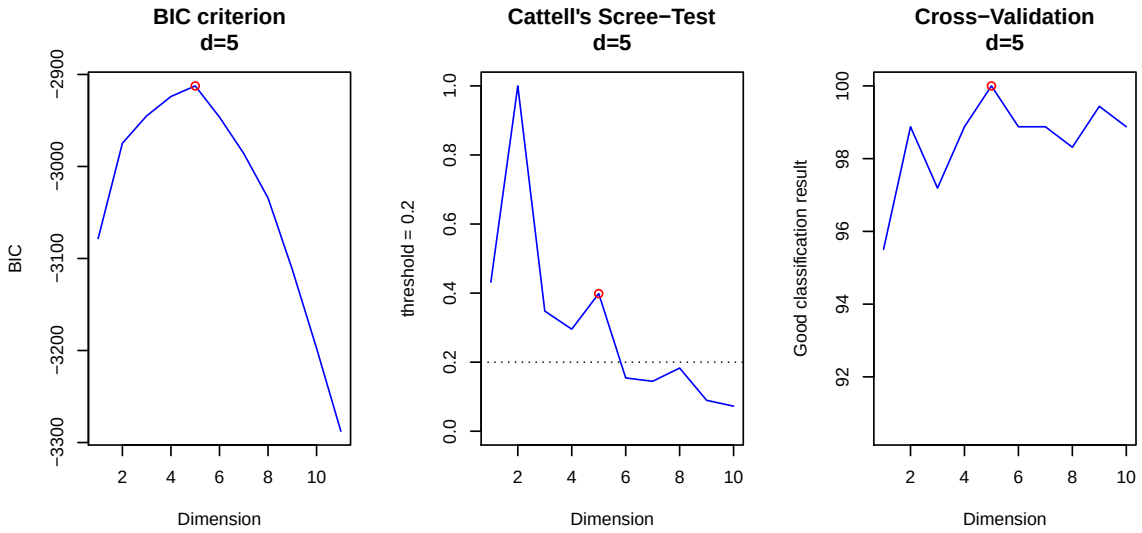

Figure 4: Intrinsic dimension selection with the BIC criterion, Cattell's scree-test and LOO-CV on the `wine` dataset with the common dimensions model $[a_{kj}b_kQ_kd]$ of HDDA.

4.2. HDDC: An introductory example

The clustering method HDDC is now introduced using the `Crabs` dataset. We chose this dataset as a first example of HDDC, as it is known to be hard to cluster. This dataset is made of $p = 5$ measurements on $n = 200$ individuals split in $K = 4$ balanced classes: male and female crabs with orange shell, male and female crabs with blue shell. For each crab, the 5 variables are: the frontal lobe size, the rear width, the carapace length, the carapace width and the body depth. This example can be run directly from the package using the command `demo("hddc")`.

First results

The clustering of this dataset is done with HDDC, all the default settings are kept:

```
R> data("Crabs")
R> A <- Crabs[, -1]
R> cls <- Crabs[, 1]
R> set.seed(1)
R> prms <- hddc(A, 4)

 Model       k       BIC
AKJBKQKDK    4    -2809.081

R> res <- predict(prms, A, cls)

Correct classification rate: 0.945.
               Initial class
Predicted class BF BM OF OM
```

```
4 50  9  0  0
2  0 41  0  0
3  0  0 48  0
1  0  0  2 50
```

The results obtained show a correct classification rate of 94% and the confusion matrix is again helpful to understand the clustering results: while the orange males (OM) are totally well classified, the blue males (BM) seem to have characteristics similar to the blue females (BF). Also, the two species are totally separated, as there is no mismatch between them.

PCA representation

Figure 5 shows the projection of the data on the first and second principal axis, obtained using the principal component analysis (PCA), as well as the PCA representation of the clustering result obtained with HDDC. Furthermore, as the estimated dimension of the intrinsic subspace of each class is equal to 1, this allows an easy representation of HDDC's subspaces using line segments. In order to illustrate the clustering process, we run HDDC with the model $[a_{kj}b_kQ_kd_k]$ using a k-means initialization, then, every 3 steps, we plot the dataset on its 2 first principal axis. The clusters, the means and the orientation of each class are also represented. The results are shown on Figure 6. This example can be run interactively with the command `demo("hddc")` where the user can choose the algorithm, the model and the initialization. It can be observed on Figure 6 that, even with an initialization far from the original classes, HDDC updates sequentially the means and the orientations of the classes to finally reach a classification close to the expected one.

