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

Laurent Bergé, Charles Bouveyron, Stéphane Girard

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

L. Bergé[1], C. Bouveyron[1] & S. Girard[2]

[1] Laboratoire SAMM, EA 4543, Université Paris 1 Panthéon-Sorbonne

[2] Team Mistis, INRIA Rhône-Alpes & LJK

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 has been called High Dimensional Discriminant Analysis (HDDA). In a similar manner, the associated clustering method has been called High Dimensional Data Clustering (HDDC) and uses the Expectation-Maximization (EM) 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-dimensional spaces. Experiments on artificial and real datasets show that HDDC and HDDA perform better than existing classical methods on high-dimensional datasets, even with small datasets. **HDclassif** is a free software and distributed under the GNU 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 [3] for a review on this topic. Popular classification methods are based on the Gaussian mixture model [13] 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 *curse of dimensionality* and was introduced by Bellman [2]. We refer to [15, 16] 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, [5, 6] 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 reparametrization in discriminant analysis has yield a new method called High Dimensional Discriminant Analysis (HDDA) [6] and the associated clustering method has been named High Dimensional Data Clustering (HDDC) [5].

The *R* package **HDclassif** (currently in version 1.1) 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 simulated 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 version 1.1 of 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 [12] 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 $\{x_1, ..., x_n\}$ 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) which unfortunately requires the estimation of a very large number of parameters (proportional to p^2) and therefore faces to numerical problems in high-dimensional spaces. Hopefully, due to the *empty space* phenomenon [18], it can be assumed that high-dimensional data live around subspaces with a dimension lower than p. Recently, [5, 6] 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 [12], 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 $k = 1, ..., K$. Let Q_k be the orthogonal matrix with the eigenvectors of Σ_k as columns. The class conditional covariance matrix Δ_k is therefore defined in the eigenspace of Σ_k by:

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

The matrix Δ_k is thus a diagonal matrix which contains the eigenvalues of Σ_k. It is further assumed that Δ_k can be divided into two blocks:

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

with $a_{kj} > b_k$, $j = 1, ..., d_k$, 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 variance of the actual data of the kth group is therefore modeled by $a_{k1}, ..., a_{kd_k}$ and the variance of the noise is modeled by b_k. The dimension d_k can be considered as well as the intrinsic dimension of the latent subspace of the kth group. Let us remark that if we constrain d_k to be equal to $(p-1)$ for all $k = 1, ..., K$, 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 to the model proposed in [19] in the unsupervised classification framework. As a consequence, our approach encompasses the mixture of probabilistic principal component analyzers introduced in [19] and extended in [14]. 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 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. 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 is natural when the data are obtained in a common acquisition process. Among the 28 models proposed in the original articles [5, 6], 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

Model	Number of parameters	Asymptotic order	Nb of prms $K=4$, $d=10$, $p=100$
$[a_{kj}b_kQ_kd_k]$	$\rho+\bar{\tau}+2K+D$	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: $\rho=Kp+K-1$ 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.

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 $\Sigma_k=\mathrm{diag}(s_{k1}^2,...,s_{kp}^2)$ with $s_k^2\in\mathbb{R}_+^p$ and Sphe-GMM refers to the Gaussian mixture model for which $\Sigma_k=s_k^2I_p$ with $s_k^2\in\mathbb{R}_+$.

2.2 High dimensional discriminant analysis

The use of the models presented in the previous paragraph gave birth to a method for high-dimensional discriminant analysis called HDDA [6]. 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 [6]. 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 $k=1,...,K$, $\mathbb{P}(Z=k|X=x)=1\Big/\sum_{\ell=1}^{K}\exp\left(\frac{1}{2}(\Gamma_k(x)-\Gamma_\ell(x))\right)$ 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 $[a_k b_k Q_k d_k]$:

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

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 have given birth to a model-based clustering method called HDDC [6]. 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 EM algorithm [10] 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 formula for model parameters can be found in [6].

The EM algorithm stops when the likelihood has reached a local maximum.

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 on 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_k and K. Indeed, the parameters d_k and K can not be determined by maximizing the likelihood since they both control the model complexity. The estimation of the intrinsic dimensions d_k is a difficult problem with no unique technique to use. In [6], 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

	Free dimensions		Common dimensions	
	Class specific noise	Common noise	Class specific noise	Common noise
Free class parameters	$[a_{kj}b_kQ_kd_k]$	$[a_{kj}bQd_k]$	$[a_{kj}b_kQ_kd]$	$[a_{kj}bQ_kd]$
One parameter per class	$[a_kb_kQ_kd_k]$	$[a_kbQ_kd_k]$	$[a_kb_kQ_kd]$	$[a_kbQ_kd]$
One common parameter	$[ab_kQ_kd_k]$	$[abQ_kd_k]$	$[ab_kQ_kd]$	$[abQ_kd]$

Table 2: Models with class specific orientation matrix.

specific dimension d_k, $k = 1, ..., K$ is estimated through the scree-test of Cattell [7] which looks for a break in the eigenvalues scree. The selected dimension is the one for which the subsequent eigenvalues differences are smaller than a threshold. The threshold can be provided by the user or selected through cross-validation or BIC [17]. In the clustering case, the number of clusters K may have to be estimated as well and and can be selected thanks to the BIC criterion. In the specific case of the models $[a_kb_kQ_kd_k]$, $[a_kb_kQ_kd]$, $[abQ_kd_k]$ and $[abQ_kd]$, it has been recently proved [4] that the maximum likelihood estimate of d_k is asymptotically consistent.

