Adv Data Anal Classif (2017) 11:243–279
DOI 10.1007/s11634-016-0242-1

REGULAR ARTICLE

Model-based regression clustering for high-dimensional data: application to functional data

Emilie Devijver[1]

Received: 30 April 2015 / Revised: 19 January 2016 / Accepted: 26 February 2016 /
Published online: 14 March 2016

Abstract Finite mixture regression models are useful for modeling the relationship between response and predictors arising from different subpopulations. In this article, we study high-dimensional predictors and high-dimensional response and propose two procedures to cluster observations according to the link between predictors and the response. To reduce the dimension, we propose to use the Lasso estimator, which takes into account the sparsity and a maximum likelihood estimator penalized by the rank, to take into account the matrix structure. To choose the number of components and the sparsity level, we construct a collection of models, varying those two parameters and we select a model among this collection with a non-asymptotic criterion. We extend these procedures to functional data, where predictors and responses are functions. For this purpose, we use a wavelet-based approach. For each situation, we provide algorithms and apply and evaluate our methods both on simulated and real datasets, to understand how they work in practice.

Keywords Model-based clustering · Regression · High-dimension · Functional data

Mathematics Subject Classification 62H30

1 Introduction

Owing to the increasing availability of high-dimensional datasets, regression models for multivariate response and high-dimensional predictors have become important tools.

✉ Emilie Devijver
emilie.devijver@math.u-psud.Fr

1 Inria Select, Université Paris Sud, Bât. 425, 91405 Orsay Cedex, France

In this article, we describe two procedures where a random target variable $\boldsymbol{Y} \in \mathbb{R}^q$ depends on explanatory variables within a cluster-specific regression model. Each cluster is represented by a parametric distribution, the entire dataset being modeled by a mixture of these distributions. The model assumes that each observation $i \in \{1, \ldots, n\}$ originates from one of K disjoint classes and that the data $(\boldsymbol{Y}_i, \boldsymbol{x}_i) \in \mathbb{R}^q \times \mathbb{R}^p$ are independent and identically distributed such that if i belongs to class $k \in \{1, \ldots, K\}$, the target variable $\boldsymbol{Y}_i$ results from a regression model

$$\boldsymbol{Y}_i = \mathbf{B}_k \boldsymbol{x}_i + \boldsymbol{\varepsilon}_i \tag{1}$$

with an unknown matrix of coefficients $\mathbf{B}_k \in \mathbb{R}^{q \times p}$ and independent errors $\boldsymbol{\varepsilon}_i \sim \mathcal{N}_q(0, \Sigma_k)$ with an unknown diagonal covariance matrix Σ_k of size $q \times q$. Each observation $i \in \{1, \ldots, n\}$ has a probability π_k to belong to the cluster $k \in \{1, \ldots, K\}$.

We work with high-dimensional data, in other words the number of parameters to estimate $K(qp + q + 1) - 1$ is larger than the number of observed target values $q \times n$. Two ways are considered in this paper, coefficients sparsity and ranks sparsity. The first approach consists in estimating the matrix $\mathbf{B}_k$ by a matrix with few nonzero coefficients. The well-known Lasso estimator, introduced by Tibshirani (1996) for linear models, is the solution chosen here. We refer to Bühlmann and van de Geer (2011) for an overview of the Lasso estimator and to Meinshausen and Bühlmann (2010) for stability selection results. In the second approach, we consider the rank sparsity in $\mathbf{B}_k$. This idea dates back to the 1950's and was initiated by Anderson (1951) for the linear model. Izenman (1975) introduced the term of reduced-rank regression for this class of models. For more recent works, we refer to Giraud (2011) and to Bunea et al. (2012). Nevertheless, the linear model is appropriate for homogeneous observations, which is a strong assumption. We extend in this paper those methods to mixture regression models.

Remark that we estimate the number of components, model parameters and cluster proportions. In the case of a large number of regressor variables, we use variable selection tools in order to detect relevant regressors. Since the structure of interest may often be contained into a subset of available variables and many attributes may be useless or even harmful to detect a reasonable clustering structure, it is important to select the relevant indices. Moreover, removing irrelevant indices enables us to get an easier model and largely enhance comprehension.

An important extension of a high-dimensional dataset is a functional dataset. We refer to Ramsay and Silverman (2005) for an overview. Moreover, a lot of recent works have been done on regression models for functional data: for example, we refer to Ciarleglio and Ogden (2014) for a study with scalar response and functional regressors. In this paper, we focus on the projection of functions onto a well-suited wavelet basis. Indeed, wavelets handle many types of functional data, because they represent global and local attributes of functions and can deal for example with discontinuities. Moreover, a large class of functions is well represented with few nonzero coefficients for a suitable wavelet, which leads to a sparse matrix of coefficients and then to a sparse regression matrix.

We propose here two procedures for clustering high-dimensional or functional data, where the high-dimensional or functional random target variable $\boldsymbol{Y} \in \mathbb{R}^q$ depends on high-dimensional or functional predictors $\boldsymbol{x} \in \mathbb{R}^p$ with a cluster-specific regression model.

Our two procedures are mainly based on three recent works. Firstly, we refer to the article of Städler et al. (2010), which studies a finite mixture regression model. Even if we work on a multivariate version of it, the model considered in the article of Städler et al. (2010) is adopted here. The second, the article of Meynet and Maugis-Rabusseau (2012), deals with model-based clustering in density estimation. They propose a procedure, called the Lasso-MLE procedure, which determines the number of clusters, the set of relevant indices for the clustering and a clustering of the observations, with high-dimensional data. We extend this procedure to regression models. Finally, Giraud (2011) suggests a low rank estimator for the linear model. To consider the matrix structure, we extend this last approach to mixture models.

We consider a finite mixture of Gaussian regression models. The two procedures we propose follow the same steps. Firstly, a penalized likelihood approach is considered to determine potential sets of relevant indices. Varying the regularization parameter, a data-driven collection of models is constructed where each model has a reasonable complexity. The second step of the procedures consists in refitting parameters by a less biased estimator, restricting the model on selected indices. Then, we select a model among the collection using the slope heuristic, which was introduced by Birgé and Massart (2007). The difference between the two procedures is in the refitting step. In the first procedure, later called Lasso-MLE procedure, the maximum likelihood estimator is used. The second procedure, called Lasso-Rank procedure, deals with low rank estimation. For each model in the collection, a subcollection of models with means estimated by various low rank matrices is constructed. It leads to sparsity for the coefficients and for the rank, and it considers the mean within its matrix structure.

The article is organized as follows. Section 2 deals with Gaussian mixture regression models and the considered collection of models is described. In Sect. 3, the two procedures that we propose to solve the problem of high-dimensional regression data clustering are described. Section 4 presents an illustrative example, to highlight each choice done in the two procedures. Section 5 states the functional data case, with a description of the projection proposed to convert these functions into coefficients data. We end this section by studying simulated and benchmark data. Finally, a conclusion section ends this article.

2 Finite mixture of Gaussian regression models

The model used is a finite Gaussian mixture regression model. Städler et al. (2010) describe this model when the deterministic predictors $\boldsymbol{x} \in \mathbb{R}^p$ are multivariate and the target variable $\boldsymbol{Y} \in \mathbb{R}$ is a scalar. In this section, it is generalized to multivariate response. Let us mention that this model has already been introduced, we refer for example to Jones and McLachlan (1992). In this paper, we deal with high-dimensional data.

2.1 The model

We observe n independent couples $((\boldsymbol{y}_1, \boldsymbol{x}_1), \ldots, (\boldsymbol{y}_n, \boldsymbol{x}_n))$, where $\boldsymbol{x}_i \in \mathbb{R}^p$ are deterministic predictors and $\boldsymbol{y}_i \in \mathbb{R}^q$ is a realization of a random variable $\boldsymbol{Y}_i \in \mathbb{R}^q$ of unknown density $s^*(.|\boldsymbol{x}_i)$, for all $i \in \{1, \ldots, n\}$. The random response variable $\boldsymbol{Y}_i \in \mathbb{R}^q$ depends on a set of explanatory variables, written $\boldsymbol{x}_i \in \mathbb{R}^p$, through a regression-type model. If $(\boldsymbol{Y}_i, \boldsymbol{x}_i)$ originates from an individual i in class k, we assume that

$$\boldsymbol{Y}_i = \mathbf{B}_k \boldsymbol{x}_i + \boldsymbol{\varepsilon}_i;$$

where $\boldsymbol{\varepsilon}_i \sim \mathcal{N}_q(0, \boldsymbol{\Sigma}_k)$, $\mathbf{B}_k \in \mathbb{R}^{q \times p}$ is the matrix of class-specific regression coefficients and $\boldsymbol{\Sigma}_k$ is $q \times q$ diagonal positive-definite matrix.

We assume the following Gaussian regression mixture model:

- $(\boldsymbol{Y}_1, \ldots, \boldsymbol{Y}_n)$ are independent,
- $\boldsymbol{Y}_i$ has the density $s^K_{\boldsymbol{\xi}}(.|\boldsymbol{x}_i)$, with

$$\begin{aligned}
s^K_{\boldsymbol{\xi}}(\boldsymbol{y}|\boldsymbol{x}) &= \sum_{k=1}^{K} \frac{\pi_k}{(2\pi)^{q/2} \det(\boldsymbol{\Sigma}_k)^{1/2}} \exp\left(- \frac{(\boldsymbol{y} - \mathbf{B}_k \boldsymbol{x})^t \boldsymbol{\Sigma}_k^{-1} (\boldsymbol{y} - \mathbf{B}_k \boldsymbol{x})}{2} \right); \\
\boldsymbol{\xi} &= (\pi_1, \ldots, \pi_K, \mathbf{B}_1, \ldots, \mathbf{B}_K, \boldsymbol{\Sigma}_1, \ldots, \boldsymbol{\Sigma}_K) \in \Xi_K \\
\Xi_K &= \left(\Pi_K \times (\mathbb{R}^{q \times p})^K \times (\mathbb{D}_q^{++})^K \right); \\
\Pi_K &= \left\{ (\pi_1, \ldots, \pi_K); \pi_k > 0 \text{ for } k \in \{1, \ldots, K\} \text{ and } \sum_{k=1}^{K} \pi_k = 1 \right\}; \\
\mathbb{D}_q^{++} &\text{ is the set of } q \times q \text{ diagonal and positive-definite matrices.}
\end{aligned}$$

We have denoted by π_k the proportion of the class k. For all $k \in \{1, \ldots, K\}$, for all $m \in \{1, \ldots, q\}$, for $\boldsymbol{x} = (\xi_1, \ldots, \xi_p) \in \mathbb{R}^p$, $[\mathbf{B}_k \boldsymbol{x}]_m = \sum_{j=1}^{p} [\mathbf{B}_k]_{m,j} \xi_j$ is the mth component of the mean of the kth mixture component.

We prefer to work with a reparametrized version of this model whose penalized maximum likelihood estimator is scale-invariant and easier to compute. Indeed, it is not equivariant under scaling of the response. More precisely, consider the transformation

$$\check{\boldsymbol{Y}} = b\boldsymbol{Y}, \check{\boldsymbol{B}} = b\boldsymbol{B}, \check{\boldsymbol{\Sigma}} = b^2 \boldsymbol{\Sigma},$$

for $b > 0$ which leaves the model invariant. A reasonable estimator based on transformed data $\check{\boldsymbol{Y}}$ should lead to an estimator which is related to the first one up to the homothetic transformation. This is not the case for the maximum likelihood estimator of $\boldsymbol{B}$ and $\boldsymbol{\Sigma}$. Secondly, the optimization of the log-likelihood is non-convex and hence, it leads to computational issues. Then, we reparametrize the model described above by generalizing the reparametrization described by Städler et al. (2010).

For all $k \in \{1, \dots, K\}$, we define the new parameters by

$$\begin{aligned} \boldsymbol{P}_k^t \boldsymbol{P}_k &= \boldsymbol{\Sigma}_k^{-1}; \\ \boldsymbol{\Phi}_k &= \boldsymbol{P}_k \mathbf{B}_k; \end{aligned}$$

where $\boldsymbol{\Phi}_k \in \mathbb{R}^{q \times p}$, $\boldsymbol{P}_k$ is the Cholesky decomposition of the positive-definite matrix $\boldsymbol{\Sigma}_k^{-1}$. Remark that $\boldsymbol{P}_k$ is a diagonal matrix of size $q \times q$.

The model within its reparametrized form is then

- $(\boldsymbol{Y}_1, \dots, \boldsymbol{Y}_n)$ are independent,
- $\boldsymbol{Y}_i$ has the density $h_\theta^K(.|\boldsymbol{x}_i)$, with

$$\begin{aligned} h_\theta^K(\boldsymbol{y}|\boldsymbol{x}) &= \sum_{k=1}^K \frac{\pi_k \det(\boldsymbol{P}_k)}{(2\pi)^{q/2}} \exp\left(-\frac{(\boldsymbol{P}_k \boldsymbol{y} - \boldsymbol{\Phi}_k \boldsymbol{x})^t (\boldsymbol{P}_k \boldsymbol{y} - \boldsymbol{\Phi}_k \boldsymbol{x})}{2}\right) \\ \theta &= (\pi_1, \dots, \pi_K, \boldsymbol{\Phi}_1, \dots, \boldsymbol{\Phi}_K, \boldsymbol{P}_1, \dots, \boldsymbol{P}_K) \in \Theta_K \\ &= \left(\Pi_K \times (\mathbb{R}^{p \times q})^K \times (\mathbb{D}_q^{++})^K\right) \\ \Pi_K &= \left\{(\pi_1, \dots, \pi_K); \pi_k > 0 \text{ for } k \in \{1, \dots, K\} \text{ and } \sum_{k=1}^K \pi_k = 1\right\} \\ \mathbb{D}_q^{++} & \text{ is the set of diagonal and positive-definite matrices in } \mathbb{R}^q. \end{aligned}$$

For the sample $((\boldsymbol{y}_1, \boldsymbol{x}_1), \dots, (\boldsymbol{y}_n, \boldsymbol{x}_n))$, the log-likelihood of this model equals

$$\begin{aligned} & l(\theta, \boldsymbol{y}_1, \boldsymbol{x}_1, \dots, \boldsymbol{y}_n, \boldsymbol{x}_n) \\ &= \sum_{i=1}^n \log\left(\sum_{k=1}^K \frac{\pi_k \det(\boldsymbol{P}_k)}{(2\pi)^{q/2}} \exp\left(-\frac{(\boldsymbol{P}_k \boldsymbol{y}_i - \boldsymbol{\Phi}_k \boldsymbol{x}_i)^t (\boldsymbol{P}_k \boldsymbol{y}_i - \boldsymbol{\Phi}_k \boldsymbol{x}_i)}{2}\right)\right); \end{aligned}$$

and the maximum likelihood estimator (MLE) is

$$\hat{\theta}^{MLE} := \underset{\theta \in \Theta_K}{\operatorname{argmin}} \left\{-\frac{1}{n} l(\theta, \boldsymbol{y}_1, \boldsymbol{x}_1, \dots, \boldsymbol{y}_n, \boldsymbol{x}_n)\right\}.$$

Since we deal with the high-dimension case, this estimator has to be regularized to get stable estimates. Therefore, we propose the ℓ_1-norm penalized MLE

$$\hat{\theta}^{\text{Lasso}}(\lambda) := \underset{\theta \in \Theta_K}{\operatorname{argmin}} \left\{-\frac{1}{n} l_\lambda(\theta, \boldsymbol{y}_1, \boldsymbol{x}_1, \dots, \boldsymbol{y}_n, \boldsymbol{x}_n)\right\}; \tag{2}$$

where

$$-\frac{1}{n} l_\lambda(\theta, \boldsymbol{y}_1, \boldsymbol{x}_1, \dots, \boldsymbol{y}_n, \boldsymbol{x}_n) = -\frac{1}{n} l(\theta, \boldsymbol{y}_1, \boldsymbol{x}_1, \dots, \boldsymbol{y}_n, \boldsymbol{x}_n) + \lambda \sum_{k=1}^K \pi_k ||\boldsymbol{\Phi}_k||_1;$$

where $||\boldsymbol{\Phi}_k||_1 = \sum_{j=1}^{p} \sum_{m=1}^{q} |[\boldsymbol{\Phi}_k]_{m,j}|$ and where $\lambda > 0$ has to be specified. Remark that this estimator is not the usual ℓ_1-estimator, called the Lasso estimator, introduced by Tibshirani (1996). Indeed, the ℓ_1-norm of the coefficients matrices $\boldsymbol{\Phi}_k$ and small variances are penalized simultaneously, which has some close relations to the Bayesian Lasso (see Park and Casella (2008)). Moreover, the reparametrization allows us to consider non-standardized data.