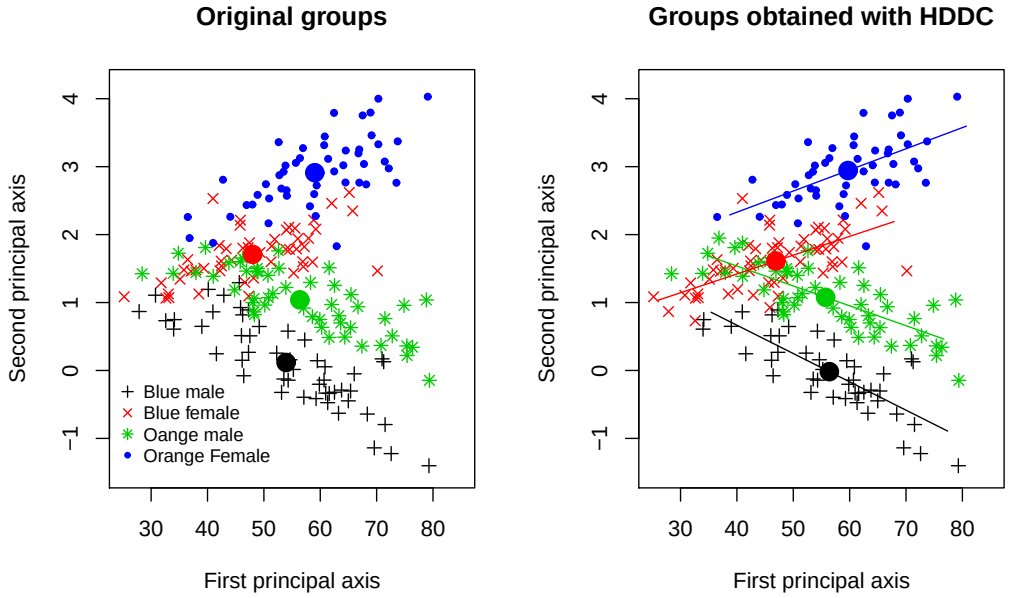

Figure 5: Clustering of the `Crabs` dataset, visualization on the 2 first principal axis. The segments represent the classes subspaces while the points are the means.

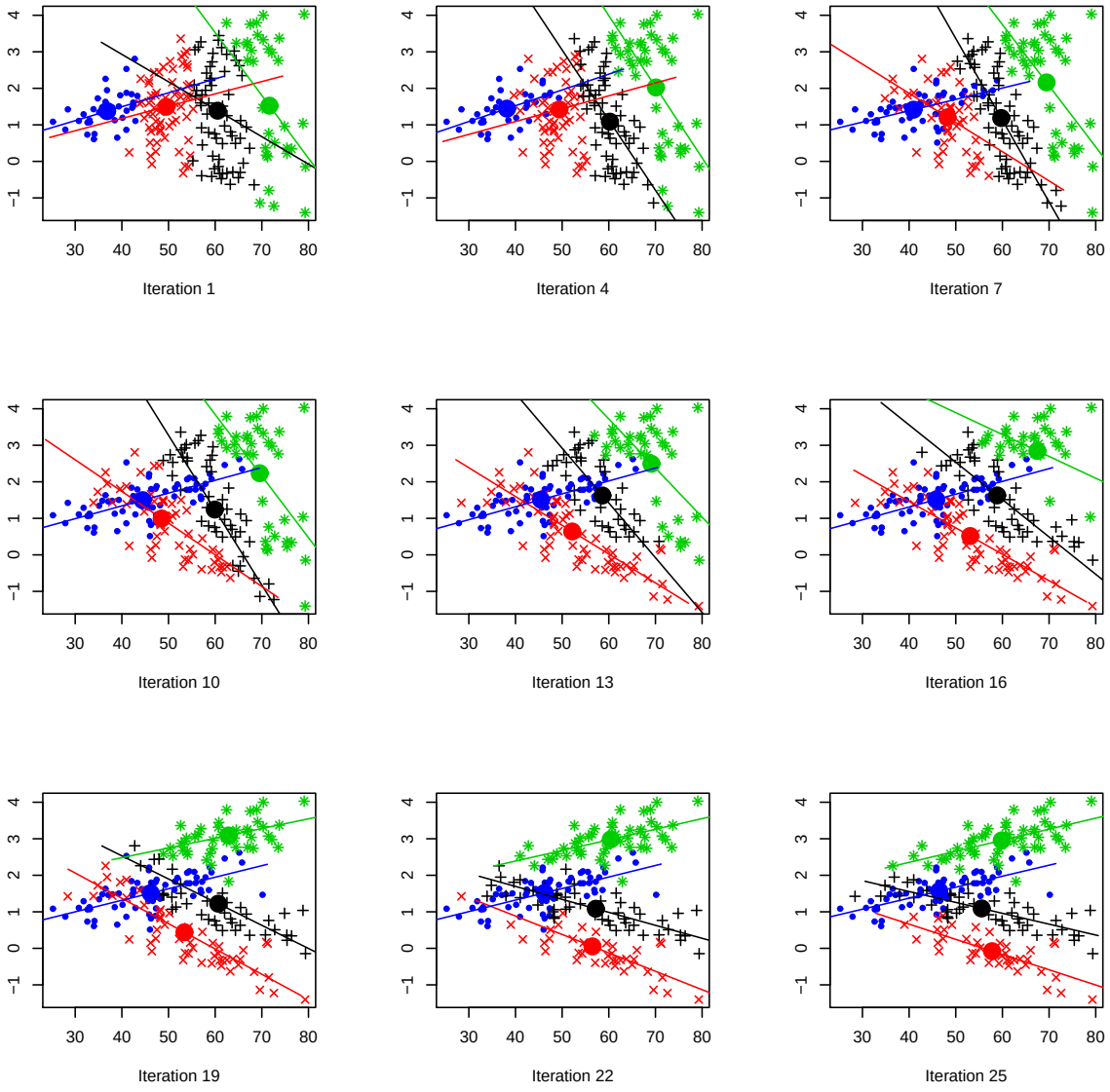

Figure 6: Clustering process of the `Crabs` dataset with HDDC. The initialization is done with k-means.

