

A divisive clustering method for functional data with special consideration of outliers

Ana Justel^{1,2} · Marcela Svarc^{3,4}

Received: 29 August 2016 / Revised: 2 June 2017 / Accepted: 31 July 2017 /
Published online: 11 August 2017
© Springer-Verlag GmbH Germany 2017

Abstract This paper presents DivClusFD, a new divisive hierarchical method for the non-supervised classification of functional data. Data of this type present the peculiarity that the differences among clusters may be caused by changes as well in level as in shape. Different clusters can be separated in different subregion and there may be no subregion in which all clusters are separated. In each step of division, the DivClusFD method explores the functions and their derivatives at several fixed points, seeking the subregion in which the highest number of clusters can be separated. The number of clusters is estimated via the gap statistic. The functions are assigned to the new clusters by combining the k -means algorithm with the use of functional boxplots to identify functions that have been incorrectly classified because of their atypical local behavior. The DivClusFD method provides the number of clusters, the classification of the observed functions into the clusters and guidelines that may be for interpreting the clusters. A simulation study using synthetic data and tests of the performance of

This work was supported by the Spanish Agencia Estatal de Investigación (AEI) and Fondo Europeo de Desarrollo Regional (FEDER), Grant CTM2016-79741-R for MICROAIPOLAR project (to A. Justel and M. Svarc) and Spanish Ministerio de Economía y Competitividad, Grant CTM2011-28736 (to A. Justel).

✉ Marcela Svarc
msvarc@udesa.edu.ar

Ana Justel
ana.justel@uam.es

¹ Department of Mathematics, Universidad Autónoma de Madrid, Madrid, Spain

² UC3M-BS Institute of Financial Big Data, Universidad Carlos III de Madrid, Madrid, Spain

³ Department of Mathematics and Sciences, Universidad de San Andrés, Victoria, Argentina

⁴ CONICET, Buenos Aires, Argentina

the DivClusFD method on real data sets indicate that this method is able to classify functions accurately.

Keywords Hierarchical clustering · Functional boxplot · Gap statistic

Mathematics Subject Classification 62H30

1 Introduction

Functional data analysis (FDA) is currently a very active area of research, mainly because it has become very easy to collect and store data in continuous time. Although generally each datum is recorded for only a finite number of time points, it is possible to analyze the data as functions rather than as vectors ([Ramsay and Silverman 2005](#)). The observations in this context are the entire curves.

Clustering methods attempt to classify similar objects into the same group and dissimilar ones into different groups. The problem of unsupervised classification with functional data is difficult to address because of the infinite dimensionality of the data and the lack of a definition for the probability density of a functional random variable ([Jacques and Preda 2014b](#)). Recently, a number of clustering methods based on different strategies have been introduced. Under the assumption that the functions belong to a Hilbert space, they can be represented by a series expansion using a convenient basis of functions (the most common bases are the Fourier basis, Haar wavelets and other types of wavelets); then, clustering can be performed on the finite vector of the first coefficients in the basis of functions in the series representation ([Abraham et al. 2003](#); [James and Sugar 2003](#); [Ray and Mallick 2006](#)). Functional principal component analysis may also be useful to reduce the dimensionality of the data. In this case, clusters can be found in the low-dimensional space of the first principal component scores ([Bouveyron and Jacques 2011](#); [Chiou and Li 2011](#); [Jacques and Preda 2013, 2014a](#)). Some other methods are based on the application of nonparametric clustering techniques with distances or dissimilarities specifically defined for functional data ([Tarpey and Kinatader 2003](#); [Tokushige et al. 2007](#); [Ieva et al. 2013](#)). [Jacques and Preda \(2014b\)](#) thoroughly described the state of the art in this field, and more references may be found therein.

As in finite-dimensional clustering problems, there is no “one size fits all” method for analyzing any data set, and the nature of the data should determine the procedure to be used. In fact, there are clustering methods that show outstanding performance on certain configurations of data but perform poorly for other data constellations. Functional data sets present the peculiarity that the differences between clusters can involve changes in level or shape. These changes may be maintained throughout the domain of the functions or may affect only a given subregion. In functional data classification problems, there is often no subregion in which all clusters are separated.