Notice that we restrict ourselves in this article to diagonal covariance matrices which are dependent of the clusters. We assume that the coordinates of the $\boldsymbol{Y}_i$ are independent, which is a strong assumption, but it allows us to reduce easily the dimension. We refer to Celeux and Govaert (1995) for different parametrizations of the covariance matrix. Nevertheless, because we work with high-dimensional data (p and q are high and n is small) we prefer a parsimonious model from the diagonal family. We allow different volume clusters because it leads to detect many clustering structures, as explained in Celeux and Govaert (1995).

2.2 Clustering with finite mixture of Gaussian regression models

Suppose that there is a known number of clusters K and assume that we get, from the observations, an estimator $\hat{\theta}$ such that $h_{\hat{\theta}}^{K}$ well approximates the unknown density $s^*(.|\boldsymbol{x}_i)$. We look at this problem as a missing data problem: if we denote by $\boldsymbol{Z} = (\boldsymbol{Z}_1, \ldots, \boldsymbol{Z}_n)$ the component membership, where $\boldsymbol{Z}_i = ([\boldsymbol{Z}_i]_1, \ldots, [\boldsymbol{Z}_i]_K)$ for $i \in \{1, \ldots, n\}$ is defined by

$$[\boldsymbol{Z}_i]_k = \begin{cases} 1 & \text{if } i \text{ originates from mixture component } k; \\ 0 & \text{otherwise;} \end{cases}$$

the complete data are $((\boldsymbol{y}_1, \boldsymbol{x}_1, z_1), \ldots, (\boldsymbol{y}_n, \boldsymbol{x}_n, z_n))$. Remark that $\boldsymbol{Z}_i$ follows a multinomial distribution with parameters 1 and $(\pi_1, \ldots, \pi_K)$.

Thanks to the estimator $\hat{\theta}$, the Maximum A Posteriori principle (MAP principle) is used to cluster data. Specifically, for all $i \in \{1, \ldots, n\}$, for all $k \in \{1, \ldots, K\}$, consider

$$\hat{\boldsymbol{\tau}}_{i,k}(\hat{\theta}) = \frac{\hat{\pi}_k \det(\hat{\boldsymbol{P}}_k) \exp\left(-\frac{1}{2}(\hat{\boldsymbol{P}}_k \boldsymbol{y}_i - \hat{\boldsymbol{\Phi}}_k \boldsymbol{x}_i)^t (\hat{\boldsymbol{P}}_k \boldsymbol{y}_i - \hat{\boldsymbol{\Phi}}_k \boldsymbol{x}_i)\right)}{\sum_{r=1}^{K} \hat{\pi}_r \det(\hat{\boldsymbol{P}}_r) \exp\left(-\frac{1}{2}(\hat{\boldsymbol{P}}_r \boldsymbol{y}_i - \hat{\boldsymbol{\Phi}}_r \boldsymbol{x}_i)^t (\hat{\boldsymbol{P}}_r \boldsymbol{y}_i - \hat{\boldsymbol{\Phi}}_r \boldsymbol{x}_i)\right)}$$

the posterior probability of $\boldsymbol{y}_i$ with i coming from the component k, where $\theta = (\pi, \boldsymbol{\Phi}, \boldsymbol{P})$. Then, data are partitioned by the following rule:

$$[\hat{\boldsymbol{Z}}_i]_k = \begin{cases} 1 & \text{if } \hat{\boldsymbol{\tau}}_{i,k}(\hat{\theta}) > \hat{\boldsymbol{\tau}}_{i,l}(\hat{\theta}) \text{ for all } l \neq k \text{ ;} \\ 0 & \text{otherwise.} \end{cases}$$

2.3 Numerical optimization

First, we introduce a generalized EM algorithm to approximate $\hat{\theta}^{\text{Lasso}}(\lambda)$ defined in (2) for $\lambda > 0$. Then, we discuss initialization and stopping rules in practice, before studying theoretically the convergence of this algorithm.

2.3.1 Generalized EM algorithm for approximating the Lasso estimator

From an algorithmic point of view, we use a generalized EM algorithm to compute the MLE and the ℓ_1-norm penalized MLE. The EM algorithm was introduced by Dempster et al. (1977) to approximate the MLE of mixture model parameters. It is an iterative procedure based on the maximization of the expected value of the log-likelihood function with respect to the conditional distribution of $\boldsymbol{Z}$ given the observations, under the current estimation of the parameters. Using the Karush–Kuhn–Tucker conditions, we extend the second step to compute the MLE, penalized or not, under rank constraint or not, as it was done in the scalar case in Städler et al. (2010). All those computations are available in Appendix "EM algorithms". The next updating formulae for the Lasso estimator defined by (2) are then obtained. Remark that it includes MLE. In the following, we consider the real n-space $\mathbb{R}^n$ with the dot product denoted by $< .,. >$, and with the associated norm denoted by $||.||_2$. For all $j \in \{1,\dots,p\}$, for all $k \in \{1,\dots,K\}$, for all $m \in \{1,\dots,q\}$, for all $i \in \{1,\dots,n\}$, we denote

$$
\begin{aligned}
[\widetilde{\boldsymbol{y}}_i^{(\text{ite})}]_{k,m} &= \sqrt{\boldsymbol{\tau}_{i,k}^{(\text{ite})}}[\boldsymbol{y}_i]_m; \\
[\widetilde{\boldsymbol{x}}_i^{(\text{ite})}]_{k,j} &= \sqrt{\boldsymbol{\tau}_{i,k}^{(\text{ite})}}[\boldsymbol{x}_i]_j; \\
\Delta_{k,m} &= \left(-n_k^{(\text{ite})}\langle[\widetilde{\boldsymbol{y}}_.^{(\text{ite})}]_{k,m}, [\boldsymbol{\Phi}_k]_{m,.}^{(\text{ite})}[\widetilde{\boldsymbol{x}}^{(\text{ite})}_.]_{k,.}\rangle\right)^2 - 4||[\widetilde{\boldsymbol{y}}_.^{(\text{ite})}]_{k,m}||_2^2; \qquad (3)
\end{aligned}
$$

$$
[\boldsymbol{S}_k]_{j,m}^{(\text{ite})} = -\sum_{i=1}^{n}[\widetilde{\boldsymbol{x}}_i^{(\text{ite})}]_{k,j}[\boldsymbol{P}_k]_{m,m}^{(\text{ite})}[\widetilde{\boldsymbol{y}}_i^{(\text{ite})}]_{k,m} + \sum_{\substack{j_2=1\\ j_2\neq j}}^{p}[\widetilde{\boldsymbol{x}}_i^{(\text{ite})}]_{k,j}[\widetilde{\boldsymbol{x}}_i^{(\text{ite})}]_{k,j_2}[\boldsymbol{\Phi}_k]_{m,j_2}^{(\text{ite})};
$$

$$
n_k = \sum_{i=1}^{n}\boldsymbol{\tau}_{i,k}^{(\text{ite})}. \qquad (4)
$$

Moreover, we consider $t^{(\text{ite})} \in (0,1]$ the largest value in $\{0.1^l, l \in \mathbb{N}\}$ such that the update of π defined in (7) leads to improving the expected complete penalized log-likelihood function. Then, we update the parameters by

$$
g_{i,k}^{(\text{ite})} = \exp\left(-\frac{1}{2}\left(\boldsymbol{P}_k^{(\text{ite})}\boldsymbol{y}_i - \boldsymbol{\Phi}_k^{(\text{ite})}\boldsymbol{x}_i\right)^t\left(\boldsymbol{P}_k^{(\text{ite})}\boldsymbol{y}_i - \boldsymbol{\Phi}_k^{(\text{ite})}\boldsymbol{x}_i\right)\right) \qquad (5)
$$

$$
\boldsymbol{\tau}_{i,k}^{(\text{ite})} = \frac{\pi_k^{(\text{ite})}\det\boldsymbol{P}_k^{(\text{ite})}g_{i,k}^{(\text{ite})}}{\sum_{r=1}^{K}\pi_r^{(\text{ite})}\det\boldsymbol{P}_r^{(\text{ite})}g_{i,r}^{(\text{ite})}}; \qquad (6)
$$

$$\pi_k^{(\text{ite}+1)} = \pi_k^{(\text{ite})} + t^{(\text{ite})}\left(\frac{n_k^{(\text{ite})}}{n} - \pi_k^{(\text{ite})}\right); \tag{7}$$

$$[\boldsymbol{P}_k]_{m_1,m_2}^{(\text{ite}+1)} = \begin{cases} \frac{n_k^{(\text{ite})}\langle[\tilde{\boldsymbol{y}}_.^{(\text{ite})}]_{k,m},[\boldsymbol{\Phi}_k]_{m,.}^{(\text{ite})}[\tilde{\boldsymbol{x}}^{(\text{ite})}_.]_{k,.}\rangle+\sqrt{\Delta_{k,m}}}{2n_k^{(\text{ite})}\|[\tilde{\boldsymbol{y}}_.^{(\text{ite})}]_{k,m}\|_2^2} & \text{if } m_1 = m_2 = m; \\ 0 & \text{elsewhere}; \end{cases} \tag{8}$$

$$[\boldsymbol{\Phi}_k]_{m,j}^{(\text{ite}+1)} = \begin{cases} \frac{-[\boldsymbol{S}_k]_{j,m}^{(\text{ite})}+n\lambda\pi_k^{(\text{ite})}}{\|[\tilde{\boldsymbol{x}}^{(\text{ite})}_.]_{k,j}\|_2^2} & \text{if } [\boldsymbol{S}_k]_{j,m}^{(\text{ite})} > n\lambda\pi_k^{(\text{ite})}; \\ -\frac{[\boldsymbol{S}_k]_{j,m}^{(\text{ite})}+n\lambda\pi_k^{(\text{ite})}}{\|[\tilde{\boldsymbol{x}}^{(\text{ite})}_.]_{k,j}\|_2^2} & \text{if } [\boldsymbol{S}_k]_{j,m}^{(\text{ite})} < -n\lambda\pi_k^{(\text{ite})}; \\ 0 & \text{else }; \end{cases} \tag{9}$$

In this case, the EM algorithm corresponds to alternating between the E-step [computation of (6)] and the M-step [computation of (7), (8) and (9)].

Remark that to approximate the MLE under rank constraint, we use an easier EM algorithm which is described in details in the Appendix. In the E-step, we compute the a posteriori probability of each observation to belong to each cluster according to the current estimations. In the M-step, we consider each observation within its estimated cluster, and we consider the linear regression estimators under rank constraint by keeping the biggest singular values.

2.3.2 Initialization and stopping rules

We initialize the clustering process with the k-means algorithm on couples $(\boldsymbol{y}_i, \boldsymbol{x}_i) \in \mathbb{R}^q \times \mathbb{R}^p$, for all $i \in \{1, \ldots, n\}$. According to this clustering, we compute the linear regression estimators in each class. Then, we run a small number of times the EM-algorithm, repeat this initialization many times and keep the one which corresponds to the highest log-likelihood.

To stop the algorithm, we propose to run it a minimum number of times, and to specify a maximum number of iterations to make sure that it will stop. Between these two bounds, we stop the running if a relative criteria on the log-likelihood is small enough and if a relative criteria on the parameters is small enough too. Those criteria are adapted from Städler et al. (2010).

By those rules, we are trying to attain a local optimum as close as possible to a global optimum of the likelihood function.

2.3.3 Numerical convergence of this algorithm

We address here the convergence properties of the generalized EM algorithm described in Sect. 2.3.1. Although convergence to stationary points has been established for the EM algorithm (see Wu 1983), it is not true without conditions which are hard to verify for a generalized EM algorithm.

The results that we prove are quite similar to those provided by Städler et al. (2010), because the algorithms are similar. We refer the interested reader to this article for proofs of the following results when $q = 1$. The same ideas of proof are working for any values of q.

Proposition 1 *The algorithm described in Sect. 2.3.1 has the descent property: for $ite \in \mathbb{N}^*$,*

$$-\frac{1}{n} l_\lambda(\theta^{(ite+1)}, \boldsymbol{y}_1, \boldsymbol{x}_1, \dots, \boldsymbol{y}_n, \boldsymbol{x}_n) \leq -\frac{1}{n} l_\lambda(\theta^{(ite)}, \boldsymbol{y}_1, \boldsymbol{x}_1, \dots, \boldsymbol{y}_n, \boldsymbol{x}_n)$$

Proposition 1 is clear by construction of $\theta^{(\text{ite}+1)}$, the M-step of the algorithm being a coordinate-wise minimization.

Proposition 2 *Assume that $\boldsymbol{y}_i \neq 0$ for all $i \in \{1, \dots, n\}$. Then, for $\lambda > 0$, the function $\theta \mapsto l_\lambda(\theta, \boldsymbol{y}_1, \boldsymbol{x}_1, \dots, \boldsymbol{y}_n, \boldsymbol{x}_n)$ defined in (2) is bounded from above for all values $\theta \in \Theta_K$.*

This is true because we penalize the log-likelihood by $\boldsymbol{\Phi} = (\boldsymbol{\Phi}_1, \dots, \boldsymbol{\Phi}_K)$, where $\boldsymbol{\Phi}_k = \boldsymbol{P}_k B_k$, and then small variances are penalized also. This is not the case for the unpenalized criteria, which is unbounded if the variance of a coordinate of a variable tends to 0 (see McLachlan and Peel 2004 for more details).

Corollary 1 *For the algorithm described in Sect. 2.3.1,*

$$-\frac{1}{n} l_\lambda(\theta^{(ite)}, \boldsymbol{y}_1, \boldsymbol{x}_1, \dots, \boldsymbol{y}_n, \boldsymbol{x}_n)$$

decreases monotonically to some value $\bar{l} > -\infty$.

According to this corollary, we know that the algorithm converges to some finite value, depending on initial values.

Theorem 1 *For the algorithm described in Sect. 2.3.1, every cluster point $\bar{\theta} \in \Theta_K$ of the sequence $\left\{\theta^{(ite)}, ite = 0, 1, \dots\right\}$ generated by the algorithm is a stationary point of the function*

$$\theta \mapsto l_\lambda(\theta, \boldsymbol{y}_1, \boldsymbol{x}_1, \dots, \boldsymbol{y}_n, \boldsymbol{x}_n).$$

This theorem proves that the algorithm converges to a stationary point. We refer to Tseng (2001) for the definition of a stationary point for non-differentiable functions.

Corollary 1 is deduced from Proposition 1 and Proposition 2. Theorem 1 is more difficult to prove, and we refer to Städler et al. (2010) for a proof when $q = 1$.

2.4 Variable selection and model collection

We deal with high-dimensional data where we observe a small number of target values and we have to estimate many coefficients. Then, we have to focus on indices that are relevant for the clustering. The notion of irrelevant indices has to be defined.