Case $n \ll p$

Furthermore, in the special case where the number of observations n 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. It is faster and more numerically stable to calculate, when $n_k < p$, the eigenvalues and eigenvectors of the 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 multiplying 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 described in a 1024-dimensional space for each class. 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 that are summarized on Table 2 and two models with common covariance matrix: the models $[a_jbQd]$ and $[abQd]$. The most general model is $[a_{kj}b_kQ_kd_k]$, all the parameters are class specific and each class subspace have as many parameters as its intrinsic dimension, it is the default model. For *hdda*, use *model* = "*best*" to run the training on all free dimensions models or *model* = "*dbest*" to run it for all the common dimensions models. The model with the largest BIC value is kept.
- ***graph***: specifies whether or not a graph should be displayed: for *hdda* the graph of the eigenvalues and the one of the estimation of the dimensions using Cattell's scree-test ; for *hddc* the graph of the evolution of the log likelihood.

- ***d***: specifies the dimension for common dimension models. It also specifies whether the dimension is found with Cattell's scree-test or with the BIC criterion. Note that if the dimension of common dimension models is not explicitly specified, it is then directly estimated using the scree-test or the BIC criterion on the covariance matrix of the whole dataset. See also [4] on the use of the BIC criterion to select the number of intrinsic dimension.
- ***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.

The routine *hddc* have the additional following options:

- ***k***: designates the number of classes for which the classification is 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.
- ***iter.max***: the maximum number of iterations.
- ***eps***: defines the threshold value of the stop criterion. The algorithm stops when the difference between two successive log-likelihoods is below this value, default is 10^{-3}.
- ***algo***: the algorithm used can be either EM (the default value), CEM (Classification EM [9]), or SEM (Stochastic EM [8]). CEM is used to have a faster convergence: at each step, a cluster is allocated to each observation using the *maximum a posteriori* rule. SEM can be used to avoid initialization problems and to try not to stop in a local maximum of the likelihood. At each iteration, it allocates a cluster to each observation using a multinomial distribution of probability t_{ik} (the posterior probability that the observation i belongs to the class k).
- ***init***: there are two ways to initialize the algorithm: starting with the parameters or with the clusters. Four initializations have been implemented:

  ◇ *random*: a class is randomly affected to each observation.

  ◇ *kmeans*: the initial class of each observation is provided by the k-means algorithm; it is the default initialization.

  ◇ *param*: it is an initialization of the parameters. It was proposed by McLachlan and Peel [13], they suggest to equalize the proportions π_k of the mixture and generate the means μ_k of the mixture model by a multivariate Gaussian distribution $N(m, S)$ where m and S are respectively the mean and the covariance matrix of the whole dataset. The covariance matrices Σ_k are initialized to S.

  ◇ *mini-em*: the algorithm is run m times, doing each times nb iterations, the result with the highest likelihood is kept as the initialization of the algorithm. The parameters m and nb can be set with the *mini.nb* option.

  The user can also provide its own initialization by giving a class vector.
- ***scaling***: logical, whether to scale the dataset or not, default is *FALSE*.
- ***mini.nb***: this parameter settles the *mini-em* initialization, it is a vector of length 2, containing m and nb, its default value is *(5, 10)*.

Also, the routine *predict.hdc* have another option:

- ***cls***: this argument takes the original class vector of the dataset, it is optional.

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 (the noise variances).
  - ◇ d: the dimensions of the intrinsic dimensions of the classes.
  - ◇ *prop*: the proportions of each class.
  - ◇ *mu*: the means of each class.
  - ◇ *ev*: the eigenvalues of Σ_k, the covariance matrix of each class.
  - ◇ Q: the orthogonal matrix defining the orientation of each class.
- ***scaling***: contains the mean and the standard error of the original dataset.
- ***BIC***: the BIC value of the estimation.

Also *hddc* have this 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 class 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, the correct classification rate and the confusion matrix are shown on the R console.

A *print* method have been implemented to sum up the main parameters of the model and there is also a *plot* method in order to show Cattell's scree-test result with the possibility to change the threshold and then to see the selected new dimensions.

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 two last experiments of this section focus on the numerical advantages of both HDDA and HDDC.

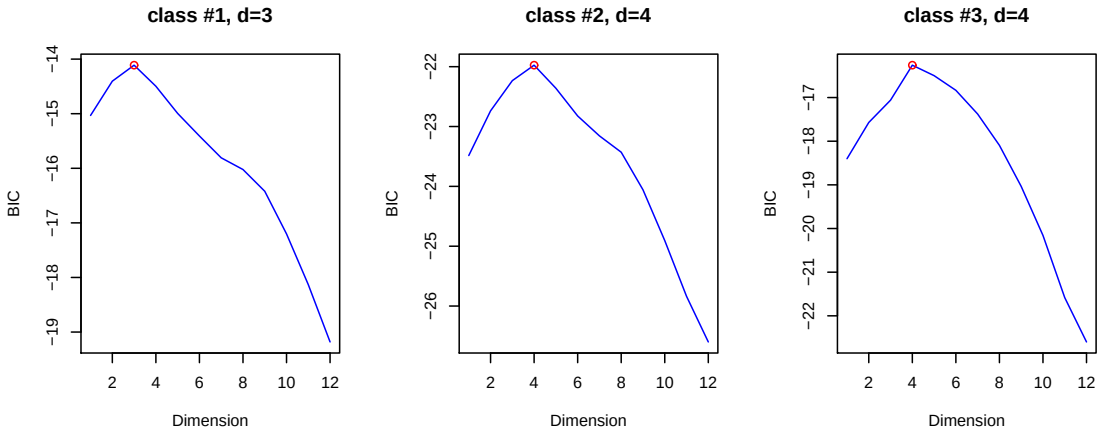

Figure 1: Selection of the intrinsic dimension of the classes in HDDA using the BIC criterion for the wine dataset.