Cluster selection

Since HDDC is a model-based clustering method, we can also use the BIC criterion to select the number of clusters K to keep. HDDC provides a simple way to do this: it displays the BIC value for each clustering result for different number of classes and select the model which maximizes it. Let us compute the clustering for 1 to 10 classes, which is the default option:

```
R> set.seed(1)
R> prms <- hddc(A)
```

```
 Model       K     BIC
ALL          1   -3513.071
AKJBKQKDK    2   -3299.155
AKJBKQKDK    3   -3138.566
AKJBKQKDK    4   -2809.081
AKJBKQKDK    5   -2842.08
AKJBKQKDK    6   -3099.988
AKJBKQKDK    7   -2949.258
AKJBKQKDK    8   -3007.329
AKJBKQKDK    9   -3045.638
AKJBKQKDK    10  -3089.492

SELECTED: model AKJBKQKDK with 4 clusters, BIC=-2809.081.
```

As discussed by Hennig (2010), the choice of the number of clusters is a complex question; particularly with this dataset where the number of 4 clusters is not obvious. Indeed, some other Gaussian mixture models may select 9 clusters and this partition could be viewed as a subpartition of the "true" classes. However, one can see here that with the model $[a_{kj}b_kQ_kd_k]$ the BIC criterion points out the number of clusters that was originally defined by the construction of this dataset, which is 4.

4.3. HDDA: The effect of the data dimension

We now experiment the effect of the dimensionality on different supervised classification methods based on the Gaussian mixture model. To this end, we simulate three classes modeled by Gaussian densities on $\mathbb{R}^p$, $p = 20, \ldots, 200$, with respect to the model $[a_kb_kQ_kd_k]$. The following parameters were used: $\{d_1, d_{2,}, d_3\} = \{2, 5, 10\}$, $\{\pi_1, \pi_2, \pi_3\} = \{0.4, 0.3, 0.3\}$, $\{a_1, a_2, a_3\} = \{150, 75, 50\}$ and $\{b_1, b_2, b_3\} = \{15, 10, 5\}$; with close means: $\{\mu_1, \mu_2, \mu_3\} = \{(0, \ldots, 0), (10, 0, \ldots, 0), (0, \ldots, 0, -10)\}$. Each orientation matrix Q_k was simulated as the orthogonal matrix of a QR factorization of a random multivariate normal distribution. The learning and testing datasets were respectively made of 250 and 1000 points. The performance of each method was measured by the average of the correct classification rates on the test dataset for 50 replications on different samples of the simulated learning and testing datasets. The model $[a_kb_kQ_kd_k]$ is used in HDDA and is compared to three other methods: QDA, linear discriminant analysis (LDA) and PCA+LDA (LDA on 15-dimensional data projected with PCA). The results of each method are represented as boxplots in Figure 7.

Unsurprisingly, the QDA method shows its weakness with high dimensionality and its performance sinks when the dimension rises. Moreover, when the dimension reached 50, QDA began to fail because of singularity problems on the covariance matrices. In particular, QDA failed half of the time in 70 dimensions and then did not work anymore with dimensions higher than 80. The LDA method is less sensitive to the dimension than QDA, but its performance declines when the dimension gets beyond 60. The method PCA+LDA improves LDA results and seems only little affected by the dimension but it cannot reach more than 82% of average correct classification rate. Finally, as expected, HDDA appears not to be sensible to large dimension as it provides good results in large as well as in low dimension. Furthermore, Figure 7 clearly shows that the results of HDDA have a low variance in comparison to the other methods and the rise of the dimension increases only slightly the variance of its results.

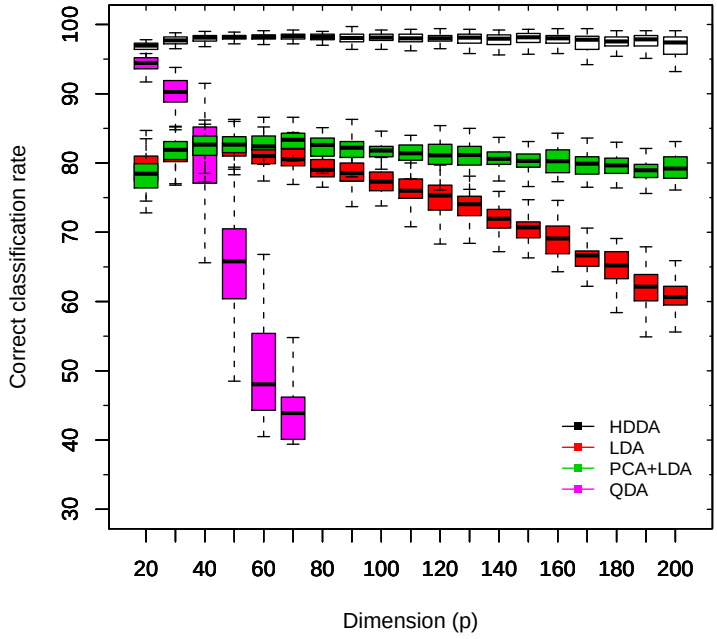

Figure 7: Boxplots of the supervised classification results for different methods on a simulated dataset. This has been done with 250 observations for learning and 1000 observations for testing, with an increasing dimensionality.

4.4. HDDC: The effect of sample size

We study here the ability of HDDC models to deal with high-dimensional datasets of small sizes. For this, three Gaussian densities in $\mathbb{R}^{60}$ are simulated in the same way as in the previous experiment. In order to investigate the effect of the sample size on clustering results in high-dimensional spaces, we try to cluster the data for different dataset sizes as this phenomenon occurs when the number of observations n is small compared to the dimension p. The number of chosen observations varies from a small value ($n = 100$) to a high value ($n = 4000$) compared to p.

Here we used HDDC with the model $[a_k b_k Q_k d_k]$ and with the `mini-em` initialization which is a popular estimation procedure. HDDC is also compared to three other clustering methods based on the Gaussian mixture model which can be found in the R package **mclust** (Fraley and Raftery 1999). The models used are (from the most complex to the simplest one): ellipsoidal, varying volume, shape and orientation (VVV), diagonal, varying volume and shape (VVI), diagonal, equal volume and shape (EEI). For each method and each number of observations, the experiment is repeated 20 times, each time for a different simulated dataset but with the same parameters.