Definition 1 A couple $(\boldsymbol{y}_m, \boldsymbol{x}_j)$ is said to be *irrelevant* for the clustering if $[\boldsymbol{\Phi}_1]_{m,j} = \dots = [\boldsymbol{\Phi}_K]_{m,j} = 0$, which means that the variable $\boldsymbol{x}_j$ does not explain the variable $\boldsymbol{y}_m$ for the clustering process. We also say that the indices (m, j) are irrelevant if the couple $(\boldsymbol{y}_m, \boldsymbol{x}_j)$ is irrelevant. A *relevant* couple is a couple which is not irrelevant: at least in one cluster k, the coefficient $[\boldsymbol{\Phi}_k]_{m,j}$ is not equal to zero. We denote by J the set of indices (m, j) of relevant couples $(\boldsymbol{y}_m, \boldsymbol{x}_j)$.

Remark that $J \subset \{1, \dots, q\} \times \{1, \dots, p\}$. We denote by ${}^c J$ the complement of J in $\{1, \dots, q\} \times \{1, \dots, p\}$. For all $k \in \{1, \dots, K\}$, we denote by $\boldsymbol{\Phi}_k^{[J]}$ the matrix of size $q \times p$ with 0 on the set ${}^c J$. We also define by $\mathcal{H}_{(K,J)}$ the model with K components and where J is the set of relevant indices:

$$\mathcal{H}_{(K,J)} = \left\{ h_\theta^{(K,J)}(\boldsymbol{y}|\boldsymbol{x}) = \sum_{k=1}^{K} \frac{\pi_k \det(\boldsymbol{P}_k)}{(2\pi)^{q/2}} \exp\left(-\frac{(\boldsymbol{P}_k \boldsymbol{y} - \boldsymbol{\Phi}_k^{[J]} \boldsymbol{x})^t (\boldsymbol{P}_k \boldsymbol{y} - \boldsymbol{\Phi}_k^{[J]} \boldsymbol{x})}{2} \right), \right.$$
$$\theta = (\pi_1, \dots, \pi_K, \boldsymbol{\Phi}_1^{[J]}, \dots, \boldsymbol{\Phi}_K^{[J]}, \boldsymbol{P}_1, \dots, \boldsymbol{P}_K) \in \Theta_{(K,J)}$$
$$\left. = \Pi_K \times \left(\mathbb{R}^{q \times p}\right)^K \times \left(\mathbb{D}_q^{++}\right)^K \right\}. \tag{10}$$

We construct a collection of models by varying the number of components K and the indices set of relevant couples J.

The subset J is constructed with the Lasso estimator defined in (2). Nevertheless, to be consistent with the definition of relevant couples, the Group-Lasso estimator could be preferred, where coefficients $\{[\boldsymbol{\Phi}_1]_{m,j}, \dots, [\boldsymbol{\Phi}_K]_{m,j}\}$ are grouped. We focus here on the Lasso estimator, but the Group-Lasso approach is described in the Appendix. It gives mainly the same results, but the Lasso estimator permits to consider also isolated irrelevant indices.

3 Two procedures

The goal of our procedures is, given a sample $((\boldsymbol{y}_1, \boldsymbol{x}_1), \dots, (\boldsymbol{y}_n, \boldsymbol{x}_n)) \in (\mathbb{R}^q \times \mathbb{R}^p)^n$, to cluster individuals. Thus, we have to estimate, according to $\mathcal{H}_{(K,J)}$ defined in (10), the number of clusters K, the relevant indices set J, and the parameter θ. We want to take advantage of the sparsity property of the ℓ_1-penalization to perform automatic index selection and to deduce J. Then, we compute another estimator restricted on coordinates with relevant indices, which will work better because it is no longer an high-dimensional issue. Thus, we avoid shrinkage problems due to the Lasso estimator. The first procedure takes advantage of the MLE, whereas the second one takes into account the matrix structure of $\boldsymbol{\Phi}$ through a low rank estimation.

3.1 Lasso-MLE procedure

This procedure is decomposed into three main steps: we construct a collection of models, then in each model we compute the MLE and finally we select the best model among the collection of models.

The first step consists in constructing a collection of models $\{\mathcal{H}_{(K,J)}\}_{(K,J)\in\mathcal{M}}$ in which $\mathcal{H}_{(K,J)}$ is defined by Eq. (10), and the collection of models is indexed by

$\mathcal{M} = \mathcal{K} \times \mathcal{J}$. We denote by $\mathcal{K} \subset \mathbb{N}^*$ the admitted number of clusters. We assume that we could bound $\mathcal{K}$ without loss of generality. We also denote $\mathcal{J} \subset \mathcal{P}(\{1, \ldots, q\} \times \{1, \ldots, p\})$ the set of admitted relevant indices set.

Fix $K \in \mathcal{K}$. To detect the relevant indices and construct the set $J \in \mathcal{J}$, we penalize the log-likelihood by an ℓ_1-penalty on the mean parameters proportional to $||\mathbf{\Phi}_k||_1 = \sum_{j=1}^{p} \sum_{m=1}^{q} |[\mathbf{\Phi}_k]_{m,j}|$. In ℓ_1-procedures, the choice of the regularization parameter is often difficult: fixing the number of components $K \in \mathcal{K}$, we propose to construct a data-driven grid G_K of regularization parameters by using the updating formulae [(6), (7), (8) and (9)] of the parameters in the EM algorithm. We give a formula for λ, the regularization parameter, depending on which coefficients we want to estimate by zero, for all $k \in \{1, \ldots, K\}$, $j \in \{1, \ldots, p\}$, $z \in \{1, \ldots, q\}$:

$$[\mathbf{\Phi}_k]_{m,j} = 0 \Leftrightarrow [\boldsymbol{\lambda}_k]_{j,m} = \frac{|[\boldsymbol{S}_k]_{j,m}|}{n\pi_k};$$

where $[\boldsymbol{S}_k]_{j,m}$ is defined by (4). Then, we define the data-driven grid for the regularization parameter by

$$G_K = \left\{[\boldsymbol{\lambda}_k]_{j,m}, k \in \{1, \ldots, K\}, j \in \{1, \ldots, p\}, m \in \{1, \ldots, q\}\right\}. \tag{11}$$

We approximate it by MLE.

Then, for each $\lambda \in G_K$, we compute the Lasso estimator defined by

$$\hat{\theta}^{\text{Lasso}}(\lambda) = \underset{\theta \in \Theta_K}{\operatorname{argmin}} \left\{ -\frac{1}{n} \sum_{i=1}^{n} \log(h_\theta^K(\boldsymbol{y}_i|\boldsymbol{x}_i)) + \lambda \sum_{k=1}^{K} \pi_k ||\mathbf{\Phi}_k||_1 \right\}.$$

For a fixed number of mixture components $K \in \mathcal{K}$ and a regularization parameter $\lambda \in G_K$, we use an EM algorithm to approximate this estimator. Then, for each $K \in \mathcal{K}$ and for each $\lambda \in G_K$, we construct the relevant indices set $J_{K,\lambda}$. We denote by $\mathcal{J}$ the collection of the relevant indices set $J_{K,\lambda}$ by varying $\lambda \in G_K$ and $K \in \mathcal{K}$:

$$\mathcal{J} = \cup_{K \in \mathcal{K}} \cup_{\lambda \in G_K} J_{K,\lambda}.$$

The second step consists in approximating the MLE

$$\hat{h}^{(K,J)} = \underset{h \in \mathcal{H}_{(K,J)}}{\operatorname{argmin}} \left\{ -\frac{1}{n} \sum_{i=1}^{n} \log(h(\boldsymbol{y}_i|\boldsymbol{x}_i)) \right\};$$

using the EM algorithm for each model $(K, J) \in \mathcal{M}$.

The third step is devoted to model selection. Rather than selecting the regularization parameter, we select the refitted model. Instead of using an asymptotic criterion, as BIC or AIC, we use the slope heuristic introduced by Birgé and Massart (2007), which is a non-asymptotic criterion for selecting a model among a collection of models. Let us explain briefly how it works. Firstly, models are grouped according to their dimension D, to obtain a collection of models $\{\mathcal{H}_D\}_{D \in \mathcal{D}}$. The model dimension is the number of parameters estimated in this model. For each dimension D, let $\hat{h}_D$ be the estimator which maximizes the likelihood function among the estimators associated to a model of dimension D. Also, the function $D/n \mapsto 1/n \sum_{i=1}^n \log(\hat{h}_D(\boldsymbol{y}_i|\boldsymbol{x}_i))$ has a linear behavior for large dimensions. We estimate the slope, denoted by $\hat{\kappa}$, which will be used to calibrate the penalty. The minimizer $\hat{D}$ of the penalized criterion $-1/n \sum_{i=1}^n \log(\hat{h}_D(\boldsymbol{y}_i|\boldsymbol{x}_i)) + 2\hat{\kappa}D/n$ is determined, and the model selected is $(K_{\hat{D}}, J_{\hat{D}})$. Remark that $D_{K,J} = K(|J| + q + 1) - 1$ is the dimension associated to the model $\mathcal{H}_{(K,J)}$.

Note that the model is selected after refitting, which avoids issue of regularization parameter selection. For an oracle inequality to justify the slope heuristic used here, see Devijver (2015).

3.2 Lasso-Rank procedure

The previous procedure does not take into account the multivariate structure, and therefore we propose a second procedure that takes into account this structure. For each model belonging to the collection $\mathcal{H}_{(K,J)}$, a subcollection is constructed, varying the rank of $\boldsymbol{\Phi}$. Let us describe this procedure.

As in the Lasso-MLE procedure, we first construct a collection of models through varying the sparsity by varying the regularization parameter of the Lasso estimator. Remark that we focus here on column sparsity to keep the matrix structure.

The second step consists in constructing a subcollection of models with rank sparsity, denoted by

$$\{\check{\mathcal{H}}_{(K,J,R)}\}_{(K,J,R)\in\check{\mathcal{M}}}.$$

The model $\check{\mathcal{H}}_{(K,J,R)}$ has K components, J is the set of relevant indices and R is the vector of the ranks of regression coefficients in each group:

$$\check{\mathcal{H}}_{(K,J,R)} = \left\{ \boldsymbol{h}_\theta^{(K,J,R)}(\boldsymbol{y}|\boldsymbol{x}) \right\} \tag{12}$$

where

$$\begin{aligned}
h_\theta^{(K,J,R)}(\boldsymbol{y}|\boldsymbol{x}) &= \sum_{k=1}^K \frac{\pi_k \det(\boldsymbol{P}_k)}{(2\pi)^{q/2}} \exp\left(-\frac{(\boldsymbol{P}_k \boldsymbol{y} - (\boldsymbol{\Phi}_k^{R_k})^{[J]}\boldsymbol{x})^t (\boldsymbol{P}_k \boldsymbol{y} - (\boldsymbol{\Phi}_k^{R_k})^{[J]}\boldsymbol{x})}{2} \right);\\
\theta &= (\pi_1, \ldots, \pi_K, (\boldsymbol{\Phi}_1^{R_1})^{[J]}, \ldots, (\boldsymbol{\Phi}_K^{R_K})^{[J]}, \boldsymbol{P}_1, \ldots, \boldsymbol{P}_K) \in \Theta_{(K,J,R)}\\
\Theta_{(K,J,R)} &= \Pi_K \times \Psi_{(K,J,R)} \times \left(\mathbb{R}_+^q\right)^K;\\
\Psi_{(K,J,R)} &= \Big\{ \left((\boldsymbol{\Phi}_1^{R_1})^{[J]}, \ldots, (\boldsymbol{\Phi}_K^{R_K})^{[J]} \right) \in \left(\mathbb{R}^{q\times p}\right)^K \Big| \operatorname{Rank}(\boldsymbol{\Phi}_k)\\
&\quad = R_k \text{ for all } k \in \{1, \ldots, K\} \Big\};
\end{aligned}$$

and $\check{\mathcal{M}} = \mathcal{K} \times \mathcal{J} \times \mathcal{R}$. We denote by $\mathcal{K} \subset \mathbb{N}^*$ the admitted number of clusters, $\mathcal{J}$ a random collection of subsets of $\{1, \ldots, p\}$ constructed by the Lasso estimator and $\mathcal{R}$ the set of vectors of size $K \in \mathcal{K}$ with rank values for each regression matrix. We compute the MLE under the rank constraint with an EM algorithm. Indeed, the estimator of $\mathbf{\Phi}_k$, for the cluster k, is constrained to have a rank equals to R_k, by keeping only the R_k largest singular values. More details are given in the Appendix. It leads to an estimator of the mean with row sparsity and low rank for each model. Then, a model is selected using the slope heuristic. This step is justified theoretically in Devijver (2015).

3.3 Some comments

Those two procedures have been implemented in Matlab, with Benjamin Auder and the Matlab code is available. Note that we run the EM algorithm several times: once to construct the regularization grid, twice for each regularization parameter for the Lasso-MLE procedure (once to approximate the Lasso estimator, and once to refit it with the MLE) and more times for the Lasso-Rank procedure (the ranks vector varies also). If we look at every regularization parameter in the grid defined in (11), there are $K \times p \times q$ values and then we compute the EM algorithm $2 \times K \times p \times q + 1$ times, which could be large. Even if each EM algorithm is fast (implemented in C), it should be repeated a lot of time, which is time-consuming. We propose to the user to select relevant regularization parameters: either a regular subcollection of G_K, to get various sparsities, or focusing on the large values of regularization parameters, to get sparse solutions.

4 Illustrative example

We illustrate our procedures on five different simulated datasets, adapted from Städler et al. (2010). Firstly, we present the models used in these simulations. Then, we validate numerically each step and we finally compare the results of our procedures with other methods. Remark that we consider here some examples to illustrate our methods, but not a complete analysis. We highlight some issues which seem to be important. Moreover, we do not illustrate the one-component case, as we are focusing on clustering. If, on some dataset, we are not convinced by the clustering, we add to the collection of models some models with one component, more or less sparse, using the same pattern (computing the Lasso estimator to get the relevant indices for various regularization parameters and refitting parameters with the MLE, under rank constraint or not) and then we select a model among this collection of linear and mixture models.

4.1 The model

Let $\boldsymbol{x} \in (\mathbb{R}^p)^n$ be a sample of size n distributed according to p-multivariate standard Gaussian with $p = 10$ or $p = 30$. We consider a mixture of two components. Besides,

Table 1 Description of the different models for $K = 2$ classes

	Model 1	Model 2	Model 3	Model 4	Model 5
n	2000	100	100	100	50
K	2	2	2	2	2
p	10	10	10	10	30
q	10	10	10	10	5
$[\mathbf{B}_1]_{m,j}$ for $(m, j) \in J$	3	3	3	5	3
$[\mathbf{B}_2]_{m,j}$ for $(m, j) \in J$	−2	−2	−2	3	−2
σ^2	1	1	3	1	1
SNR	3.6	3.6	1.88	7.8	3.6

we fix the number of relevant indices to be 4 in each cluster. More precisely, the first four variables of $\boldsymbol{Y}$ are explained respectively by the first four variables of $\boldsymbol{x}$. Fix $\pi = \left(\frac{1}{2}, \frac{1}{2}\right)$ and $\boldsymbol{P}_k = \sigma I_q$ for all $k \in \{1, 2\}$, where I_q denotes the identity matrix of dimension q. Take a sample of $\boldsymbol{Y}$ according to a q multivariate Gaussian mixture with $q = 5$ or $q = 10$, with mean $\mathbf{B}_k\mathbf{x}$ and with covariance matrix $\boldsymbol{\Sigma}_k = (\boldsymbol{P}_k^t \boldsymbol{P}_k)^{-1} = \sigma^2 I_q$, for the cluster k.