4.1 HDDA: an introductory example

To introduce HDDA, we first use the "wine" dataset that can be found in the package. It is the results of a chemical analysis of wines from the same region in Italy but derived from $K = 3$ different cultivars. 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 are much larger than others. Then we choose to scale the dataset, which consists to put the mean to 0 and the standard error to 1 for each variable, using the command *scaling=TRUE*.

First results

The dataset is split into two different samples. The first one is used to learn the model while the second one will be used to test the method performance. The learning dataset is made of 40 randomly selected observations while the test is made on the 138 remaining ones. The estimation is done with the default model which is $[a_{kj}b_kQ_kd_k]$. The used R code and the results of the classification are shown below:

```
R>  data(wine)
R>  #the first column is the class vector.
R>  w <- wine[,-1]
R> cls <- wine[,1]
R> #we set the random seed so that the example can be replicated:
R> set.seed(1234)
R> #40 individuals are randomly chosen among 178:
R> ind <- sample(178,40)
R> #HDDA on the learning dataset:
R> prms <- hdda(w[ind,], cls[ind], scaling=TRUE)
R> #the results on the test dataset:
R> res <- predict(prms, w[-ind,], cls[-ind])
Correct classification rate : 0.9710145 .
               Initial class
```

```
Predicted class  1  2  3
              1 42  1  0
              2  1 55  0
              3  0  2 37
```

It first appears that the method shows good results 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. It is also easy to see the model parameters since a *print* method has been implemented to sum them up:

```
R> prms
HIGH DIMENSIONAL DISCRIMINANT ANALYSIS

MODEL : AKJBKQKDK

  Prior probabilities of groups :
     1     2     3
   0.4 0.325 0.275

       Intrinsic dimensions of the classes :
        1 2 3
  dim : 2 3 2

      Akj :
Class   a1   a2   a3
    1 1.66 1.03    .
    2 3.48 2.59 1.25
    3 3.11 1.59    .

        1    2     3
Bk : 0.19 0.19 0.179

BIC :  -34.87
```

Finally, one can see that the dimensions that are used to estimate the model are much lower than the initial ones. Indeed, here, the estimated intrinsic dimensions of the three classes are respectively 2, 3 and 2, which are lower dimensions than the 13 initial ones.

Intrinsic dimension selection

The choice of the number of intrinsic dimensions is now discussed since the d_k can be estimated using either the Cattell's scree-test or BIC. HDDA is first used with the BIC criterion to determine the intrinsic dimension of each class-specific subspace (using *d='BIC'*) and the result of the selection is then plotted using the command *graph=TRUE*.

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

Figure 1 shows the selection of the intrinsic dimensions of the classes using the BIC criterion. The dimensions can also be selected using Cattell's scree-test. Let see the scree-test result using the *plot* method on the previous parameters:

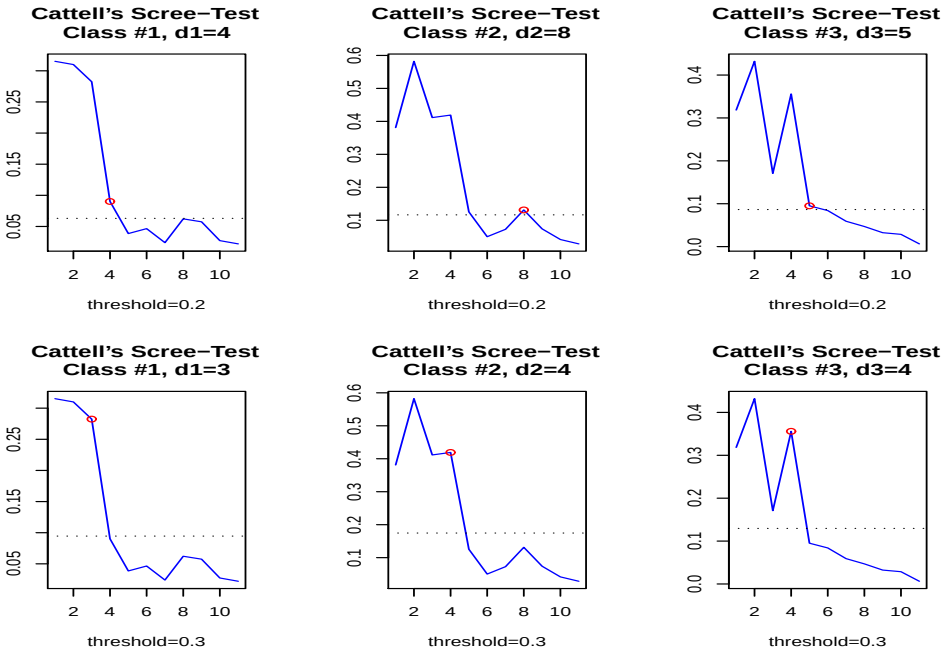

Figure 2: Effect of the Cattell's threshold on the dimension selection for the wine dataset with HDDA. The figures at the top have a threshold of 0.2 while the threshold is of 0.3 at the bottom.

```
R> plot(prms) #the default threshold is 0.2
R> #the threshold is risen to 0.3:
R> plot(prms, threshold=0.3)
```

Figure 2 shows the results of the two scree-tests. A change on the scree-test threshold, from 0.2 to 0.3, leads to a selection of less intrinsic dimensions. Then, with a higher threshold, the BIC criterion and the scree-test both select the same number of dimension 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. As an illustration, the dimensions selected here for the whole dataset are (3,4,4) while the dimensions in the first experiment are (2,3,2) which, nevertheless, shows good results (97% of good classification).