Considering the derivatives of observed functions can be very useful for highlighting differences in shape that are not visible in trajectories. [Keogh and Pazzani \(2001\)](#) used the first derivative to improve Dynamic Time Warping, and [Alonso et al. \(2012\)](#) considered a distance based on derivatives in the supervised classification of functional data. Moreover, the simultaneous exploration of trajectories and derivatives is

of key importance because, in functional data, the different groups may differ in the trajectories in some cases and in the derivatives of the observed functions in others. The simple example with three groups that is shown in Fig. 1a illustrates this idea. The functions of groups 1 and 2 are constant, although at different levels, whereas the functions of group 3 have a constant positive slope. However, the levels of the functions of group 3 are similar to those of the functions of group 1. In any region of the domain of the functions, only two clusters can be identified, as shown in Fig. 1b (group 2 on one side and groups 1 and 3 on the other). Upon switching to first derivatives, we observe that all derivative functions are constant. Groups 1 and 2 have null derivatives, whereas the derivative of group 3 is positive. Upon analyzing any subregion of the derivatives, we once again observe separation between only two clusters (groups 1 and 2 on one side and group 3 on the other). These clusters, which are shown in Fig. 1c, are different from the clusters found using the original functions. In summary, applying clustering methods to the trajectories allows the functions to be separated by level, whereas when such methods are applied to the derivatives, the functions are separated by shape. We will only succeed in identifying all three groups if we follow a two-step clustering scheme. For instance, Fig. 1d illustrates a case in which two of the groups are identified in the first step based on the difference in their trajectories, and in the second step, the last of the groups is extracted based on the separation observed between the derivatives.

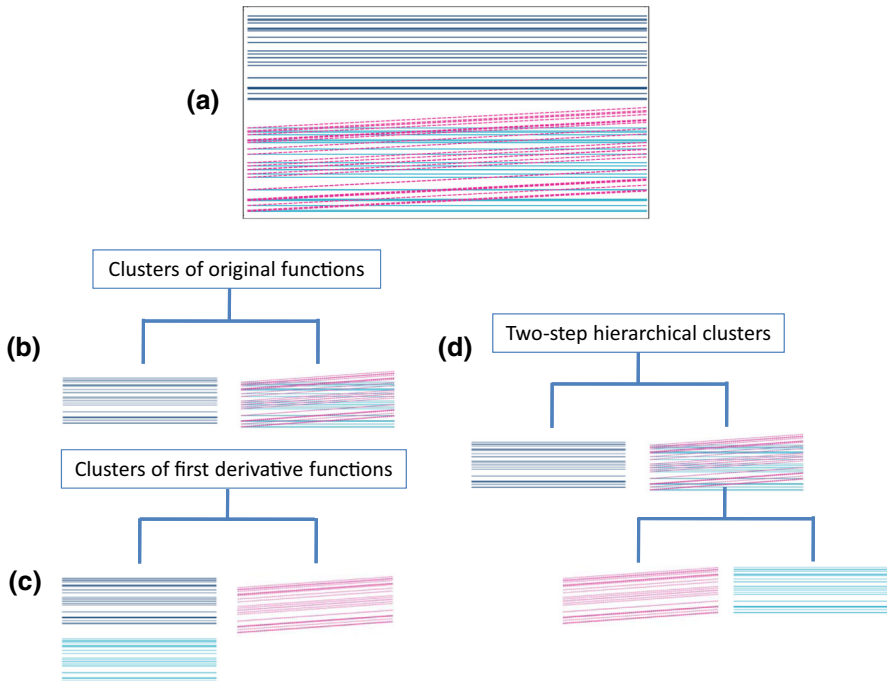


Fig. 1 a Set of artificial data: the *light solid lines* belong to group 1, the *dark solid lines* belong to group 2, and the *dashed lines* belong to group 3; b clusters identified from the original functions; c clusters identified from the first-derivative functions; and d two-step hierarchical clustering