The difficulty of the clustering is partially controlled by the signal-to-noise ratio (denoted SNR). In this context, we extend the natural idea of the SNR with the following definition, where $\text{Tr}(A)$ denotes the trace of the matrix A,

$$\text{SNR} = \frac{\text{Tr}(\text{Var}(\boldsymbol{Y}))}{\text{Tr}(\text{Var}(\boldsymbol{Y}|\mathbf{B}_k = 0 \text{ for all } k \in \{1, \ldots, K\}))}.$$

Remark that it only controls the distance between the signal with or without the noise and not the distance between signals across clusters.

We consider five different models, varying n, the SNR and the distance between the clusters. Details are available in Table 1.

We run our procedures with the number of components varying in $\mathcal{K} = \{2, \ldots, 5\}$.

Model 1 serves for illustrating our procedures in low dimensional models. Moreover, it is chosen in the next section to illustrate each step of the procedure (variable selection, construction of models and model selection). Model 5 is considered because it is high-dimensional. Model 2 is easier than the others, because clusters are not so close to each other according to the noise. Model 3 is similar to Models 1 and 2, but n is small and the noise is more important. We will see that the clustering is more difficult in this case. Model 4 has a larger SNR, nevertheless, the problem of clustering is difficult, because $\mathbf{B}_k$'s are close to each other.

Our procedures are run 20 times and we compute statistics over those 20 simulations. It is a small number of simulations, but the whole procedure is time-consuming and results are convincing enough.

For the initialization step, we repeat 50 times the initialization and keep the one which maximizes the log-likelihood function after 10 iterations. Those choices are size-dependent. We have done a numerical study which is not reported here to fit them.

4.2 Sparsity and model selection

To illustrate both procedures, all analyses made in this section are done from Model 1, since the choice of each step is clear.

Firstly, we compute the grid of regularization parameters. More precisely, each regularization parameter is computed from maximum likelihood estimations (using EM algorithm) and gives an associated sparsity (computed by the Lasso estimator, using again the EM algorithm). In Figs. 1 and 2, the collection of relevant indices selected by this grid is plotted.

Firstly, note that the number of relevant indices selected by the Lasso decreases when the regularization parameter increases. Moreover, we analyze which indices are selected, that is to say if we select true relevant or false relevant indices. If the

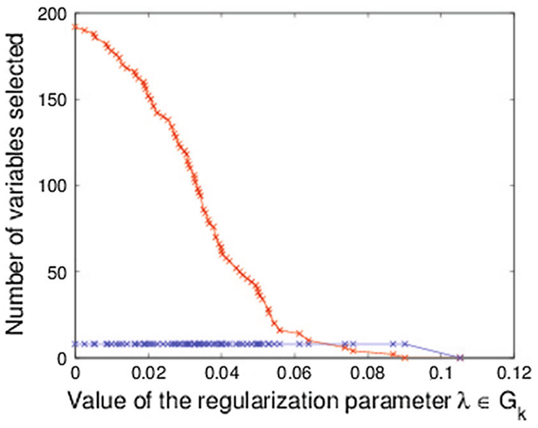


Fig. 1 For one simulation, the number of false relevant (in *red color*) and true relevant (in *blue color*) indices generated by the Lasso, by varying the regularization parameter λ in the grid G_2

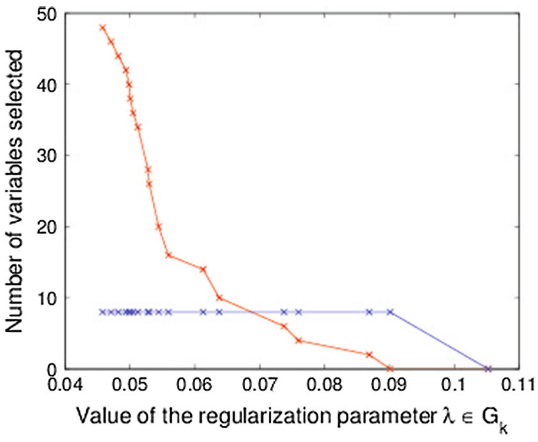


Fig. 2 For one simulation, zoom in on the number of false relevant (in *red color*) and true relevant (in *blue color*) indices generated by the Lasso, by varying the regularization parameter λ around the interesting values

regularization parameter is not too large, the true relevant indices are selected. Even more, if the regularization parameter is well-chosen, we select only the true relevant indices. In our example, we remark that if $\lambda = 0.09$, we have selected exactly the true relevant indices. This grid construction seems to be well-chosen according to our simulations.

From this variable selection, each procedure (Lasso-MLE or Lasso-Rank) leads to a collection of models, by varying the sparsity thanks to the grid of regularization parameters and the number of components.

Within this collection of models, we select a model using the slope heuristic criterion.

We want to select the best model by optimizing a penalized criterion. This penalty is computed by performing a linear regression on the couples of points $\{(D/n; -1/n \sum_{i=1}^{n} \log(\hat{h}_D(\boldsymbol{y}_i|\boldsymbol{x}_i)))\}$. The slope $\hat{\kappa}$ allows us to have access to the best model, the one with dimension $\hat{D}$ minimizing

$$-\frac{1}{n}\sum_{i=1}^{n} \log(\hat{h}_D(\boldsymbol{y}_i|\boldsymbol{x}_i)) + 2\hat{\kappa}\frac{D}{n}.$$

In practice, we have to look if couples of points have a linear behavior. For each procedure, we construct a different collection of models and we have to justify this behavior. Figures 3 and 4 represent the log-likelihood in function of the dimension of the models, for collections of models constructed respectively by the Lasso-MLE procedure and by the Lasso-Rank procedure.

The couples are plotted by points, whereas the estimated slope is specified by a dotted line. We observe more than a line (4 for the Lasso-MLE procedure, more for the Lasso-Rank procedure). This phenomenon is explained by a linear behavior for each model subcollection, when we fix the number of classes and the rank. However, the linear behavior is almost the same and leads to select the same penalty coefficient. Nevertheless, slopes are almost the same and select the same model. In practice, we estimate the slope with the Capushe package described in Baudry et al. (2012).

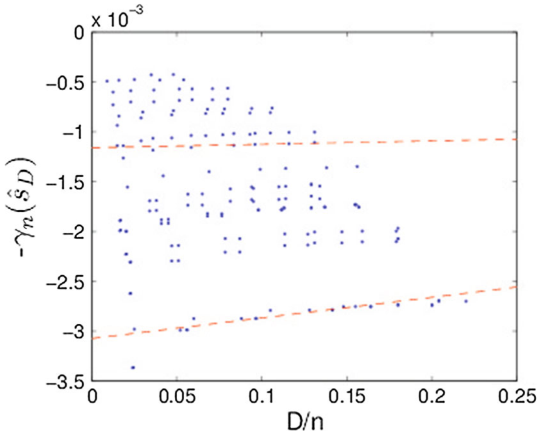


Fig. 3 For one simulation, slope graph get by our Lasso-Rank procedure. For large dimensions, we observe a linear part

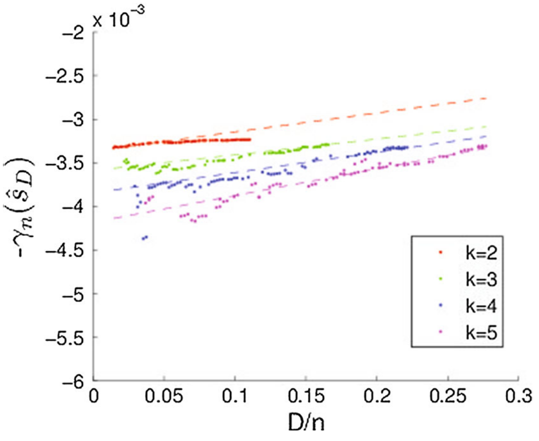


Fig. 4 For one simulation, slope graph get by our Lasso-MLE procedure. For large dimensions, we observe a linear part

4.3 Assessment

We compare our procedures to three other procedures on the simulated models.

Firstly, let us give some remarks about Model 1. For each procedure, we get a good clustering and a very low Kullback–Leibler divergence. Indeed, the sample size is large and estimations are good. That is the reason why, in this subsection, we focus on Models 2, 3 and 4.

To compare our procedures with others, the Kullback–Leibler divergence of the density selected from the true density and the ARI (the Adjusted Rand Index, measuring the similarity between two data clusterings, knowing that the closer the ARI to 1, the more similar the two partitions) are computed, and we note which indices are selected and how many clusters are selected. For more details on the ARI, see Hubert and Arabie (1985).

From the Lasso-MLE collection of models, we consider two models, to compare our procedures with. We compute the oracle model (the model which minimizes the Kullback–Leibler divergence of it from the true density) and the model which is selected by the BIC criterion instead of the slope heuristic. According to the oracle model, we know how good we could get for this collection of models for the Kullback–Leibler divergence and how this model, is good for clustering data.

The third procedure we compare with is the MLE, assuming that we know how many clusters there are, fixed to 2. We use this procedure to show that variable selection is needed.

In each case, we use the MAP principle to get the clustering.

We do not plot the Kullback–Leibler divergence for the MLE procedure, because values are too high and make the boxplots unreadable.

From Figs. 5 and 6 we see that, for Model 2, the Kullback–Leibler divergence is small and the ARI is close to 1, except for the MLE procedure. Boxplots are still readable with those values, but it is important to highlight that variable selection is needed, even in a model with reasonable dimension. The collections of models are then well constructed. Model 3 is more difficult, because the noise is higher. That is why

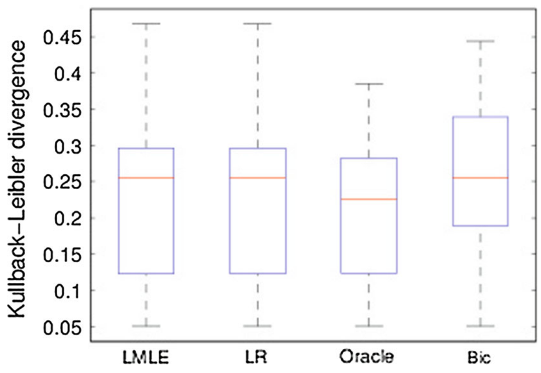


Fig. 5 *Boxplots* of the Kullback–Leibler divergence of the density selected from the true density over the 20 simulations, for the Lasso-MLE procedure (LMLE), the Lasso-Rank procedure (LR), the oracle (Oracle), the BIC estimator (BIC) for Model 2

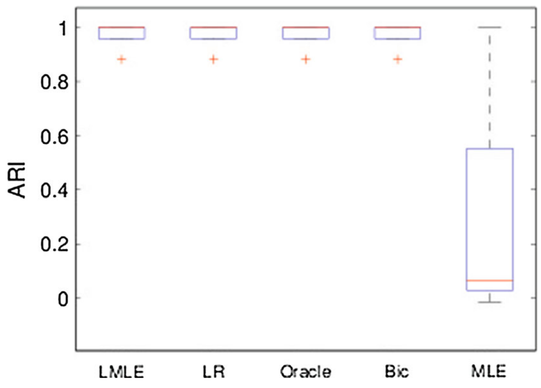


Fig. 6 *Boxplots* of the ARI over the 20 simulations, for the Lasso-MLE procedure (LMLE), the Lasso-Rank procedure (LR), the oracle (Oracle), the BIC estimator (BIC) and the MLE (MLE) for Model 2

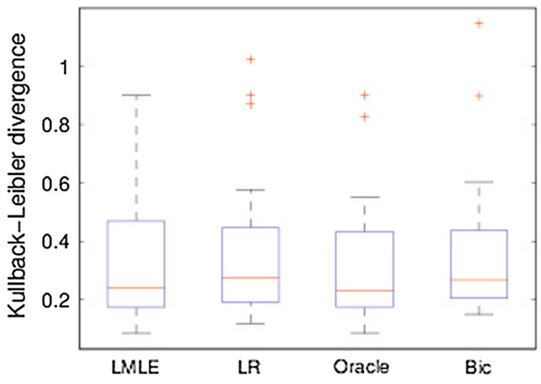


Fig. 7 *Boxplots* of the Kullback–Leibler divergence of the density selected from the true density over the 20 simulations, for the Lasso-MLE procedure (LMLE), the Lasso-Rank procedure (LR), the oracle (Oracle), the BIC estimator (BIC) for Model 3

results, summarized in Figs. 7 and 8, are not as good as for Model 2. Nevertheless, our procedures lead to the best ARI and the Kullback–Leibler divergences are close to the one of the oracle. Similar remarks are made for the results for Model 4. In this study, means are closer, according to the noise. Results are summarized in Figs. 9 and 10. Model 5 is a high-dimensional model. Models selected by the BIC criterion are bad, in comparison with models selected by our procedures, or in comparison with the oracle

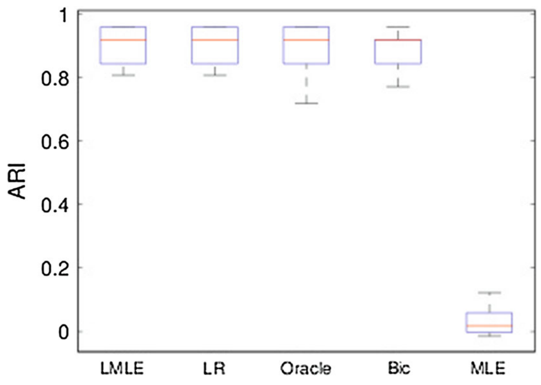


Fig. 8 *Boxplots* of the ARI over the 20 simulations, for the Lasso-MLE procedure (LMLE), the Lasso-Rank procedure (LR), the oracle (Oracle), the BIC estimator (BIC) and the MLE (MLE) for Model 3

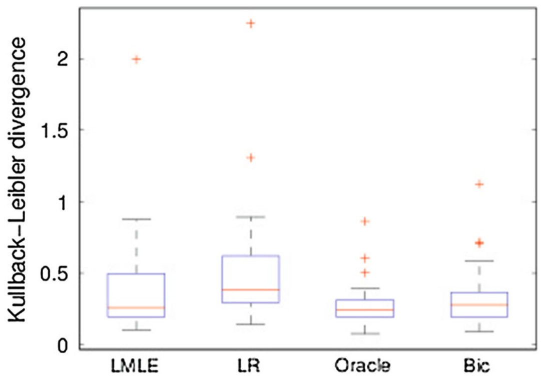


Fig. 9 *Boxplots* of the Kullback–Leibler divergence of the density selected from the true density over the 20 simulations, for the Lasso-MLE procedure (LMLE), the Lasso-Rank procedure (LR), the oracle (Oracle), the BIC estimator (BIC) for Model 4

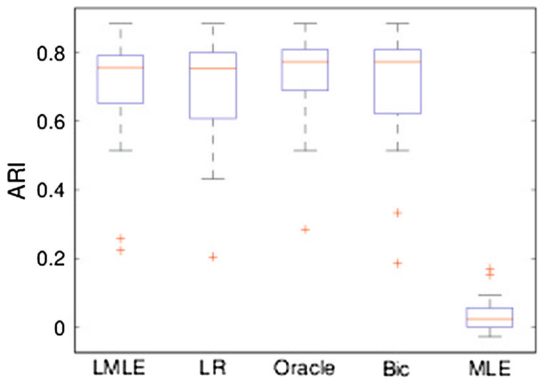


Fig. 10 *Boxplots* of the ARI over the 20 simulations, for the Lasso-MLE procedure (LMLE), the Lasso-Rank procedure (LR), the oracle (Oracle), the BIC estimator (BIC) and the MLE (MLE) for Model 4

models. They are bad for estimation, according to the Kullback–Leibler divergence boxplots in Fig. 11, but also for clustering, according to Fig. 12. Our models are not as well as constructed previously, due to the high-dimensional context. It explains the high Kullback–Leibler divergence. However, our performances in clustering are good.