Using common dimension models

Common dimension models can also be used to classify this dataset:

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

The results of dimension selection with the Cattell's scree-test and the BIC criterion are displayed in Figure 3. The two criteria both choose 5 dimensions. In order to test the effectiveness of

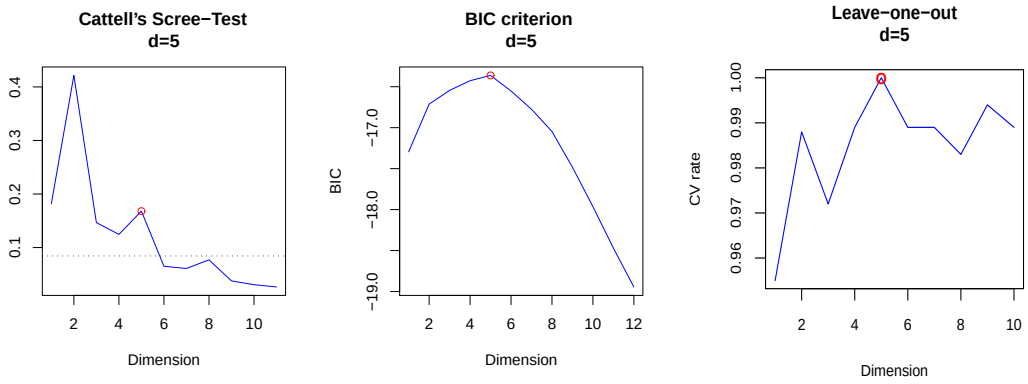

Figure 3: Intrinsic dimension selection with the Cattell's scree-test, the BIC criterion and cross-validation on the wine dataset for a common dimension model (model $[a_{kj}b_kQ_kd]$) of HDDA.

the dimension choice, we also apply cross-validation on the dataset using the same unconstrained common dimension model: $[a_{kj}b_kQ_kd]$. The cross-validation has been done with different number of common intrinsic dimensions, ranging from 1 to 10, with the leave-one-out scheme. The results are presented in the right panel of Figure 3 where it is shown that the best cross-validation result is for 5 dimensions, which is also the dimension selected by the default parameters of the model. As a conlusion, the dimension selection using either the two criteria is often the most efficient for predicting.

4.2 HDDC: an introductory example

HDDC is now introduced using the "Crabs" dataset. 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 collected 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> #the first column is the class vector.
R> A <- Crabs[,-1]
R> cls <- Crabs[,1]
R> prms <- hddc(A, 4)

     Model       k      BIC
    AKJBKQKDK    4    -14.15

R> e <- predict(prms, A, cls)
Correct classification rate : 0.94.
               Initial class
Predicted class BF BM OF OM
```

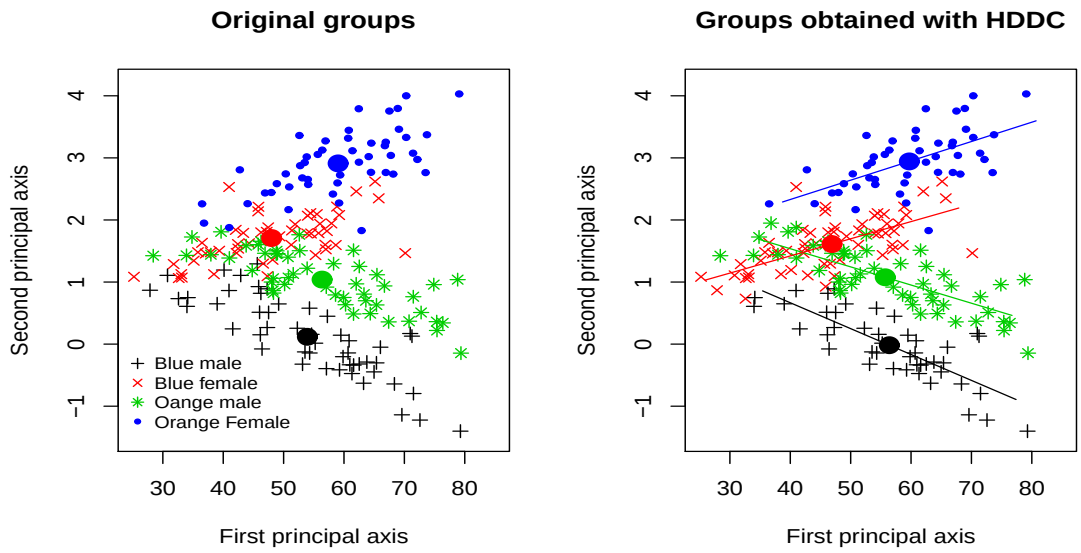

Figure 4: Clustering of the "Crabs" dataset, visualization on the 2 first principal axes. The segments represent the classes subspaces while the points are the means.