The results of this experiment are presented in Figure 8. This figure combines the boxplots of each method, the mean of the correct classification results when the algorithm converged (red curves) and the number of times the algorithm failed for numerical reasons (black curves).

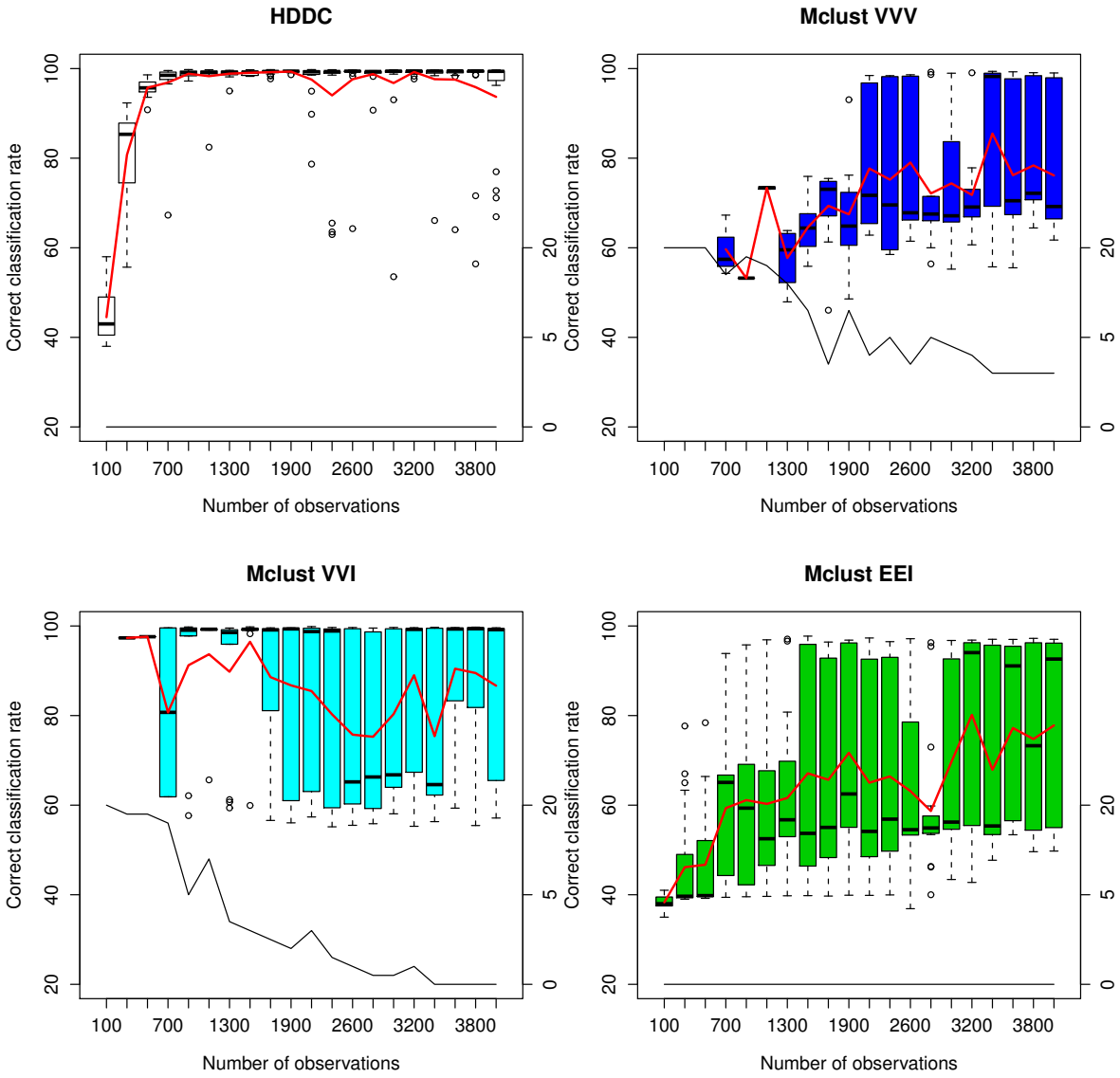

Figure 8: Effect of the dimension on correct classification rates on simulated data with the functions `hddc` (model $[a_k b_k Q_k d_k]$) and `Mclust` (models VVV, VVI and EEI). The red lines represent the means of the correct classification rates (when the algorithm converged) while the black lines represent the numbers of times that the algorithms could not be fitted in the 20 simulations (its scale is at right).

It appears that all tested models of the function `Mclust` are very sensitive to both the high dimension of the data and the size of the dataset since their clustering results have a high variance whatever the value of n. In particular, `Mclust` with its most complex model (VVV) is unsurprisingly very sensitive to the size of the dataset since it is highly over-parametrized in high-dimensional spaces. Indeed, models with non constrained variances require the estimation of $O(p^2)$ parameters which is, in some cases here, dramatically larger than the number of observations ($p = 60$). As one can see, `Mclust` with the VVV model often fails for numerical

reasons and does not work at all for datasets smaller than 500 observations. The model VVI, which is a more parsimonious model than VVV, seems well appropriate for this kind of data but presents a high variance of its results and often fails for numerical reasons. It is also unable to cluster datasets of sizes smaller than 300 observations. The model EEI is, conversely to the two previous ones, a very parsimonious model. Using this model within `Mclust` allows the algorithm to always provide a result. These results are however very sensitive to the dataset size and have a large variance when n is larger than 1500. As expected, since the model used for the simulation is one of the HDDC models, HDDC works well for a large range of dataset sizes. It is however interesting to notice that HDDC is very stable (small variance of the results) and that its robustness weakens only for datasets smaller than 300. In such cases, it would be preferable to use a more parsimonious model of HDDC (the model $[abQ_kd_k]$ for instance). Notice that a similar behavior is expected for the models of McNicholas and Murphy (2008a); Baek *et al.* (2009) and Bouveyron and Brunet (2012).