Note that the Kullback–Leibler divergence is smaller for the Lasso-MLE procedure, thanks to the maximum likelihood refitting. Moreover, the true model has no matrix

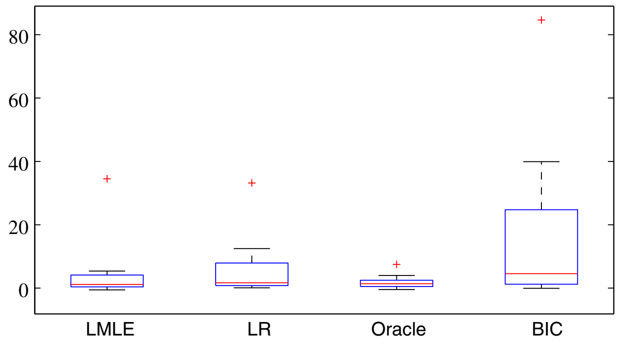


Fig. 11 *Boxplots* of the Kullback–Leibler divergence of the density selected from the true density over the 20 simulations, for the Lasso-MLE procedure (LMLE), the Lasso-Rank procedure (LR), the oracle (Oracle), the BIC estimator (BIC) for Model 5

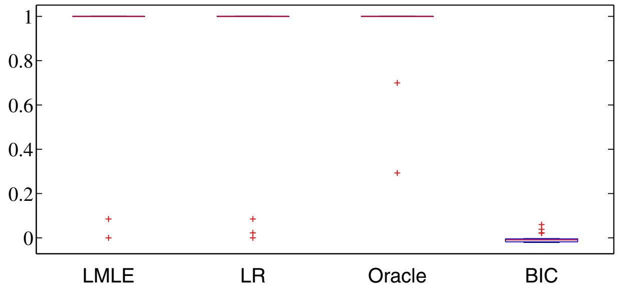


Fig. 12 *Boxplots* of the ARI over the 20 simulations, for the Lasso-MLE procedure (LMLE), the Lasso-Rank procedure (LR), the oracle (Oracle), the BIC estimator (BIC) and the MLE (MLE) for Model 5

Table 2 Mean number {TR, FR} of true relevant and false relevant indices over the 20 simulations for each procedure, for models 2, 3 and 4

	Model 2		Model 3		Model 4	
Procedure	TR	FR	TR	FR	TR	FR
Lasso-MLE	8 (0)	2.2 (6.9)	8(0)	4.3 (28.8)	8(0)	2(13,5)
Lasso-Rank	8 (0)	24 (0)	8 (0)	24(0)	8(0)	24(0)
Oracle	8 (0)	1.5 (3.3)	7.8 (0.2)	2.2 (11.7)	8 (0)	0.8 (2.6)
BIC estimator	8 (0)	2.6 (15.8)	7.8 (0.2)	5.7 (64.8)	8 (0)	2.6 (11.8)

The standard deviations are put into parenthesis

structure. If we compare with the MLE, where we do not use the sparsity hypothesis, we conclude that estimations are not satisfactory, due to the high-dimensional context. The Lasso-MLE procedure, the Lasso-Rank procedure and the BIC model work almost as well as the oracle, which means that models are well selected.

In Table 2, we summarize results about indices selection. For each model and for each procedure, we compute how many true relevant and false relevant indices are selected.

The true model has 4 relevant indices, which are always recognized. The Lasso-MLE has less false relevant indices than the others, which means that the true structure was found. The Lasso-Rank has 24 false relevant indices, because of the matrix structure: the true rank in each component was 4, then the estimator restricted on relevant indices is a 4×4 matrix and we get 12 false relevant indices in each component. Nevertheless, it is the smallest matrix if select every relevant indices. The BIC estimator and the oracle have a large variability for the false relevant indices.

Remark that all procedures have selected the true number of components.

Thanks to the MLE, the first procedure has good performance in estimation (better than the second one). However, depending on the data, the second procedure could be more attractive. If there is a matrix structure, for example if most of the variation of the response $\boldsymbol{Y}$ is caught by a small number of linear combinations of the predictors, the second procedure will work better.

5 Functional data

A main aspect of our methods is the fact that they can be applied to functional data as well. Indeed, in different fields of applications, the considered data are functions. The functional data analysis has been popularized first by Ramsay and Silverman (2005). This book gives a description of the main tools to analyze functional data. We also refer to Ferraty and Vieu (2006). In a density estimation framework, we refer to Gareth and Sugar (2003), who used a model-based approach for clustering functional data. About functional regression, however, the main part of the existing literature is focused on $\boldsymbol{Y}$ scalar and $\boldsymbol{x}$ functional. For example, we refer to Zhao et al. (2012) for using wavelet basis in linear model, Yao et al. (2011) for functional mixture regression, or Ciarleglio and Ogden (2014) for using wavelet basis in functional mixture regression. In this section, we focus on $\boldsymbol{Y}$ and $\boldsymbol{x}$ both functional. In this regression context, we are interested in clustering: it leads to identifying individuals involved in the same relation between $\boldsymbol{Y}$ and $\boldsymbol{x}$. Denote that, with functional data, we have to denoise and to smooth signals to remove the noise and capture only the important patterns in the data. Here, we explain how our procedures are applied in this context. Note that we could apply our procedures with scalar response and functional regressor, or, on the contrary, with functional response and scalar regressor. We explain how our procedures work in the more general case. Remark that we focus here on the wavelet basis, to take advantage of the time-scale decomposition, but the same analysis is available on Fourier basis or splines.

5.1 Functional regression model

We observe deterministic functional predictors $(f_i)_{1 \leq i \leq n}$ defined on I_F and a sample $(g_i)_{1 \leq i \leq n}$ of functional response associated with the random variables G_i defined on I_G. If the observation $i \in \{1, \dots, n\}$ originates from the cluster k, the model assumes that there exists a function $\boldsymbol{B}_k : I_F \times I_G \to \mathbb{R}$ such that, for $t \in I_G$,

$$G_i(t) = \int_{I_F} f_i(u)\mathbf{B}_k(u,t)du + \varepsilon_i(t), \tag{13}$$

where ε_i is a residual function. This linear model, restricted on each cluster, is introduced in Ramsay and Silverman (2005).

Unfortunately, we observe data on discretized time, and we have access to $((f_i(t_j))_{1\leq j\leq p})_{1\leq i\leq n}$ and $((g_i(t_m))_{1\leq m\leq q})_{1\leq i\leq n}$.

If we assume that, for all $(t_m)_{1\leq m\leq q}$, for all $i \in \{1,\ldots,n\}$, $\boldsymbol{\varepsilon}_i(t_m) \sim \mathcal{N}(0, \boldsymbol{\Sigma}_k)$, Model (13) is an integrated version of Model (1). Depending on the cluster k, the linear relation of $\boldsymbol{G}_i$ with respect to $\boldsymbol{f}_i$ is described by the function $\mathbf{B}_k$.

5.2 Two procedures to deal with functional data

5.2.1 Projection onto a wavelet basis

We project the functional data onto some basis, to get data as described in the Gaussian mixture regression model (1). In this article, we deal with wavelet basis, because they represent localized features of functions in a sparse way. If the coefficient matrices $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$ associated to function f_i and g_i are sparse, the regression matrix $\mathbf{B}$ has more chance to be sparse. Moreover, if the basis is well-suited, we are able to represent a signal with few coefficients, which reduces the dimension. For details about the wavelet theory, see Mallat (1999).

We begin with an overview of some important aspects of wavelet basis.

Let ψ be a real wavelet function, satisfying for $t \in \mathbb{R}$,

$$\psi \in L^1 \cap L^2, t\psi \in L^1, \text{ and } \int_{\mathbb{R}} \psi(t)dt = 0.$$

We denote for $(l,h) \in \mathbb{Z}^2$ by $\psi_{l,h}$ the function defined from ψ by dyadic dilation and translation:

$$\psi_{l,h}(t) = 2^{l/2}\psi(2^l t - h) \text{ for } (l,h) \in \mathbb{Z}^2, \text{ for } t \in \mathbb{R}.$$

We define the wavelet coefficients of a signal f by

$$d_{l,h}(f) = \int_{\mathbb{R}} f(t)\psi_{l,h}(t)dt \text{ for } (l,h) \in \mathbb{Z}^2.$$

Let φ be a scaling function, related to ψ, and let $\varphi_{l,h}$ be the dilatation and translation of φ for $(l,h) \in \mathbb{Z}^2$. We also define, for $(l,h) \in \mathbb{Z}^2$, $\mathbf{B}_{l,h}(f) = \int_{\mathbb{R}} f(t)\varphi_{l,h}(t)dt$.

Note that scaling functions serve to construct approximations of the function of interest, while the wavelet functions serve to provide the details not captured by successive approximations.

We denote by V_l the space generated by $\{\varphi_{l,h}\}_{h\in\mathbb{Z}}$ and by W_l the space generated by $\{\psi_{l,h}\}_{h\in\mathbb{Z}}$ for all $l \in \mathbb{Z}$. Remark that

$$V_{l-1} = V_l \oplus W_l \text{ for all } l \in \mathbb{Z};$$
$$\mathrm{L}^2 = \oplus_{l\in\mathbb{Z}} W_l.$$

Let $L \in \mathbb{N}^*$. A signal f is decomposed by the approximation at the level L $A_L(f)$ and by the details $(d_{l,h})_{l<L}$, where

$$A_L(f) = \sum_{l>L}\sum_{h\in\mathbb{Z}} d_{l,h}(f)\psi_{l,h} = \sum_{h\in\mathbb{Z}} \boldsymbol{B}_{L,h}(f)\varphi_{L,h}.$$

This decomposition emphasizes the local nature of the wavelets, which, through the variable selection, highlights useful patterns for the clustering analysis.

We consider the sample $(f_i, g_i)_{1\le i\le n}$ and introduce the wavelet expansion of f_i onto the basis $\mathcal{B} = ((\varphi_{L,h})_{h\in\mathbb{Z}}, (\psi_{l,h})_{l\le L,h\in\mathbb{Z}})$: for all $t \in [0, 1]$,

$$f_i(t) = \underbrace{\sum_{h\in\mathbb{Z}} \mathbf{B}_{L,h}(f_i)\varphi_{L,h}(t)}_{A_L(f)(t)} + \sum_{l\le L}\sum_{h\in\mathbb{Z}} d_{l,h}(f_i)\psi_{l,h}(t).$$

The collection $\{\mathbf{B}_{L,h}(f_i), d_{l,h}(f_i)\}_{l\le L,h\in\mathbb{Z}}$ is the Discrete Wavelet Transform (DWT) of f_i in the basis $\mathcal{B}$.

We denote by $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$ the wavelet coefficient decomposition vectors, and the orthonormal characterization leads to

$$((f_i(t_j))_{1\le j\le p}) = \boldsymbol{W}\boldsymbol{x}_i;$$

where $\boldsymbol{x}_i$ the matrix of coefficients in the basis $\mathcal{B}$, and $\boldsymbol{W}$ a matrix defined by φ and ψ. The DWT is performed by a computationally fast pyramid algorithm (see Mallat 1989). In the same way, there exists $\boldsymbol{W}^{'}$ such that $((g_i(t_m))_{1\le m\le q}) = \boldsymbol{W}^{'}\boldsymbol{y}_i$, with $(\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n)$ a sample of wavelet coefficient decomposition vectors. As the matrices $\boldsymbol{W}$ and $\boldsymbol{W}^{'}$ are orthogonal, the mixture structure and Gaussian noise are kept. We consider the wavelet coefficient dataset $((\boldsymbol{y}_1, \boldsymbol{x}_1), \ldots, (\boldsymbol{y}_n, \boldsymbol{x}_n))$, which defines n observations whose probability distribution is modeled by the finite Gaussian mixture regression model (1).

5.2.2 Our procedures

We apply our both procedures to wavelet coefficient and obtain a clustering of the data. Rather than considering discretized functions $((g_i(t_m))_{1\le m\le q}, ((f_i(t_j))_{1\le j\le p}))_{1\le i\le n}$, we run our procedures on the wavelet coefficients sample $(\boldsymbol{y}_i, \boldsymbol{x}_i)_{1\le i\le n}$. The notion of relevant indices is quiet natural in this context: we are looking for patterns which explain the response variable, to focus on those who discriminate for the clustering.

5.3 Numerical experiments

We illustrate our procedures on functional data by using the Matlab wavelet toolbox (see Misiti et al. 2004 for details). Firstly, we simulate functional datasets, where the true model belongs to the collection of models. Then, we run our procedure on an electricity dataset, to cluster successive days. We have access to time series on load consumption measured every half-hour during 70 days. We extract the signal of each day and construct couples by each day and its eve and we aim at clustering these couples. To finish, we test our procedures on the well-known Tecator dataset. For each meat sample the data consists of a 100 channel spectrum of absorbances and the fat contents. These experiments illustrate different aspects of our procedures. Indeed, the simulated example illustrates that our procedures work in a functional context. The second example demonstrates the use of them for classification of real data already studied and in which we clearly understand the clusters. The last example also illustrates the use of the clustering to perform prediction.

5.3.1 Simulated functional data

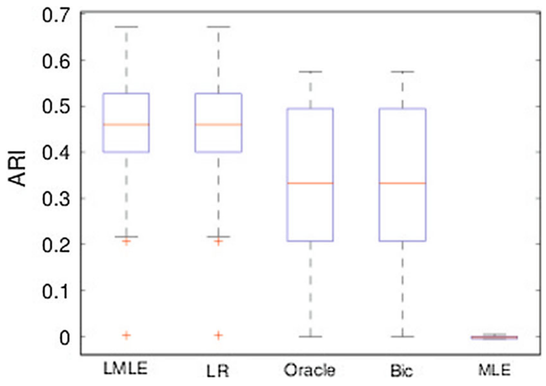
Firstly, we simulate a mixture regression model. Let f be a sample of a noised cosine function, discretized on a 15 points grid. Let g be, depending on the cluster, either f or $-f$, computed by a white-noise. Then, there are two clusters.

We use the Daubechies-2 basis at level 2 to decompose the signal.

Our procedures are run 20 times. The number of clusters is known, $\mathcal{K} = K = 2$. We compare the models we get by our procedures with the oracle model among the collection constructed by the Lasso-MLE procedure and with the model selected by the BIC criterion among this collection (Fig. 13).

The ARIs are lower for models constructed by our procedures. This simulated dataset shows that our procedures also perform clustering of functional data, considering the projection of the data onto a suitable wavelet basis.

Fig. 13 *Boxplots* of the ARI over the 20 simulations, for the Lasso-MLE procedure (LMLE), the Lasso-Rank procedure (LR), the oracle (Oracle), the BIC estimator (Bic) and the MLE (MLE)



5.3.2 Electricity dataset

We study the clustering of the electricity dataset. This example is studied in Misiti et al. (2007). We work on a sample of size $n = 70$ of couples of days, which is plotted in Fig. 14. For each couple, we have access to the half-hour load consumption.

We want to cluster the relationship between two successive days. In Fig. 15, we plot a week of load consumption.