```
4 49  9  0  0
1  0 41  0  0
2  1  0 48  0
3  0  0  2 50
```

The obtained results show a correct classification rate of 94% and the confusion matrix is again helpful to understand the clustering: while the Orange Males (OM) are totally well classified, the Blue Male (BM) seems to have characteristics similar to the Blue Females (BF). Also there is only one mismatch between the two species.

PCA representation

Figure 4 shows the projection of the data on the first and second principal axes as well as 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 axes. The clusters, the means and the orientation of each class are also represented. The results are shown on Figure 5. This example can be run interactively in the *demo*(*hddc*) where the user can choose the algorithm, the model and the initialization and, then, see the effect on the clustering results. It can be observed on Figure 5 that, even with an initialization far from the truth, the EM algorithm updates sequentially the means and the orientations of the classes to finally reach a classification close to the actual one.

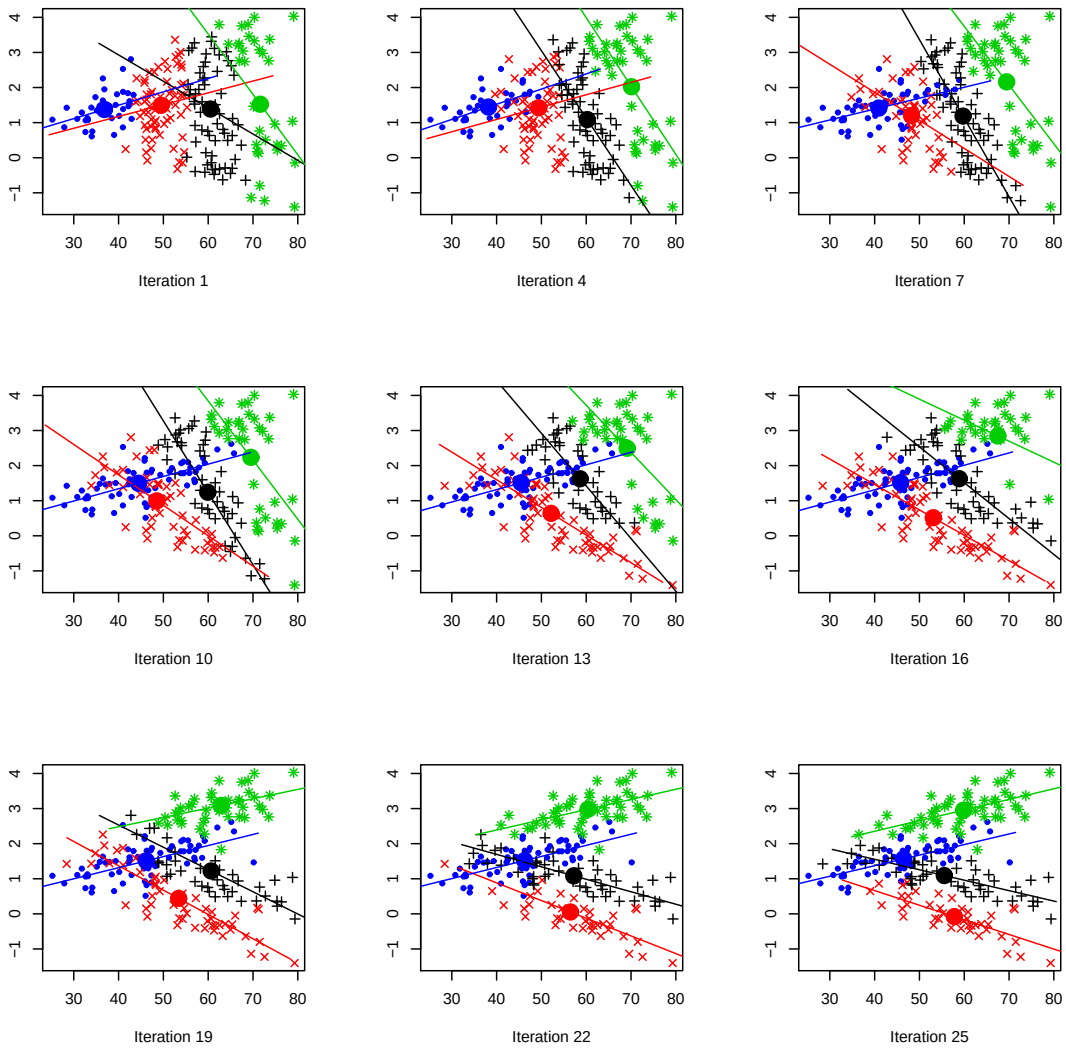

Figure 5: Clustering process of HDDC. The initialization is done with k-means.

Cluster selection

As it is a model-based clustering method, we can 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 7 classes:

```
R> prms <- hddc(A, k=1:7)
  Model     k      BIC
AKBKQKDK    1    -17.56
AKBKQKDK    2    -17.83
AKBKQKDK    3    -15.87
AKBKQKDK    4    -14.15
AKBKQKDK    5    -17.94
AKBKQKDK    6    -18.17
AKBKQKDK    7    -17.96