4.5. HDDC: Comparison with variable selection

We focus now on the comparison of HDDC with variable selection methods for clustering. Recently, Raftery and Dean (2006) proposed a Bayesian approach for variable selection in the model-based clustering context. Their approach recasts the variable selection problem into a model selection problem and the BIC criterion is used to decide whether a variable should be retained or not. The whole procedure is embedded in a backward-forward algorithm to explore combinations of variables. The package **clustvarsel** implements this method. For this experiment, we used the `Crabs` (also used in Raftery and Dean 2006) and the `USPS358` datasets and we compared on both datasets the clustering performances of HDDC, `clustvarsel` and `Mclust`. The results of `Mclust` are reported as reference results. The `USPS358` dataset is a subset of the original USPS dataset which will be used and described in detail in Section 5.1. The `USPS358` dataset contains only the 1756 images associated to the digits 3, 5 and 8 and each image is represented as 256-dimensional vector.

Table 3 and 4 present, for each studied method, the correct classification rate (CCR, 2nd

Function	CCR	Used model	Computing time
`Mclust`	0.575	VEV on 5 var.	0.2 sec.
`clustvarsel`	0.925	EEV on 4 var.	4.4 sec.
`hddc`	**0.945**	$[a_{kj}b_kQ_kd]$ with $d = 1$	1.8 sec.

Table 3: Comparison between HDDC and variable selection (package **clustvarsel**) on the `Crabs` dataset ($n = 200$, $p = 5$, $K = 4$). The results of `Mclust` are reported as reference results.

Function	CCR	Used model	Computing time
`Mclust`	0.555	EEE on 256 var.	1 028.7 sec.
`clustvarsel`	0.483	EEV on 6 var.	6 922.5 sec.
`hddc`	**0.930**	$[a_{kj}b_kQ_kd_k]$ with $d = \{7, 7, 6\}$	301.2 sec.

Table 4: Comparison between HDDC and variable selection (package **clustvarsel**) on the `USPS358` dataset ($n = 1756$, $p = 256$, $K = 3$). The results of `Mclust` are reported as reference results.

column) which measures the adequation of the proposed partition with the known one, the model chosen by the procedure (3rd column) as well as the computing time (4th column, in seconds) for the whole procedure (model selection and clustering). On the one hand, regarding the `Crabs` dataset, one can notice that this apparently simple dataset (ratio n/p large) is difficult to cluster with traditional Gaussian mixture models since `Mclust` failed to propose a good partition of the data. The selection of variables made by `clustvarsel` allows to clearly improve the clustering performances of `Mclust` which reaches 92.5% of adequacy with the known labels by selecting 3 among the 5 original variables. HDDC selects using BIC the model $[a_{kj}b_kQ_kd]$ with $d = 1$ and provides a partition of the data which has an adequacy rate with the known labels of 94%. On the other hand, the high-dimensional `USPS358` dataset seems to be as well a difficult dataset since `Mclust` again failed to propose a good partition of the data. However, the selection of variables made by `clustvarsel` in this case seems to be non discriminative since it deteriorates the clustering results compared to `Mclust`. HDDC selects the model $[a_{kj}b_kQ_kd_k]$ with $\{d_1, d_2, d_3\} = \{7, 7, 6\}$ and reaches 93% of clustering accuracy. The results of this experiment suggest that it is preferable to keep all variables in the model, with nevertheless different roles, than discarding some of them. Finally, it is also important to notice that HDDC is significantly less time consuming than `clustvarsel`, and particularly in high dimensions.

4.6. HDDC: computing time comparison

Here the effect of the dimension and of the number of observations on the computing time is finally tested. In order to realize this experiment, a dataset has been simulated according to the same protocol as before with different dimensions, $p = 50, \ldots, 200$, and number of observations, $n = 200, \ldots, 1000$. HDDC is again compared here to the **mclust** package. The experiment has been run on a laptop PC with a 2.10 GHz Intel core 2 duo T4300 processor and 4 GB of RAM. Both methods are used with their default parameters and the presented results are the average times on 20 replications.

The results, given in Table 5, show how the computing time rises with the dimension and the number of observations. It clearly appears that the function `hddc` is faster than `Mclust` for high-dimensional datasets. It is also interesting to remark that the impact of the rise of dimension or of the number of observations is much less important on HDDC than on

Function	Dimensions	Number of observations				
		200	400	600	800	1000
`hddc`	50	0.16	0.40	0.52	0.67	0.86
	100	0.52	0.91	1.15	1.22	1.34
	150	0.61	1.40	2.11	2.35	2.72
	200	0.69	1.79	3.70	3.82	4.48
`Mclust`	50	0.50	2.06	4.21	8.30	11.29
	100	1.19	4.92	10.59	17.80	27.37
	150	2.02	9.56	20.92	35.76	54.20
	200	2.88	14.36	34.19	62.05	96.10

Table 5: Average computation times of the functions `hddc` and `Mclust` on simulated datasets with varying dimensions and observation numbers (in seconds).

`Mclust`. In particular, `Mclust` is 20 times slower than HDDC on a 200-dimensional dataset with 1000 observations whereas it is only 3 times slower on a 50-dimensional dataset with 200 observations.