The aim of this paper is to introduce a new divisive hierarchical clustering method designed specifically for FDA, named the DivClusFD method. The idea is to extend the previously described two-step clustering scheme to include all steps of subdivision that are necessary for identifying the complete cluster structure in a functional data set. In each of these splitting steps, the DivClusFD method explores the functions and their derivatives based on their discretizations into certain fixed points, seeking the subregion in which the functions can be separated into the largest number of clusters. A later review of all functions or derivatives in the same cluster enables the reassignment of functions that have been incorrectly classified because of their atypical local behavior. Such a global revision phase improves the cluster division, in the sense of making it more resistant to outliers, and ends each splitting step. The output of the DivClusFD method includes the number of clusters, the classification of the observed functions and guidelines that may be for interpreting the clusters. The DivClusFD method and some details of its practical implementation are introduced in Sect. 2. A simulation study with synthetic data is presented in Sect. 3, and the performance of the DivClusFD method on well-known real data sets is analyzed in Sect. 4; the results reported in both sections indicate that our method is able to group functions accurately. Conclusions are given in the last section.

2 The DivClusFD method of cluster construction

Let $\{X_1, \dots, X_n\}$ be a data set of n smooth real-valued functions defined on a compact set $[a, b]$, i.e., $X_i : [a, b] \rightarrow \mathbb{R}$. For $i = 1, \dots, n$, X_i can be seen as the realization of a stochastic process, and for each $t \in [a, b]$, $X_i(t)$ can be seen as the realization of a random variable $X(t)$ of dimension one. Let X_i^l denote the l -th derivative of the function X_i , and let $\{X_1^1, \dots, X_n^1\}$ and $\{X_1^2, \dots, X_n^2\}$ denote the sets of first and second derivatives, respectively.

For functional data sets in which there might be a useful unknown grouping into more than one cluster, two ideas motivate the appropriateness of a new hierarchical procedure for cluster construction that considers both the observed functions and their derivatives: (1) it may occur that no level differences are observed among functions belonging to different clusters in any subregion, yet the functions may have different shapes that can be recognized through differences in the levels of their derivatives, and (2) there may be no subregion of the functions (or their derivatives) in which all clusters are separated simultaneously.

The DivClusFD method begins by searching if the observed functions (or their derivatives) differ in subregions of $[a, b]$ and can provide a local clustering in such subregions; in particular we look for the subregion where the number of induced clusters (w.r. t. level or shape) is maximized. If the findings indicate a single group throughout the subregions of the functions and their derivatives, the search ends. Otherwise, there exist at least g clusters, where g is the largest number of clusters found. The partition $\{C_1, \dots, C_g\}$ of $\{1, \dots, n\}$ identifies the indices of the functions that belong to the same cluster. After assigning the functions to the clusters, the DivClusFD method looks for possible divisions within each cluster. The same procedure is repeated in each splitting step. That is, the DivClusFD method screens the functions

and their derivatives within each identified cluster to divide them into groups based on differences found in some subregion of the functions or their derivatives. This step is repeated over each new cluster until no evidence of further cluster division is found. This method of cluster construction can be considered to belong to the family of divisive hierarchical methods, although it shows some small differences from traditional hierarchical methods. In each splitting step of the DivClusFD method, each cluster can be divided into more than two clusters. Moreover, the result of the last division step does not provide as many clusters as there are data cases, as in traditional hierarchical clustering.

The practical implementation of the DivClusFD method described above requires specific criteria for the splitting of any cluster that includes functional data with indices in a class that we denote as $C_r = \{i_1, \dots, i_{n_r}\}$. These criteria are necessary both for finding the subregion with evidence of the largest number of clusters and for assigning the functional data to the newly identified clusters.

For finding the subregion with evidence of the largest number of clusters, a grid of N discrete points is defined in $[a, b]$, $a = t_1 < t_2 < \dots < t_{N-1} < t_N = b$. When we evaluate all functions in the grid, the result is a univariate sample of size n_r of $X(t_j)$, $\mathcal{X}_{j,0} = \{X_{i_1}(t_j), \dots, X_{i_{n_r}}(t_j)\}$. We might define two different grids for the derivatives; however, for simplicity, we evaluate the derivatives at the same points. Thus, $\mathcal{X}_{j,l} = \{X_{i_s}^l(t_j); s = 1, \dots, n_r\}$ is obtained as a univariate sample of $X^l(t_j)$, for $l = 1, 2$ and $j = 1, \dots, N$. The number of clusters in each $\mathcal{X}_{j,l}$ is estimated using the *gap statistic* (GS) (Tibshirani et al. 2001); the estimation procedure is summarized in Sect. 2.1. This method of estimation also allows us to establish a criterion for choosing, in the case of a tie, only one point t_j and one feature \hat{l} (the functions or the first or second derivatives) from among all those for which the estimated number of clusters is the largest. In Sect. 2.3, we present an algorithm for subdividing any cluster using the DivClusFD method, which includes the instructions that must be executed to make these selections.