SELECTED : model AKBKQKDK with 4 clusters.
```

It appears that BIC seems to be a reliable criterion in selecting the number of clusters as it points out to the right number of classes for 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 $\mathbb{R}^p$, $p = 20, ..., 200$, with respect to the model $[a_k b_k Q_k d_k]$ with the following parameters : $\{d_1, d_2, , d_3\} = \{2, 5, 10\}$, $\{\pi_1, \pi_2, \pi_3\} = \{0.4, 0.3, 0.3\}$, $\{a_1, a_2, a_3\} = \{150, 75, 50\}$ and $\{b_1, b_2, b_3\} = \{15, 10, 5\}$, with close means and random orientation matrices Q_k. The learning and testing datasets are respectively made of 250 and 1000 points. The performance of each method is 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 $[a_k b_k Q_k d_k]$ is used in HDDA and is compared to three other methods: QDA, LDA and PCA+LDA (LDA on 15-dimensional data projected with PCA). The results of each method are represented as boxplots in Figure 6.

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 dimension 70 and then did not work anymore with dimension 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, HDDA appears not to be sensible to large dimension as it provides good results in large as well as in low dimension. Furthermore, the figure 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.

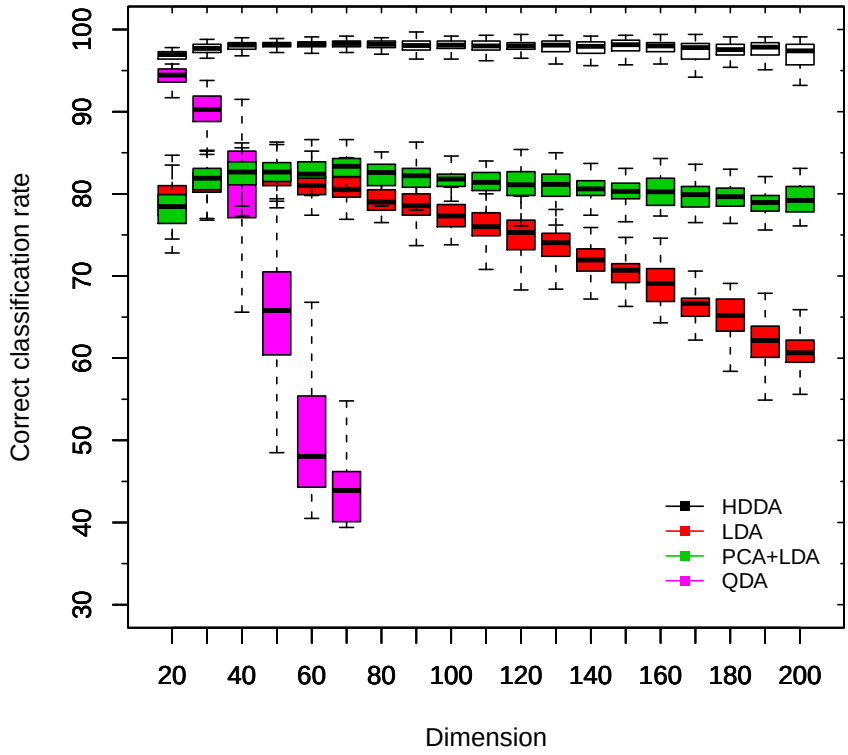

Figure 6: Boxplots of the supervised classification results for different methods on a simulated dataset.

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 Section 4.3. 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. For this experiment, we used HDDC with the model $[a_k b_k Q_k d_k]$ with a "mini-em" initialization. HDDC is also compared here to three other clustering methods based on the Gaussian mixture model which can be found in the *R* package *Mclust* [11]. 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.

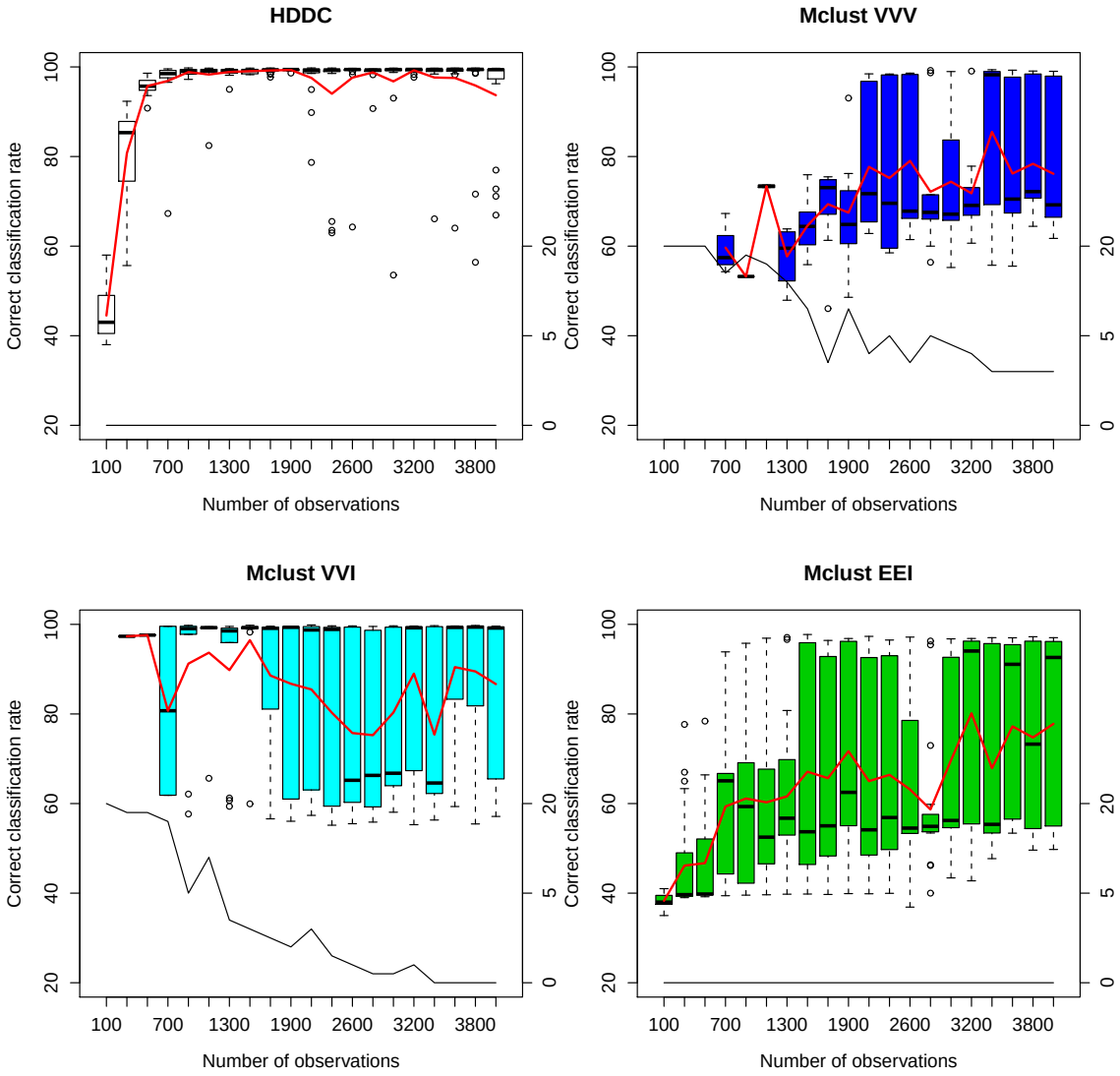

Figure 7: Effect of the dimension on correct classification rates on simulated data with HDDC (model $[a_k b_k Q_k d_k]$) and Mclust (models VVV, VVI and EEI). The red lines represent the means of the correct classification rates (when the algorithm converged) while the black lines represent the numbers of times that the algorithms could not be fitted in the 20 simulations (its scale is at right).

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

Table 3: Average computation times of HDDC and Mclust on simulated datasets with varying dimensions and observation numbers (in seconds).

The results of this experiment are presented in Figure 7. The 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). It first appears that all tested models of 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. 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 of Mclust, 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 of Mclust 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. Finally, HDDC appears to be insensitive to the *curse of the dimensionality* for a large range of dataset sizes (n larger than 300). Naturally, the robustness of HDDC weakens for very small datasets. In such a case, it would be preferable to use a more parsimonious model of HDDC (the model $[abQ_kd_k]$ for instance).