5. Applications

The methods HDDA and HDDC are now applied on two real-world datasets that have in common to be in high dimension. The first one contains images represented as 256-dimensional observations whereas the second one is made of spectra with more than 6,000 dimensions.

5.1. Optical character recognition

HDDA is first tested on the optical character recognition (OCR) dataset used for the study of the United States postal service (USPS), which consists in the recognition of handwritten numbers as shown in Figure 9 and which can be found on the site of the University of Aachen at http://www-i6.informatik.rwth-aachen.de/~keysers/usps.html. There are 7,291 images for learning and 2,007 images for testing. The data is divided in 10 classes, each digit is a 16×16 gray level image represented as a 256-dimensional vector. In this experiment, four supervised classification methods are compared: HDDA, LDA, PCA+LDA and the support vector machines (SVM) with the radial basis function (RBF) kernel. The aim of this experiment is to see the effect of the size of the learning dataset on the prediction results. For this, HDDA is computed with the model $[a_{kj}bQ_kd]$ and with the threshold of Cattell's scree-test fixed at 0.05. Indeed a common noise is particularly efficient for this dataset and this low threshold leads to keep an average of 15 dimensions which seems parsimonious enough (compared to the 256 dimensions) and high enough to provide good classification results. The performance of the methods is measured by the average correct classification rate computed on 50 replications, for different sizes of the learning dataset, $n = 100, \ldots, 2000$. Figure 10 shows the results of the experiment and highlights that HDDA works very well compared to the other methods when the size of the learning dataset is small. One can see that a PCA step improves the prediction results of LDA and allows this method to work with small learning dataset. This experiment illustrates that HDDA provides very satisfying results in high-dimensional space and with small learning datasets. Table 6 shows in addition the computation time of the four methods on the whole training and testing datasets. The presented results are the average times on 20 replications. It appears that HDDA is again faster compared to the other methods due to its parsimonious model. The computing time of HDDA with the model $[a_jbQd]$ has been also added to Table 6 in order to show that a linear method with only one covariance matrix to estimate can again faster the computation.

Figure 9: Some examples of the `USPS` dataset used for the OCR experiment.

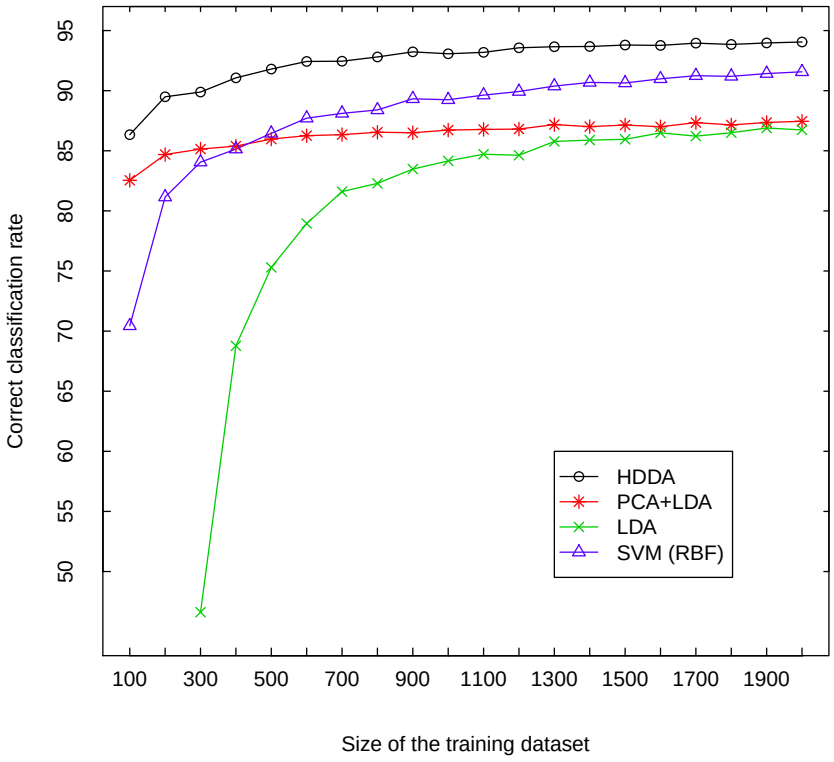

Figure 10: Influence of the size of the learning dataset on prediction results obtained with HDDA and other classification methods on the `USPS` dataset.

Method	$[a_{kj}b_kQ_kd_k]$	$[a_jbQd]$	LDA	PCA+LDA	SVM
Time	2.00	0.88	13.23	3.34	97.29

Table 6: Comparison of computing times (in seconds) of training and predicting on the `USPS` learning and testing datasets.

5.2. Maldi mass-spectrometry

In this last experimental section, the two methods HDDA and HDDC are applied to the problem of cancer detection using Maldi mass spectrometry. Maldi mass spectrometry is a non invasive biochemical technique which is useful in searching for disease bio-markers, assessing tumor progression or evaluating the efficiency of drug treatment, to name just a few applications. In particular, a promising field of application is the early detection of the colorectal cancer, which is one of the principal causes of cancer-related mortality, and Maldi imaging could in few years avoid in some cases the colonoscopy method which is invasive and quite expensive. The `Maldi2009` dataset has been provided by Theodore Alexandrov from the Center for Industrial Mathematics (University of Bremen, Germany) and is made of 112 spectra of length 16,331. Figure 11 shows the mean spectra of the cancer and control (healthy people) classes on the mass-to-charge (m/z) interval 900–3500 Da. Among the 112 spectra,