In each couple, the first day corresponds to regressors and the second day corresponds to the response. In our opinion, a linear model is not appropriate, as the behavior

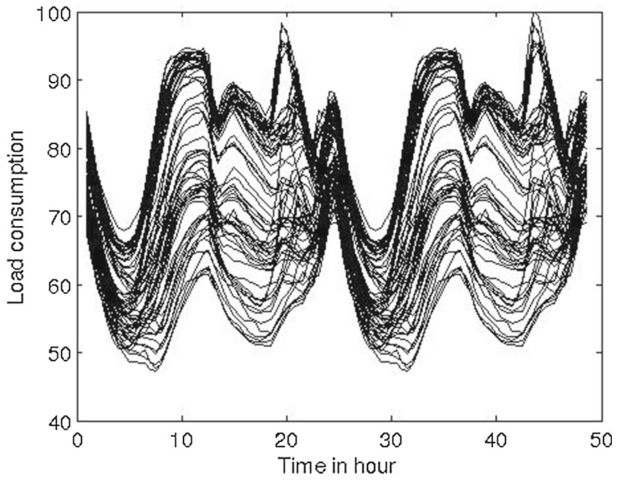


Fig. 14 Plot of the 70-sample of half-hour load consumption, on 2 days

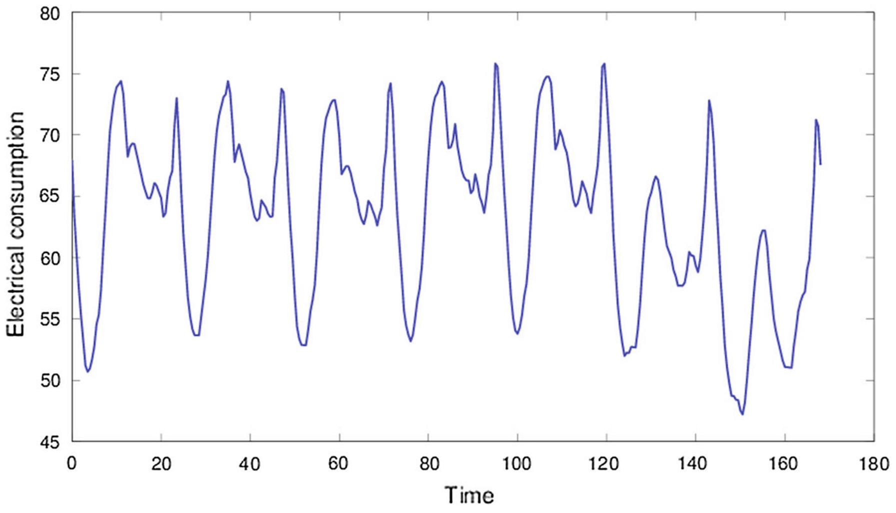


Fig. 15 Plot of a week of load consumption

from the eve to the day depends on which day we consider: there is a difference between working days and weekend days, which are both involved in Fig. 15.

To apply our procedures, we project the functions onto a wavelet basis. The symmlet-4 basis, at level 5, is used.

We run our procedures with the number of clusters varying from 2 to 6. Both our procedures select a model with 4 components. The first one considers couples of weekdays, the second Friday–Saturday, the third component is Saturday–Sunday and the fourth considers Sunday–Monday. This result is faithful with the knowledge we have about these data. Indeed, working days are similar, depending on the eve, whereas days off are different from their eve. Moreover, Misiti et al. (2007) obtained the same classification for this example.

5.3.3 Tecator dataset

This example deals with spectrometric data. More precisely, a food sample has been considered, which contained finely chopped pure meat with different fat contents. The data consist in a 100-channel spectrum of absorbances in the wavelength range $850 - 1050$ nm and of the fat percentage. We observe a sample of size $n = 215$. These data have been studied in a lot of articles, we refer for example to Ferraty and Vieu (2006). They test prediction and classification, supervised (where the fat content becomes a class, determined by whether the content is larger or smaller than 20 %), or not (ignoring the response variable). In this work, we take a different approach: we cluster data according to the relation between the fat content and the absorbance spectrum. We do not predict the response variable, because we do not know the class of a new observation. Estimate the class of a new observation is a difficult problem, in which we are not involved in this article.

One advantage of our procedures is that we know which coefficients, in the wavelet basis decomposition of the spectrum, are useful to describe the fat content.

The sample will be split into two subsamples, 165 observations for the learning set and 50 observations for the test set. This split is such that there is the same marginal distribution for the response in each sample.

The spectrum is a function, and we decompose it onto the Haar basis, at level 6. Our model does not take into account a constant coefficient to describe the response. Thereby, before running our procedure, we center the response according to the learning sample and each function $\boldsymbol{x}_i$ for every observation in the whole sample. Then, we estimate the mean of the response by the mean $\hat{\mu}$ over the learning sample.

We construct models on the training set using our procedure Lasso-MLE. Thanks to the estimations, we have access to relevant indices and we reconstruct signals by keeping only relevant indices. We also have access to the estimated a posteriori probability, and hence know the estimated probability for every observation to belong to each cluster. The procedure selects two models that we describe here. In Figs. 16 and 17, clusters performed on the training set are represented for the different models. The graph on the left is a candidate for representing each cluster, constructed by the mean of the spectra with an a posteriori probability greater than 0.6. We plot the reconstruction of the curve by keeping only variables corresponding to relevant indices in the wavelet

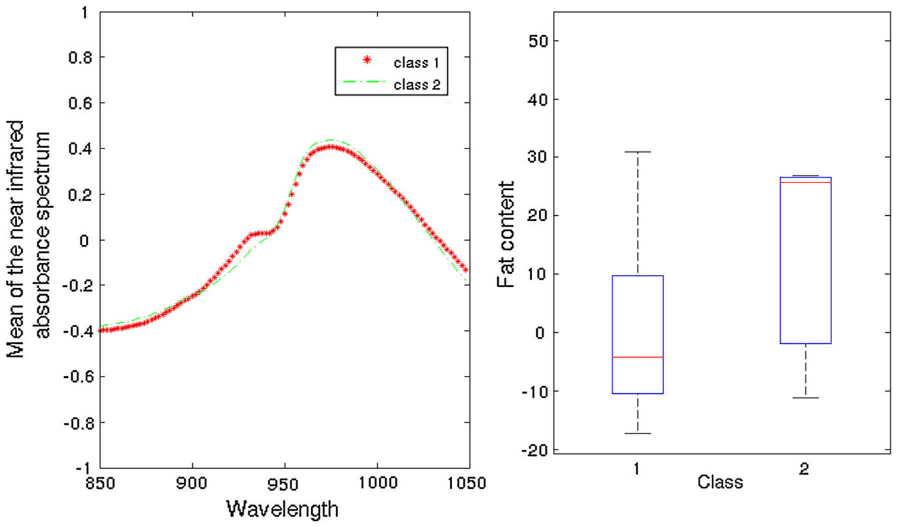


Fig. 16 Summarized results for Model 1. The graph on the *left* is a candidate for representing each cluster, constructed by the mean of reconstructed spectrum over an a posteriori probability greater than 0.6. On the *right side*, we present the *boxplot* of the fat values in each class, for observations with an a posteriori probability greater than 0.6

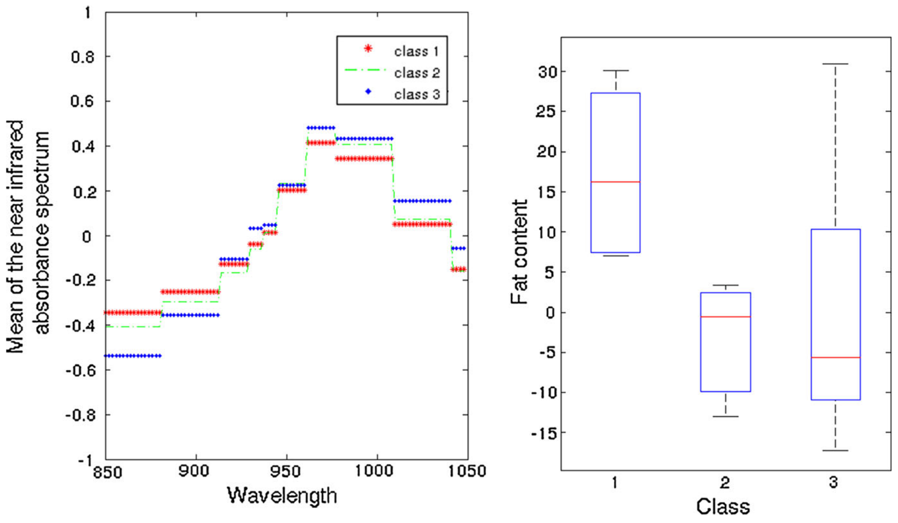


Fig. 17 Summarized results for Model 2. The graph on the *left* is a candidate for representing each cluster, constructed by the mean of reconstructed spectrum over an a posteriori probability greater than 0.6. On the *right side*, we present the boxplot of the fat values in each class, for observations with an a posteriori probability greater than 0.6

decomposition. On the right side, we present the boxplot of the fat contents in each class, for observations with an a posteriori probability greater than 0.6.

The first model has two classes, which are distinguished in the absorbance spectrum by the bump on wavelength around 940 nm. The first cluster is dominating, with $\hat{\pi}_1 =$

Table 3 Mean absolute percentage of error of the predicted value, for each model, for the learning sample

	Linear model in the class with higher probability	Mixing estimation
Model 1	0.200	0.198
Model 2	0.055	0.056

Table 4 Mean absolute percentage of error of the predicted value, for each model, for the test sample

	Linear model in the class with higher probability	Mixing estimation
Model 1	0.22196	0.21926
Model 2	0.20492	0.20662

0.95. The fat content is smaller in the first cluster than in the second cluster. According to the signal reconstruction, we see that almost all indices have been selected. This model seems consistent with the classification goal.

The second model has 3 classes and we remark several important wavelengths. Around 940 nm, there is a difference between classes, corresponding to the bump in Model 1. Around 970 nm, there is also a difference between classes, with higher or smaller values. The first class is dominating, with $\hat{\pi}_1 = 0.89$. Just a few number of indices have been selected and we are able to understand which coefficients are discriminating.

The first model selects only two classes, but almost all indices, whereas the second model has more classes and less indices: there is a trade-off between clusters and variable selection for the dimension reduction.

Based on these classifications, we compute the response according to the linear model. We use two ways to compute the fitted response $\hat{\mathbf{y}}$: either considering the linear model in the cluster selected by the MAP principle, or mixing estimations in each cluster thanks to the a posteriori probabilities. We summarize the results via the Mean Absolute Percentage Error, MAPE $= \frac{1}{n}\sum_{i=1}^{n} |(\hat{\mathbf{y}}_i - \mathbf{y}_i)/\mathbf{y}_i|$, presented in Table 3.

We next work with the test sample. We use the response and the regressors to know the a posteriori of each observation. Then, using our models, we compute the predicted fat values from the spectrometric curve, as before according to two ways, mixing or choosing the classes (Table 4).

Because the models are constructed on the learning sample, the MAPE for this sample are lower than for the test sample. Nevertheless, results are similar, saying that models are well constructed. This is particularly the case for Model 1, which is more consistent over a new test sample.

To conclude this study, we highlight the advantages of our Lasso-MLE procedure on these data. It provides a clustering of data, similar to the one done with supervised clustering in Ferraty and Vieu (2006), but we explain how this clustering is done.

This work has been done with the Lasso-MLE procedure. However, the same kind of results were obtained with the Lasso-Rank procedure.

6 Conclusion

In this article, two procedures are proposed to cluster regression data. Detecting the relevant clustering indices, the procedures are especially designed for high-dimensional data. We use an ℓ_1-regularization procedure to select indices and then deduce a reasonable random collection of models. Thus, we recast estimations of parameters of these models into a general model selection problem. These procedures are compared with commonly-used criteria on simulated data: the BIC criterion used to select a model, the maximum-likelihood estimator and the oracle model. In addition, we compare our procedures to other methods on benchmark data. Our procedures are applicable to functional data through projecting coefficients onto a wavelet basis.

Acknowledgements I am indebted to Jean-Michel Poggi and Pascal Massart for suggesting me to study this problem, and for stimulating discussions. I am also grateful to Jean-Michel Poggi for carefully reading the manuscript and making many useful suggestions. I thank Yves Misiti and Benjamin Auder for their help to speed up the code. I also thank referees for very interesting improvements and suggestions, and editors for their help for writing this paper. I am also grateful to Irène Gijbels for carefully proofreading the manuscript.

Appendix

In these appendices, we derive first the updating formulae in the EM algorithm in "EM algorithms". In "Group-Lasso MLE and Group-Lasso Rank procedures", we extend our procedures with the Group-Lasso estimator to select relevant indices, rather than using the Lasso estimator.

EM algorithms

EM algorithm for the Lasso estimator

Introduced by Dempster et al. (1977), the EM (Expectation-Maximization) algorithm is used to compute MLE, penalized or not.

The expected complete negative log-likelihood is denoted by

$$Q(\theta|\theta') = -\frac{1}{n} E_{\theta'}(l_c(\theta, \boldsymbol{Y}, \boldsymbol{x}, \boldsymbol{Z})|\boldsymbol{Y})$$

in which

$$l_c(\theta, \boldsymbol{Y}, \boldsymbol{x}, \boldsymbol{Z}) = \sum_{i=1}^{n}\sum_{k=1}^{K}[\boldsymbol{Z}_i]_k \log\left(\frac{\det(\boldsymbol{P}_k)}{(2\pi)^{q/2}} \exp\left(-\frac{1}{2}(\boldsymbol{P}_k\boldsymbol{Y}_i - \boldsymbol{x}_i\boldsymbol{\Phi}_k)^t(\boldsymbol{P}_k\boldsymbol{Y}_i - \boldsymbol{x}_i\boldsymbol{\Phi}_k)\right)\right) + [\boldsymbol{Z}_i]_k \log(\pi_k);$$

where $[\boldsymbol{Z}_i]$ are unobserved independent and identically distributed random variables following a multinomial distribution with parameters 1 and $(\pi_1, \ldots, \pi_K)$, describing the component-membership of the ith observation in the finite mixture regression model.

The expected complete penalized negative log-likelihood is

$$Q_{\text{pen}}(\theta|\theta') = Q(\theta|\theta') + \lambda \sum_{k=1}^{K} \pi_k ||\boldsymbol{\Phi}_k||_1.$$

Calculus for updating formulae

– E-step: compute $Q(\theta|\theta^{(\text{ite})})$, or, equivalently, compute for $k \in \{1, \ldots, K\}$, $i \in \{1, \ldots, n\}$,

$$\begin{aligned}\boldsymbol{\tau}_{i,k}^{(\text{ite})} &= E_{\theta^{(\text{ite})}}([\boldsymbol{Z}_i]_k|\boldsymbol{Y}_i)\\ &= \frac{\pi_k^{(\text{ite})} \det \boldsymbol{P}_k^{(\text{ite})} \exp\left(-\frac{1}{2}\left(\boldsymbol{P}_k^{(\text{ite})}\boldsymbol{y}_i - \boldsymbol{x}_i\boldsymbol{\Phi}_k^{(\text{ite})}\right)^t \left(\boldsymbol{P}_k^{(\text{ite})}\boldsymbol{y}_i - \boldsymbol{x}_i\boldsymbol{\Phi}_k^{(\text{ite})}\right)\right)}{\sum_{r=1}^{K} \pi_k^{(\text{ite})} \det \boldsymbol{P}_k^{(\text{ite})} \exp\left(-\frac{1}{2}\left(\boldsymbol{P}_k^{(\text{ite})}\boldsymbol{y}_i - \boldsymbol{x}_i\boldsymbol{\Phi}_k^{(\text{ite})}\right)^t \left(\boldsymbol{P}_k^{(\text{ite})}\boldsymbol{y}_i - \boldsymbol{x}_i\boldsymbol{\Phi}_k^{(\text{ite})}\right)\right)}\end{aligned}$$

Thanks to the MAP principle, the clustering is updated.