4.5 HDDC: computing time comparison

Here is finally tested the effect of the dimension and of the number of observations on the computing time. In order to realize this experiment, a dataset has been simulated 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 3, show how the computing time rises with the dimension and the number of observations. It clearly appears that HDDC is faster than Mclust and that the impact of the rise of dimension or of the number of observations is much less important on HDDC than on Mclust.

5 Applications

The methods HDDA and HDDC are now applied on two real-world datasets that have in common to be in high dimension[2]. The first one contains images represented as 256-dimensional observations whereas the second one is made of spectra with more than 6,000 dimensions.

Figure 8: Some examples of the USPS dataset used for the OCR experiment.

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

5.1 Optical character recognition

HDDA is first tested on the OCR dataset used for the study of the United States Postal service (USPS)[1], which consists in the recognition of handwritten numbers. 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 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, $n = 100, ..., 2000$. Figure 9 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 4 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 the table in order to show that a linear method with only one covariance matrix to estimate can again faster the computation.

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

[1]This dataset can be found on the site of the university of Aachen: `http://www-i6.informatik.rwth-aachen.de/~keysers/usps.html`.

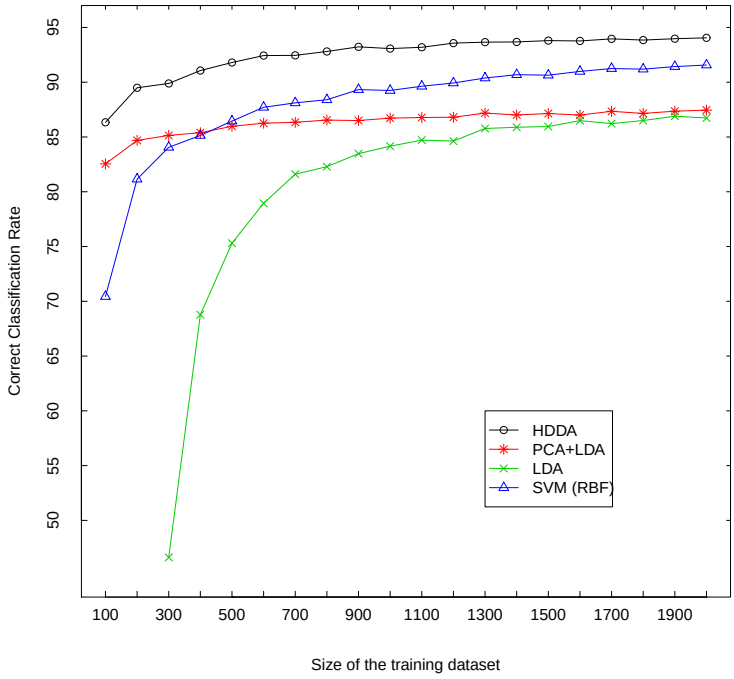

Figure 9: Influence of the size of the learning dataset on prediction results obtained with HDDA and other classification methods on the USPS dataset.

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 10 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, 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 [1], 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 cross-validation is done for each classification method and the results are shown on Table 5. First is to notice that LDA is totally inefficient on this dataset that has a large number of parameters. SVM works very well with no missclassification but with a prohibitive computation time. HDDA gives satisfying results, upper than 94% of good classification for the two models and has fast computation time as the method is more than 3 times faster than SVM.

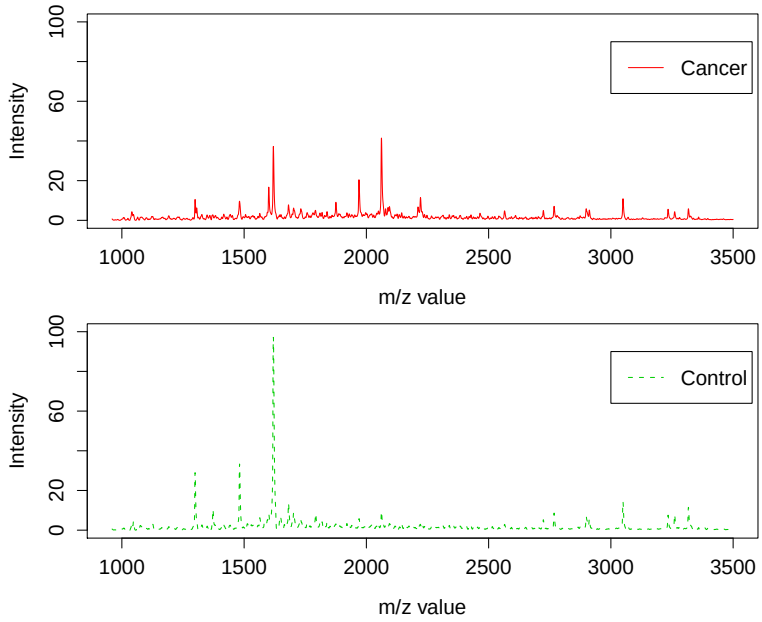

Figure 10: Mean spectra of the cancer class (up) and of the control class (bottom) on the m/z interval 900–3500 Da.

Method	Result	Computing time
$[a_{kj}b_kQ_kd_k]$	0.94	142.46
$[a_jbQd]$	0.95	145.50
LDA	0.52	24.01
SVM	1.00	457.36

Table 5: Results of cross-validation on the Maldi mass-spectrometry dataset, and computing times (in seconds).

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 comparison sake, PCA-EM and mixture of PPCA (Mixt-PPCA) [19] 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 6. All the methods present good results for such a complex problem, although the best level of classification has been obtained with HDDC with a missclassification 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 [6] which takes into account that high-dimensional data live in low-dimensional subspaces. The proposed models