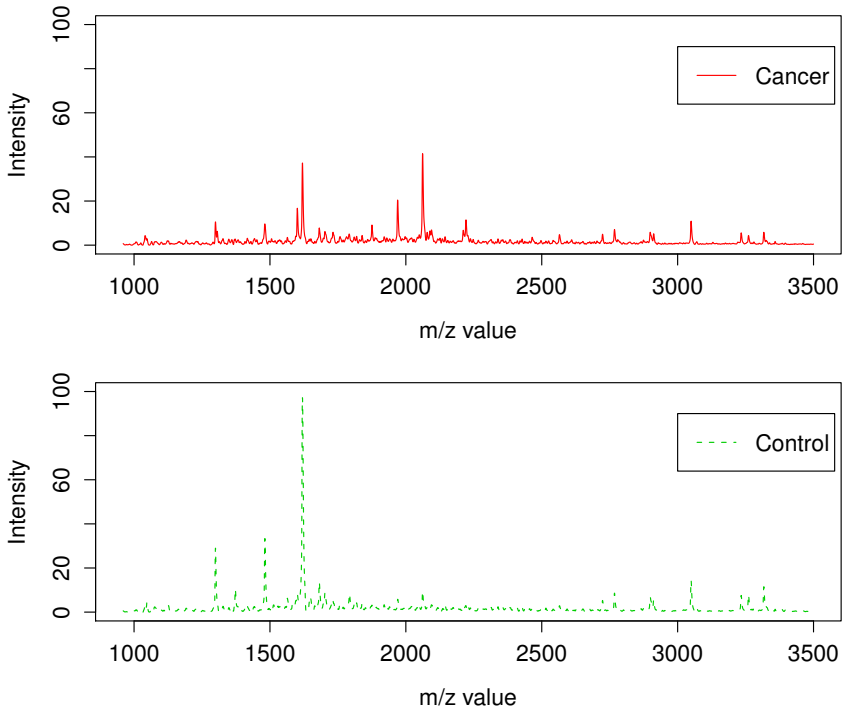

Figure 11: Mean spectra of the cancer class (up) and of the control class (bottom) on the m/z interval 900–3500 Da.

Method	CCR	Computing time
$[a_{kj}b_kQ_kd_k]$	0.955	26.37
$[a_jbQd]$	0.973	32.52
LDA	0.524	24.01
SVM	1.000	457.36

Table 7: Correct classification rates of LOO-CV on the `Maldi2009` dataset, and computing times (in seconds).

64 are spectra from patients with the colorectal cancer (referred to as cancer hereafter) and 48 are spectra from healthy persons (referred to as control). Each of the 112 spectra is a high-dimensional vector of 16,331 dimensions which covers the m/z ratios from 960 to 11,163 Da. Following the experimental protocol of Alexandrov, Decker, Mertens, Deelder, Tollenaar, Maass, and Thiele (2009), only 6,168 dimensions corresponding to m/z ratios between 960 and 3,500 Da are used since there is no discriminative information on the reminder.

Supervised classification of the spectra

Here HDDA is tested in terms of effectiveness and computation time. The method is used with two different models: the less constrained model, $[a_{kj}b_kQ_kd_k]$, and the most constrained one, $[a_jbQd]$, each time the intrinsic dimensions are selected thanks to the BIC criterion. Then it is compared to LDA and SVM with a RBF kernel. A LOO-CV is done for each classification method; we used the option `LOO = TRUE` to do the LOO-CV with HDDA. The results are shown on Table 7. First is to notice that LDA is totally inefficient on this dataset

Method	Class	Cluster	
		Cancer	Control
PCA-EM	Cancer	48	16
	Control	1	47
	Misclassification rate = 0.15		
Mixt-PPCA	Cancer	62	2
	Control	10	38
	Misclassification rate = 0.11		
HDDC	Cancer	62	6
	Control	0	45
	Misclassification rate = 0.05		

Table 8: Confusion matrices of the three studied clustering methods on the `Maldi2009` dataset.

that has a large number of parameters. SVM works very well with no misclassification but with a very high computation time. HDDA gives satisfying results, upper than 95% of good classification for the two models, and has fast computation time as the LOO-CV for HDDA is more than 17 times faster than SVM. So HDDA combines a computing time close to LDA with performances close to SVM.

Unsupervised classification of the spectra

HDDC is now applied on this dataset to test its effectiveness on very high dimensional datasets (with $n \ll p$). For the sake of comparison, `Mclust` with the VVV model on principal components (PCA-EM) and mixture of probabilistic principal component analysis (Mixt-PPCA, Tipping and Bishop 1999) have been applied to this subset as well. It has been asked to all methods to cluster the dataset into 2 groups. HDDC is set with the most unconstrained model $[a_{kj}b_kQ_kd_k]$ and with a scree-test threshold of 0.1. The results are shown in Table 8. All the methods present good results for such a complex problem, although the best level of classification has been obtained with HDDC with a misclassification rate of 5%.

6. Conclusion

This paper has presented the R package **HDclassif** which is devoted to the clustering and the discriminant analysis of high-dimensional data. The package provides the classification functions HDDA and HDDC associated to a new Gaussian mixture model first proposed by Bouveyron *et al.* (2007b) which takes into account that high-dimensional data live in low-dimensional subspaces. The proposed models are more parsimonious than other Gaussian mixture models available in other R packages. After having presented the theoretical aspects of the methods and illustrated their use within the package **HDclassif**, this paper has shown the efficiency of both methods through comparisons with reference methods on simulated and real datasets.