– M-step: optimize $Q_{\text{pen}}(\theta|\theta^{(\text{ite})})$ with respect to θ.
For this, rewrite the Karush–Kuhn–Tucker conditions. We have

$$\begin{aligned}&Q_{\text{pen}}(\theta|\theta^{(\text{ite})})\\ &= -\frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{K} E_{\theta^{(\text{ite})}}\left([\boldsymbol{Z}_i]_k \log\left(\frac{\det(\boldsymbol{P}_k)}{(2\pi)^{q/2}} \exp\left(-\frac{1}{2}(\boldsymbol{P}_k\boldsymbol{y}_i - \boldsymbol{x}_i\boldsymbol{\Phi}_k)^t(\boldsymbol{P}_k\boldsymbol{y}_i - \boldsymbol{x}_i\boldsymbol{\Phi}_k)\right)\right)|\boldsymbol{Y}\right)\\ &\quad -\frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{K} E_{\theta^{(\text{ite})}}([\boldsymbol{Z}_i]_k \log\pi_k|\boldsymbol{Y}) + \lambda\sum_{k=1}^{K}\pi_k||\boldsymbol{\Phi}_k||_1\\ &= -\frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{K} -\frac{1}{2}(\boldsymbol{P}_k\boldsymbol{y}_i - \boldsymbol{x}_i\boldsymbol{\Phi}_k)^t(\boldsymbol{P}_k\boldsymbol{y}_i - \boldsymbol{x}_i\boldsymbol{\Phi}_k)E_{\theta^{(\text{ite})}}([\boldsymbol{Z}_i]_k|\boldsymbol{Y})\\ &\quad -\frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{K}\sum_{m=1}^{q} \log\left(\frac{[\boldsymbol{P}_k]_{m,m}}{\sqrt{2\pi}}\right) E_{\theta^{(\text{ite})}}([\boldsymbol{Z}_i]_k|\boldsymbol{Y})\\ &\quad -\frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{K} E_{\theta^{(\text{ite})}}([\boldsymbol{Z}_i]_k|\boldsymbol{Y})\log\pi_k + \lambda\sum_{k=1}^{K}\pi_k||\boldsymbol{\Phi}_k||_1. \qquad (14)\end{aligned}$$

Firstly, we optimize this formula with respect to π: this is equivalent to optimizing

$$-\frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{K}\boldsymbol{\tau}_{i,k}^{(\text{ite})}\log(\pi_k) + \lambda\sum_{k=1}^{K}\pi_k||\boldsymbol{\Phi}_k||_1.$$

We obtain

$$\pi_k^{(\text{ite}+1)} = \pi_k^{(\text{ite})} + t^{(\text{ite})}\left(\frac{\sum_{i=1}^{n}\boldsymbol{\tau}_{i,k}^{(\text{ite})}}{n} - \pi_k^{(\text{ite})}\right); \qquad (15)$$

where $t^{(\text{ite})} \in (0,1]$ is the largest value in $\{0.1^l, l \in \mathbb{N}\}$ such that the update of π defined in (15) leads to improving the expected complete penalized log-likelihood function.
To optimize (14) with respect to $(\mathbf{\Phi}, \mathbf{P})$, we rewrite the expression: it is similar to the optimization of

$$-\frac{1}{n}\sum_{i=1}^{n}\left(\boldsymbol{\tau}_{i,k}^{(\text{ite})}\sum_{m=1}^{q}\log([\boldsymbol{P}_k]_{m,m}) - \frac{1}{2}(\boldsymbol{P}_k\widetilde{\boldsymbol{y}}_i^{(\text{ite})} - \widetilde{\boldsymbol{x}}_i^{(\text{ite})}\mathbf{\Phi}_k)^t(\boldsymbol{P}_k\widetilde{\boldsymbol{y}}_i^{(\text{ite})} - \widetilde{\boldsymbol{x}}_i^{(\text{ite})}\mathbf{\Phi}_k)\right)$$
$$+\lambda\pi_k||\mathbf{\Phi}_k||_1$$

for all $k \in \{1,\ldots,K\}$, with for all $j \in \{1,\ldots,p\}$ for all $i \in \{1,\ldots,n\}$, for all $k \in \{1,\ldots,K\}$,

$$([\widetilde{\boldsymbol{y}}_i^{(\text{ite})}]_{k,.}, [\widetilde{\boldsymbol{x}}_i^{(\text{ite})}]_{k,.}) = \sqrt{\boldsymbol{\tau}_{i,k}^{(\text{ite})}}(\boldsymbol{y}_i, \boldsymbol{x}_i).$$

Remark that this is equivalent to optimizing

$$-\frac{1}{n}n_k^{(\text{ite})}\sum_{m=1}^{q}\log([\boldsymbol{P}_k]_{m,m}) + \frac{1}{2n}\sum_{i=1}^{n}\sum_{m=1}^{q}\left([\boldsymbol{P}_k]_{m,m}[\widetilde{\boldsymbol{y}}_i^{(\text{ite})}]_{k,m} - [\mathbf{\Phi}_k]_{m,.}[\widetilde{\boldsymbol{x}}_i^{(\text{ite})}]_{k,.}\right)^2$$
$$+\lambda\pi_k||\mathbf{\Phi}_k||_1; \tag{16}$$

with $n_k^{(\text{ite})} = \sum_{i=1}^{n}\boldsymbol{\tau}_{i,k}^{(\text{ite})}$. With respect to $[\boldsymbol{P}_k]_{m,m}$, the optimization of (16) is the solution of

$$-\frac{n_k^{(\text{ite})}}{n}\frac{1}{[\boldsymbol{P}_k]_{m,m}} + \frac{1}{2n}\sum_{i=1}^{n}2[\widetilde{\boldsymbol{y}}_i^{(\text{ite})}]_{k,m}\left([\boldsymbol{P}_k]_{m,m}[\widetilde{\boldsymbol{y}}_i^{(\text{ite})}]_{k,m} - [\mathbf{\Phi}_k]_{m,.}[\widetilde{\boldsymbol{x}}_i^{(\text{ite})}]_{k,.}\right) = 0$$

for all $k \in \{1,\ldots,K\}$, for all $m \in \{1,\ldots,q\}$, which is equivalent to

$$-1 + \frac{1}{n_k^{(\text{ite})}}[\boldsymbol{P}_k]_{m,m}^2\sum_{i=1}^{n}[\widetilde{\boldsymbol{y}}_i^{(\text{ite})}]_{k,m}^2 - \frac{1}{n_k^{(\text{ite})}}[\boldsymbol{P}_k]_{m,m}\sum_{i=1}^{n}[\widetilde{\boldsymbol{y}}_i^{(\text{ite})}]_{k,m}[\mathbf{\Phi}_k]_{m,.}[\widetilde{\boldsymbol{x}}_i^{(\text{ite})}]_{k,.} = 0$$
$$\Leftrightarrow -1 + [\boldsymbol{P}_k]_{m,m}^2\frac{1}{n_k^{(\text{ite})}}||[\widetilde{\boldsymbol{y}}^{(\text{ite})}]_{k,m}||_2^2 - [\boldsymbol{P}_k]_{m,m}\frac{1}{n_k^{(\text{ite})}}\langle[\widetilde{\boldsymbol{y}}^{(\text{ite})}]_{k,m}, [\mathbf{\Phi}_k]_{m,.}[\widetilde{\boldsymbol{x}}^{(\text{ite})}]_{k,.}\rangle = 0.$$

The discriminant is

$$\Delta_{k,m} = \left(-\frac{1}{n_k^{(\text{ite})}}\langle[\widetilde{\boldsymbol{y}}^{(\text{ite})}]_{k,m}, [\mathbf{\Phi}_k]_{m,.}[\widetilde{\boldsymbol{x}}^{(\text{ite})}]_{k,.}\rangle\right)^2 - \frac{4}{n_k^{(\text{ite})}}||[\widetilde{\boldsymbol{y}}^{(\text{ite})}]_{k,m}||_2^2.$$

Then, for all $k \in \{1,\ldots,K\}$, for all $m \in \{1,\ldots,q\}$,

$$[\boldsymbol{P}_k]_{m,m} = \frac{n_k^{(\text{ite})}\langle[\widetilde{\boldsymbol{y}}^{(\text{ite})}]_{k,m}, [\mathbf{\Phi}_k]_{m,.}[\widetilde{\boldsymbol{x}}^{(\text{ite})}]_{k,.}\rangle + \sqrt{\Delta_{k,m}}}{2n_k^{(\text{ite})}||[\widetilde{\boldsymbol{y}}^{(\text{ite})}]_{k,m}||_2^2}.$$

We also look at Eq. (14) as a function of the variable $\boldsymbol{\Phi}$: according to the partial derivative with respect to $[\boldsymbol{\Phi}_k]_{m,j}$, we obtain for all $m \in \{1,\dots,q\}$, for all $k \in \{1,\dots,K\}$, for all $j \in \{1,\dots,p\}$,

$$\sum_{i=1}^{n} [\widetilde{\boldsymbol{x}}_i^{(\text{ite})}]_{k,j} \left([\boldsymbol{P}_k]_{m,m} [\widetilde{\boldsymbol{y}}_i^{(\text{ite})}]_{k,m} - \sum_{j_2=1}^{p} [\widetilde{\boldsymbol{x}}_i^{(\text{ite})}]_{k,j_2} [\boldsymbol{\Phi}_k]_{m,j_2} \right) \\ - n\lambda\pi_k \operatorname{sgn}([\boldsymbol{\Phi}_k]_{m,j}) = 0,$$

where sgn is the sign function. Then, for all $k \in \{1,\dots,K\}$, $j \in \{1,\dots,p\}$, $m \in \{1,\dots,q\}$,

$$[\boldsymbol{\Phi}_k]_{m,j} = \frac{\sum_{i=1}^{n} [\widetilde{\boldsymbol{x}}_i^{(\text{ite})}]_{k,j} [\boldsymbol{P}_k]_{m,m} [\widetilde{\boldsymbol{y}}_i^{(\text{ite})}]_{k,m} - \sum_{\substack{j_2=1\\ j_2\neq j}}^{p} [\widetilde{\boldsymbol{x}}_i^{(\text{ite})}]_{k,j} [\widetilde{\boldsymbol{x}}_i^{(\text{ite})}]_{k,j_2} [\boldsymbol{\Phi}_k]_{m,j_2} - n\lambda\pi_k \operatorname{sgn}([\boldsymbol{\Phi}_k]_{j,m})}{\|[\widetilde{\boldsymbol{x}}^{(\text{ite})}]_{k,j}\|_2^2}.$$

Let, for all $k \in \{1,\dots,K\}$, $j \in \{1,\dots,p\}$, $m \in \{1,\dots,q\}$,

$$[\boldsymbol{S}_k]_{j,m} = -\sum_{i=1}^{n} [\widetilde{\boldsymbol{x}}_i^{(\text{ite})}]_{k,j} [\boldsymbol{P}_k]_{m,m} [\widetilde{\boldsymbol{y}}_i^{(\text{ite})}]_{k,m} + \sum_{\substack{j_2=1\\ j_2\neq j}}^{p} [\widetilde{\boldsymbol{x}}_i^{(\text{ite})}]_{k,j} [\widetilde{\boldsymbol{x}}_i^{(\text{ite})}]_{k,j_2} [\boldsymbol{\Phi}_k]_{m,j_2}.$$

Then

$$[\boldsymbol{\Phi}_k]_{m,j} = \frac{-[\boldsymbol{S}_k]_{j,m} - n\lambda\pi_k \operatorname{sgn}([\boldsymbol{\Phi}_k]_{m,j})}{\|[\widetilde{\boldsymbol{x}}^{(\text{ite})}]_{k,j}\|_2^2} = \begin{cases} \frac{-[\boldsymbol{S}_k]_{j,m} + n\lambda\pi_k}{\|[\widetilde{\boldsymbol{x}}^{(\text{ite})}]_{k,j}\|_2^2} & \text{if } [\boldsymbol{S}_k]_{j,m} > n\lambda\pi_k \\ -\frac{[\boldsymbol{S}_k]_{j,m} + n\lambda\pi_k}{\|[\widetilde{\boldsymbol{x}}^{(\text{ite})}]_{k,j}\|_2^2} & \text{if } [\boldsymbol{S}_k]_{j,m} < -n\lambda\pi_k \\ 0 & \text{elsewhere.} \end{cases}$$

From these equalities, we write the updating formulae.

$$\pi_k^{(\text{ite+1})} = \pi_k^{(\text{ite})} + t^{(\text{ite})} \left(\frac{n_k^{(\text{ite})}}{n} - \pi_k^{(\text{ite})} \right);$$

$$[\boldsymbol{P}_k]_{m_1,m_2}^{(\text{ite+1})} = \begin{cases} \frac{n_k^{(\text{ite})} \langle [\widetilde{\boldsymbol{y}}_.^{(\text{ite})}]_{k,m}, [\boldsymbol{\Phi}_k]_{m,.}^{(\text{ite})} [\widetilde{\boldsymbol{x}}^{(\text{ite})}_{.}]_{k,.} \rangle + \sqrt{\Delta_{k,m}}}{2 n_k^{(\text{ite})} \|[\widetilde{\boldsymbol{y}}_.^{(\text{ite})}]_{k,m}\|_2^2} & \text{if } m_1 = m_2 = m; \\ 0 & \text{elsewhere;} \end{cases}$$

$$[\boldsymbol{\Phi}_k]_{j,m}^{(\text{ite+1})} = \begin{cases} \frac{-[S_k^{(\text{ite})}]_{j,m} + n\lambda(\pi_k^{(\text{ite})})}{\|[\widetilde{\boldsymbol{x}}^{(\text{ite})}]_{k,j}\|_2^2} & \text{if } [S_k^{(\text{ite})}]_{j,m} > n\lambda(\pi_k^{(\text{ite})}); \\ -\frac{[S_k^{(\text{ite})}]_{j,m} + n\lambda(\pi_k^{(\text{ite})})}{\|[\widetilde{\boldsymbol{x}}^{(\text{ite})}]_{k,j}\|_2^2} & \text{if } [S_k^{(\text{ite})}]_{j,m} < -n\lambda(\pi_k^{(\text{ite})}); \\ 0 & \text{else.} \end{cases}$$

EM algorithm for the rank procedure

To take into account the matrix structure, we want to make a dimension reduction on the rank of the regression matrix. If we know the clustering, we compute the low rank estimator for a linear model in each cluster.

Indeed, an estimator of fixed rank r is known in the linear regression case: denoting A^+ the Moore-Penrose pseudo-inverse of A and $[A]_r = UD_rV^t$ in which D_r is obtained from D by setting $(D_r)_{i,i} = 0$ for $i \geq r+1$, with UDV^t the singular decomposition of A, if $\boldsymbol{Y} = \mathbf{B}\boldsymbol{x} + \boldsymbol{\Sigma}$, an estimator of $\mathbf{B}$ with rank r is $\hat{\mathbf{B}}_r = [(\boldsymbol{x}^t\boldsymbol{x})^+\boldsymbol{x}^t\boldsymbol{y}]_r$.

However, we do not know the clustering of the observations,. We will use an EM algorithm, where the E-step computes the a posteriori probability .

We suppose in this case that $\boldsymbol{\Sigma}_k$ and π_k are known, for all $k \in \{1,\ldots,K\}$. We use this algorithm to determine $\boldsymbol{\Phi}_k$, for all $k \in \{1,\ldots,K\}$, with ranks fixed to $\boldsymbol{R} = (R_1,\ldots,R_K)$.