PCA-EM		
	cluster	
class	cancer	control
cancer	48	16
control	1	47
Missclassification rate = *0.15*		

Mixt-PPCA		
	cluster	
class	cancer	control
cancer	62	2
control	10	38
Missclassification rate = *0.11*		

HDDC		
	cluster	
class	cancer	control
cancer	62	6
control	0	45
Missclassification rate = *0.05*		

Table 6: Confusion matrices of the three studied clustering methods on the Maldi mass-spectrometry dataset.

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.

References

[1] T. Alexandrov, J. Decker, B. Mertens, A.M. Deelder, R.A. Tollenaar, P. Maass, and H. Thiele. Biomarker discovery in MALDI-TOF serum protein profiles using discrete wavelet transformation. *Bioinformatics*, 25(5):643–649, 2009.

[2] R. Bellman. *Dynamic programming*. Princeton University Press, 1957.

[3] H.-H. Bock. Probabilistic models in cluster analysis. *Comput. Statist. Data Anal.*, 23(1):5–28, 1996.

[4] C. Bouveyron, G. Celeux, and S. Girard. Intrinsic dimension estimation by maximum likelihood in probabilistic PCA. Technical Report 440372, Université Paris 1 Panthéon-Sorbonne, 2010.

[5] C. Bouveyron, S. Girard, and C. Schmid. High Dimensional Data Clustering. *Comput. Statist. Data Anal.*, 52:502–519, 2007.

[6] C. Bouveyron, S. Girard, and C. Schmid. High Dimensional Discriminant Analysis. *Comm. Statist. Theory Methods*, 36(14):2607–2623, 2007.

[7] R. Cattell. The scree test for the number of factors. *Multivariate Behav. Res.*, 1(2):245–276, 1966.

[8] G. Celeux and J. Diebolt. The SEM algorithm: a probabilistic teacher algorithm from the EM algorithm for the mixture problem. *Computational Statistics Quaterly*, 2(1):73–92, 1985.

[9] G. Celeux and G. Govaert. A classification EM algorithm for clustering and two stochastic versions. *Comput. Statist. Data Anal.*, 14:315–332, 1992.

[10] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39(1):1–38, 1977.

[11] C. Fraley and A. Raftery. MCLUST: Software for Model-Based Cluster Analysis. *Journal of Classification*, 16:297–306, 1999.

[12] G. McLachlan. *Discriminant Analysis and Statistical Pattern Recognition*. Wiley, New York, 1992.

[13] G. McLachlan and D. Peel. *Finite Mixture Models*. Wiley Interscience, New York, 2000.

[14] G. McLachlan, D. Peel, and R. Bean. Modelling high-dimensional data by mixtures of factor analyzers. *Comput. Statist. Data Anal.*, 41:379–388, 2003.

[15] T. Pavlenko. On feature selection, curse of dimensionality and error probability in discriminant analysis. *Journal of Statistical Planning and Inference*, 115:565–584, 2003.

[16] T. Pavlenko and D. Von Rosen. Effect of dimensionality on discrimination. *Statistics*, 35(3):191–213, 2001.

[17] G. Schwarz. Estimating the dimension of a model. *Ann. Statist.*, 6:461–464, 1978.

[18] D. Scott and J. Thompson. Probability density estimation in higher dimensions. In *Fifteenth Symposium in the Interface*, pages 173–179, 1983.

[19] M. Tipping and C. Bishop. Mixtures of probabilistic principal component analysers. *Neur. Comput.*, 11(2):443–482, 1999.