Among the possible extensions of this work, it would be first interesting to allow HDDA and HDDC to deal with semi-supervised problems (mixture of labeled and unlabeled data). This extension is planned for the next release of the package. Another interesting extension would

be to add a ℓ_1 penalty on the loading matrices Q_k within HDDA and HDDC to be able to select the original variables which are the most useful for the considered task. This would allow the practitioner to better understand the classification results provided by the method. Finally, it would be also interesting to gather in a unique package all discriminant analysis and clustering techniques based on the factor analysis model. This would include the methods based on the models of McNicholas and Murphy (2008a), Baek *et al.* (2009) and Bouveyron and Brunet (2012).

Acknowledgments

We would like to thank the three reviewers and the editor whose comments and suggestions greatly helped in improving the content and the presentation of this paper.

References

Alexandrov T, Decker J, Mertens B, Deelder AM, Tollenaar RA, Maass P, Thiele H (2009). "Biomarker Discovery in MALDI-TOF Serum Protein Profiles Using Discrete Wavelet Transformation." *Bioinformatics*, **25**(5), 643–649.

Baek J, McLachlan G, Flack L (2009). "Mixtures of Factor Analyzers with Common Factor Loadings: Applications to the Clustering and Visualisation of High-Dimensional Data." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **32**(7), 1298–1309.

Bellman R (1957). *Dynamic Programming*. Princeton University Press.

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 & Data Analysis*, **41**, 561–575.

Bock HH (1996). "Probabilistic Models in Cluster Analysis." *Computational Statistics & Data Analysis*, **23**(1), 5–28.

Bouveyron C, Brunet C (2012). "Simultaneous Model-Based Clustering and Visualization in the Fisher Discriminative Subspace." *Statistics and Computing*, **22**(1), 301–324.

Bouveyron C, Celeux G, Girard S (2011). "Intrinsic Dimension Estimation by Maximum Likelihood in Probabilistic PCA." *Pattern Recognition Letters*, **32**(14), 1706–1713.

Bouveyron C, Girard S, Schmid C (2007a). "High Dimensional Data Clustering." *Computational Statistics & Data Analysis*, **52**, 502–519.

Bouveyron C, Girard S, Schmid C (2007b). "High-Dimensional Discriminant Analysis." *Communications in Statistics: Theory and Methods*, **36**(14), 2607–2623.

Cattell R (1966). "The Scree Test for the Number of Factors." *Multivariate Behavioral Research*, **1**(2), 245–276.

Celeux G, Diebolt J (1985). "The SEM Algorithm: A Probabilistic Teacher Algorithm from the EM Algorithm for the Mixture Problem." *Computational Statistics Quaterly*, **2**(1), 73–92.

Celeux G, Govaert G (1992). "A Classification EM algorithm for Clustering and Two Stochastic Versions." *Computational Statistics & Data Analysis*, **14**, 315–332.

Dempster A, Laird N, Rubin D (1977). "Maximum Likelihood from Incomplete Data via the EM Algorithm." *Journal of the Royal Statistical Society B*, **39**(1), 1–38.

Fraley C, Raftery A (1999). "**mclust**: Software for Model-Based Cluster Analysis." *Journal of Classification*, **16**, 297–306.

Hennig C (2010). "Methods for Merging Gaussian Mixture Components." *Advances in Data Analysis and Classification*, **4**(1), 3–34.

McLachlan G (1992). *Discriminant Analysis and Statistical Pattern Recognition*. John Wiley & Sons, New York.

McLachlan G, Peel D (2000). *Finite Mixture Models*. John Wiley & Sons, New York.

McLachlan G, Peel D, Bean R (2003). "Modelling High-Dimensional Data by Mixtures of Factor Analyzers." *Computational Statistics & Data Analysis*, **41**, 379–388.

McNicholas P, Murphy B (2008a). "Model-Based Clustering of Longitudinal Data." *The Canadian Journal of Statistics*, **38**(1), 153–168.

McNicholas P, Murphy B (2008b). "Parsimonious Gaussian Mixture Models." *Statistics and Computing*, **18**(3), 285–296.

Pavlenko T (2003). "On Feature Selection, Curse of Dimensionality and Error Probability in Discriminant Analysis." *Journal of Statistical Planning and Inference*, **115**, 565–584.

Pavlenko T, Rosen DV (2001). "Effect of Dimensionality on Discrimination." *Statistics*, **35**(3), 191–213.

Raftery A, Dean N (2006). "Variable Selection for Model-Based Clustering." *Journal of the American Statistical Association*, **101**(473), 168–178.

R Development Core Team (2011). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org/.

Schwarz G (1978). "Estimating the Dimension of a Model." *The Annals of Statistics*, **6**, 461–464.

Scott D, Thompson J (1983). "Probability Density Estimation in Higher Dimensions." In *Fifteenth Symposium in the Interface*, pp. 173–179.

Tipping M, Bishop C (1999). "Mixtures of Probabilistic Principal Component Analysers." *Neural Computation*, **11**(2), 443–482.

Affiliation:

Laurent Bergé
Laboratoire GREThA – UMR CNRS 5113
Université Montesquieu – Bordeaux IV
Avenue Léon Duguit
33608 Pessac cedex, France
E-mail: laurent.berge@u-bordeaux4.fr

Charles Bouveyron
Laboratoire SAMM, EA 4543
Université Paris 1 Panthéon-Sorbonne
90 rue de Tolbiac
75013 Paris, France
E-mail: charles.bouveyron@univ-paris1.fr

Stéphane Girard
Team Mistis, INRIA Rhône-Alpes & LJK
655 avenue de l'Europe, Montbonnot
38330 Saint-Ismier, France
E-mail: stephane.girard@inrialpes.fr

Journal of Statistical Software http://www.jstatsoft.org/
published by the American Statistical Association http://www.amstat.org/
Volume 46, Issue 6 *Submitted:* 2010-12-02
January 2012 *Accepted:* 2011-11-19