– E-step: compute for $k \in \{1,\ldots,K\}$, $i \in \{1,\ldots,n\}$,

$$\begin{aligned}\boldsymbol{\tau}_{i,k}^{(\text{ite})} &= E_{\theta^{(\text{ite})}}([\boldsymbol{Z}_i]_k|Y)\\ &= \frac{\pi_k^{(\text{ite})} \det \boldsymbol{P}_k^{(\text{ite})} \exp\left(-\frac{1}{2}\left(\boldsymbol{P}_k^{(\text{ite})}\boldsymbol{y}_i - \boldsymbol{x}_i\boldsymbol{\Phi}_k^{(\text{ite})}\right)^t\left(\boldsymbol{P}_k^{(\text{ite})}\boldsymbol{y}_i - \boldsymbol{x}_i\boldsymbol{\Phi}_k^{(\text{ite})}\right)\right)}{\sum_{r=1}^K \pi_k^{(\text{ite})} \det \boldsymbol{P}_k^{(\text{ite})} \exp\left(-\frac{1}{2}\left(\boldsymbol{P}_k^{(\text{ite})}\boldsymbol{y}_i - \boldsymbol{x}_i\boldsymbol{\Phi}_k^{(\text{ite})}\right)^t\left(\boldsymbol{P}_k^{(\text{ite})}\boldsymbol{y}_i - \boldsymbol{x}_i\boldsymbol{\Phi}_k^{(\text{ite})}\right)\right)}.\end{aligned}$$

– M-step: assign each observation to its estimated cluster, by the MAP principle applied thanks to the E-step. We say that i originates from component number $\text{argmax}_{k\in\{1,\ldots,K\}}\boldsymbol{\tau}_{i,k}^{(\text{ite})}$. Then, for all $k \in \{1,\ldots,K\}$, we define $\widetilde{\mathbf{B}}_k^{(\text{ite})} = (\boldsymbol{x}_{|k}^t\boldsymbol{x}_{|k})^{-1}\boldsymbol{x}_{|k}^t\boldsymbol{y}_{|k}$, in which $\boldsymbol{x}_{|k}$ and $\boldsymbol{y}_{|k}$ correspond to the observations belonging to the cluster k. For all $k \in \{1,\ldots,K\}$, we decompose $\widetilde{\mathbf{B}}_k^{(\text{ite})}$ in singular value: $\widetilde{\mathbf{B}}_k^{(\text{ite})} = USV^t$. Then, we estimate the regression matrix by $\hat{\mathbf{B}}_k^{(\text{ite})} = U[\boldsymbol{S}]_{R(k)}V^t$ in the cluster $k \in \{1,\ldots,K\}$.

Group-Lasso MLE and Group-Lasso Rank procedures

One way to perform these procedures is to consider the Group-Lasso estimator rather than the Lasso estimator to select relevant indices. Indeed, this estimator is more natural according to the relevant indices definition. Nevertheless, results are very similar, because we select grouped indices by the Lasso estimator or by the Group-Lasso estimator. In this section, we describe our procedures with the Group-Lasso estimator.

Context-definitions

Both our procedures take advantage of the Lasso estimator to select relevant indices and to reduce the dimension in case of high-dimensional data. First, recall what is meant by relevant indices.

Definition 2 A couple $(\boldsymbol{y}_m, \boldsymbol{x}_j)$ is said to be *irrelevant* for the clustering if $[\boldsymbol{\Phi}_1]_{m,j} = \ldots = [\boldsymbol{\Phi}_K]_{m,j} = 0$, which means that the variable $\boldsymbol{x}_j$ does not explain the variable $\boldsymbol{y}_m$ for the clustering process. We also say that the indices (m, j) are irrelevant if the couple $(\boldsymbol{y}_m, \boldsymbol{x}_j)$ is irrelevant. A *relevant* couple is a couple which is not irrelevant: at least in one cluster k, the coefficient $[\boldsymbol{\Phi}_k]_{m,j}$ is not equal to zero. We denote by J the set of indices (m, j) of relevant couples $(\boldsymbol{y}_m, \boldsymbol{x}_j)$.

According to this definition, we introduce the Group-Lasso estimator.

Definition 3 The *Group-Lasso estimator* for mixture regression models with regularization parameter $\lambda \geq 0$ is defined by

$$\hat{\theta}_\lambda^{\text{Group-Lasso}} := \underset{\theta \in \Theta_K}{\operatorname{argmin}} \left\{ -\frac{1}{n} \tilde{l}_\lambda(\theta) \right\};$$

where

$$-\frac{1}{n}\tilde{l}_\lambda(\theta) = -\frac{1}{n} l(\theta) + \lambda \sum_{j=1}^{p} \sum_{m=1}^{q} \sqrt{k} ||[\boldsymbol{\Phi}]_{m,j}||_2;$$

where $||[\boldsymbol{\Phi}]_{m,j}||_2^2 = \sum_{k=1}^K |[\boldsymbol{\Phi}_k]_{m,j}|^2$ and with λ to specify.

This Group-Lasso estimator has the advantage to shrink to zero grouped indices rather than indices one by one. It is consistent with the relevant indices definition.

However, depending on the data, it is interesting to look for which indices are equal to zero first. One way is to extend this work with Lasso-Group-Lasso estimator, described for the example for the linear model in Simon et al. (2013).

Let us describe two additional procedures, which use the Group-Lasso estimator rather than the Lasso estimator to detect relevant indices.

Group-Lasso-MLE procedure

This procedure is decomposed into three main steps: we construct a collection of models, then in each model we compute the MLE and we select the best one among all the models.

The first step consists in constructing a collection of models $\{\mathcal{H}_{(K,\tilde{J})}\}_{(K,\tilde{J}) \in \mathcal{M}}$ in which $\mathcal{H}_{(K,\tilde{J})}$ is defined by

$$\mathcal{H}_{(K,\tilde{J})} = \{h_\theta(.|x)\}; \tag{17}$$

where

$$h_\theta(y|x) = \sum_{k=1}^{K} \frac{\pi_k \det(\boldsymbol{P}_k)}{(2\pi)^{q/2}} \exp\left(-\frac{(\boldsymbol{P}_k y - \boldsymbol{\Phi}_k^{[\tilde{J}]} x)^t (\boldsymbol{P}_k y - \boldsymbol{\Phi}_k^{[\tilde{J}]} x)}{2} \right),$$

and

$$\theta = (\pi_1, \ldots, \pi_K, \boldsymbol{\Phi}_1, \ldots, \boldsymbol{\Phi}_K, \boldsymbol{P}_1, \ldots, \boldsymbol{P}_K) \in \Pi_K \times \left(\mathbb{R}^{q\times p}\right)^K \times \left(\mathbb{D}_q^{++}\right)^K .$$

The collection of models is indexed by $\mathcal{M} = \mathcal{K} \times \widetilde{\mathcal{J}}$. We denote by $\mathcal{K} \subset \mathbb{N}^*$ the admitted number of classes. We bound $\mathcal{K}$ without loss of generality. Denote also $\widetilde{\mathcal{J}}$ a collection of subsets of $\{1, \ldots, q\} \times \{1, \ldots, p\}$, constructed by the Group-Lasso estimator.

To detect the relevant indices and construct the set $\widetilde{J} \in \widetilde{\mathcal{J}}$, we will use the Group-Lasso estimator defined by (3). For $K \in \mathcal{K}$, we propose to construct a data-driven grid G_K of regularization parameters by using the updating formulae of the mixture parameters in the EM algorithm.

Then, for each $\lambda \in G_K$, we compute the Group-Lasso estimator defined by

$$\hat{\theta}_\lambda^{\text{Group-Lasso}} = \underset{\theta\in\Theta_K}{\operatorname{argmin}} \left\{ -\frac{1}{n}\sum_{i=1}^{n} \log(h_\theta(\boldsymbol{y}_i|\boldsymbol{x}_i)) + \lambda \sum_{j=1}^{p}\sum_{m=1}^{q} \sqrt{K}||[\boldsymbol{\Phi}]_{m,j}||_2 \right\}.$$

For a fixed number of mixture components $K \in \mathcal{K}$ and a regularization parameter λ, we use a generalized EM algorithm to approximate this estimator. Then, for each $K \in \mathcal{K}$ and for each $\lambda \in G_K$, we construct the set of relevant indices $\widetilde{J}_\lambda$. We denote by $\widetilde{\mathcal{J}}$ the collection of all these sets.

The second step consists in approximating the MLE

$$\hat{h}^{(K,\widetilde{J})} = \underset{t\in\mathcal{H}_{(K,\widetilde{J})}}{\operatorname{argmin}} \left\{ -\frac{1}{n}\sum_{i=1}^{n} \log(t(\boldsymbol{y}_i|\boldsymbol{x}_i)) \right\};$$

using the EM algorithm for each model $(K, \widetilde{J}) \in \mathcal{M}$.

The third step is devoted to model selection, for which we use the slope heuristic described in Birgé and Massart (2007).

Group-Lasso-Rank procedure

We propose a second procedure to take into account the matrix structure. For each model belonging to the collection $\mathcal{H}_{(K,\widetilde{J})}$, a subcollection is constructed, varying the rank of $\boldsymbol{\Phi}$. We describe this procedure in more details.

As in the Group-Lasso-MLE procedure, we first construct a collection of models, thanks to the ℓ_1-approach. We obtain an estimator for θ, denoted by $\hat{\theta}_\lambda^{\text{Group-Lasso}}$, for each model belonging to the collection. We deduce the set of relevant indices, denoted by $\widetilde{J}$ and this for all $K \in \mathcal{K}$: we deduce $\widetilde{\mathcal{J}}$ the collection of set of relevant indices.

The second step consists in constructing a subcollection of models with rank sparsity, denoted by

$$\{\check{\mathcal{H}}_{(K,\widetilde{J},R)}\}_{(K,\widetilde{J},R)\in\widetilde{\mathcal{M}}}.$$

The model $\{\check{\mathcal{H}}_{(K,\widetilde{J},R)}\}$ has K components, the set $\widetilde{J}$ for relevant indices and R is the vector of the ranks of the matrix of regression coefficients in each group:

$$\check{\mathcal{H}}_{(K,\widetilde{J},R)} = \left\{ h_\theta^{(K,\widetilde{J},R)}(\boldsymbol{y}|\boldsymbol{x}) \right\} \tag{18}$$

where

$$h_\theta^{(K,\widetilde{J},R)}(\boldsymbol{y}|\boldsymbol{x}) = \sum_{k=1}^{K} \frac{\pi_k \det(\boldsymbol{P}_k)}{(2\pi)^{q/2}} \exp\left(-\frac{(\boldsymbol{P}_k \boldsymbol{y} - (\boldsymbol{\Phi}_k^{R_k})^{[\widetilde{J}]}\boldsymbol{x})^t (\boldsymbol{P}_k \boldsymbol{y} - (\boldsymbol{\Phi}_k^{R_k})^{[\widetilde{J}]}\boldsymbol{x})}{2} \right);$$

$$\theta = (\pi_1, \ldots, \pi_K, \boldsymbol{\Phi}_1^{R_1}, \ldots, \boldsymbol{\Phi}_K^{R_K}, \boldsymbol{P}_1, \ldots, \boldsymbol{P}_K) \in \Pi_K \times \Psi_K^R \times \left(\mathbb{R}_+^q\right)^K;$$

$$\Psi_K^R = \left\{ (\boldsymbol{\Phi}_1^{R_1}, \ldots, \boldsymbol{\Phi}_K^{R_K}) \in \left(\mathbb{R}^{q\times p}\right)^K | \mathrm{Rank}(\boldsymbol{\Phi}_1) = R_1, \ldots, \mathrm{Rank}(\boldsymbol{\Phi}_K) = R_K \right\};$$

and $\widetilde{\mathcal{M}}^R = \mathcal{K} \times \widetilde{\mathcal{J}} \times \mathcal{R}$. Denote by $\mathcal{K} \subset \mathbb{N}^*$ the possible number of components, $\widetilde{\mathcal{J}}$ a collection of subsets of $\{1, \ldots, q\} \times \{1, \ldots, p\}$ and $\mathcal{R}$ the set of vectors of size $K \in \mathcal{K}$ with rank values for each mean matrix. We compute the MLE under the rank constraint thanks to an EM algorithm. Indeed, we constrain the estimation of $\boldsymbol{\Phi}_k$, for all k, to have a rank equal to R_k, in keeping only the R_kth largest singular values. More details are given in "EM algorithm for the rank procedure". This finally leads to an estimator of the mean with row sparsity and low rank for each model.

References

Anderson TW (1951) Estimating linear restrictions on regression coefficients for multivariate normal distributions. Ann Math Stat 22(3):327–351

Baudry J-P, Maugis C, Michel B (2012) Slope heuristics: overview and implementation. Stat Comput 22(2):455–470

Birgé L, Massart P (2007) Minimal penalties for Gaussian model selection. Probab Theory Relat Fields 138(1–2):33–73

Bühlmann P, van de Geer S (2011) Statistics for high-dimensional data: methods, theory and applications, Springer Series in Statistics. Springer, Berlin

Bunea F, She Y, Wegkamp M (2012) Joint variable and rank selection for parsimonious estimation of high-dimensional matrices. Ann Stat 40(5):2359–2388

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

Ciarleglio A, Ogden T (2014) Wavelet-based scalar-on-function finite mixture regression models. Comput Stat Data Anal 93:86–96

Dempster AP, Laird NM, Rubin DB (1977) Maximum likelihood from incomplete data via the EM algorithm. Discussion. J R Stat Soc Ser B 39:1–38

Devijver E (2015) Finite mixture regression: a sparse variable selection by model selection for clustering. Electron J Stat 9(2):2642–2674

Devijver E (2015) Joint rank and variable selection for parsimonious estimation in high-dimension finite mixture regression model. arXiv:1501.00442

Ferraty F, Vieu P (2006) Nonparametric functional data analysis: theory and practice, Springer series in statistics. Springer, New York

Gareth J, Sugar C (2003) Clustering for sparsely sampled functional data. J Am Stat Assoc 98(462):397–408

Giraud C (2011) Low rank multivariate regression. Electron J Stat 5:775–799

Hubert L, Arabie P (1985) Comparing partitions. J Classif 2(1):193–218

Izenman A (1975) Reduced-rank regression for the multivariate linear model. J Multivar Anal 5(2):248–264

Jones PN, McLachlan GJ (1992) Fitting finite mixture models in a regression context. Aust J Stat 34:233–240

Mallat S (1989) A theory for multiresolution signal decomposition: the wavelet representation. IEEE Trans Pattern Anal Mach Intell 11:674–693

Mallat S (1999) A wavelet tour of signal processing. Academic Press, Dublin

McLachlan G, Peel D (2004) Finite Mixture Models, Wiley series in probability and statistics. Wiley, New York

Meinshausen N, Bühlmann P (2010) Stability selection. J R Stat Soc Ser B Stat Methodol 72(4):417–473

Meynet C, Maugis-Rabusseau C (2012) A sparse variable selection procedure in model-based clustering. Research report, September

Misiti M, Misiti Y, Oppenheim G, Poggi J-M (2004) Matlab Wavelet Toolbox User's Guide. Version 3. The Mathworks Inc, Natick, MA

Misiti M, Misiti Y, Oppenheim G, Poggi J-M (2007) Clustering signals using wavelets. In: Sandoval Francisco, Prieto Alberto, Cabestany Joan, Graña Manuel (eds) Computational and Ambient Intelligence, vol 4507, Lecture Notes in Computer Science. Springer, Berlin Heidelberg, pp 514–521

Park T, Casella G (2008) The Bayesian lasso. J Am Stat Assoc 103(482):681–686

Ramsay JO, Silverman BW (2005) Functional data analysis, Springer series in statistics. Springer, New York

Simon N, Friedman J, Hastie T, Tibshirani R (2013) A sparse-group lasso. J Comput Graph Stat 22:231–245

Städler N, Bühlmann P, Van de Geer S (2010) ℓ_1-penalization for mixture regression models. Test 19(2): 209–256

Tibshirani R (1996) Regression shrinkage and selection via the lasso. J R Stat Soc Ser B 58(1):267–288

Tseng P (2001) Convergence of a block coordinate descent method for nondifferentiable minimization. J Optim Theory Appl 109:475–494

Wu J (1983) On the convergence properties of the EM algorithm. Ann Stat 1:95–103

Yao F, Fu Y, Lee T (2011) Functional mixture regression. Biostatistics 2:341–353

Zhao Y, Ogden T, Reiss P (2012) Wavelet-based LASSO in functional linear regression. J Comput Graph Stat 21(3):600–617