To assign the functions to the new clusters, we first apply the k -means algorithm to the univariate sample $\mathcal{X}_{\hat{j},\hat{l}} = \{X_{i_1}^{\hat{l}}(t_{\hat{j}}), \dots, X_{i_{n_r}}^{\hat{l}}(t_{\hat{j}})\}$ to find a partition $\{C_{r_1}, \dots, C_{r_{\hat{k}}}\}$ of C_r , where \hat{k} is the largest estimated number of subclusters of C_r . Because this construction is based only on the local information at $t_{\hat{j}}$, we next introduce a global revision criterion that considers information obtained from the complete curves $\{X_{i_1}^{\hat{l}}, \dots, X_{i_{n_r}}^{\hat{l}}\}$ that are separated into the \hat{k} clusters by the partition $\{C_{r_1}, \dots, C_{r_{\hat{k}}}\}$. *Functional boxplots* (Sun and Genton 2011) are used to identify and reassign functions that have been incorrectly classified because of their atypical behavior at $t_{\hat{j}}$. The definition and construction of functional boxplots are summarized in Sect. 2.2, and the instructions that must be executed by the division algorithm to perform the outlier diagnosis can be found in Sect. 2.3.

2.1 Local estimation of the number of clusters with the gap statistic

Tibshirani et al. (2001) introduced the GS as a means of estimating the number of clusters in a data set. This procedure has several advantages. It can be used to estimate the number of groups based on any clustering strategy (we use the classical k -means

algorithm). It is designed to be able to detect single class configurations; this represents a major difference compared with most of the existing procedures for detecting the number of clusters, which assume that there are at least two clusters. For the one-dimensional case, the authors have proven some interesting theoretical properties.

In each splitting step of the DivClusFD method where a cluster $C_r = \{i_1, \dots, i_{n_r}\}$ is split into subclusters, we estimate the number of subclusters using the gap statistic for the one-dimensional samples $\mathcal{X}_{j,l} = \{X_{i_1}^l(t_j), \dots, X_{i_{n_r}}^l(t_j)\}$, for $l = 0, 1, 2$ and $j = 1, \dots, N$. Hence, although it may be used for more general applications, we describe the gap statistic only for one-dimensional samples and for the case in which cluster partitions are obtained by using the k -means algorithm for $\mathcal{X}_{l,j}$.

From now on, we consider the estimation of the number of clusters in only one of the $\mathcal{X}_{j,l}$ samples; the procedure can be immediately extended to the remaining samples by varying l and j . Let $\{C_{r_1}, \dots, C_{r_k}\}$ denote the partition of C_r into k clusters obtained using the k -means algorithm. Then, the pooled within-cluster sum of squares around the cluster means is given by

$$W_k \equiv W_k(j, l) = \sum_{m=1}^k \frac{1}{n_{r_m} - 1} \sum_{s \in C_{r_m}} \left(X_s^l(t_j) - \bar{X}_{r_m}^l(t_j) \right)^2, \tag{1}$$

where \bar{X}_{r_m} is the mean of the the values $X_s^l(t_j)$, with $s \in C_{r_m}$.

Tibshirani et al. (2001) based their work on the idea of assuming a single-component null model and rejecting it in favor of a model with $k > 1$ clusters, where k is the number of clusters with the strongest evidence. Their proposal was to compare $\log(W_k)$ with its expected value under an appropriate null reference distribution of the data and to estimate the number of clusters as the value \hat{k} for which $\log(W_{\hat{k}})$ falls farthest below this expectation. They defined

$$Gap_{n_r}(k) = E_{n_r}^*(\log(W_k)) - \log(W_k), \tag{2}$$

where $E_{n_r}^*$ denotes the expectation value for a sample of size n_r from the reference distribution. This expected value is computed via bootstrap resampling. In the one-dimensional case, Tibshirani et al. (2001) considered a uniform distribution as the reference distribution since among all unimodal distributions, the uniform distribution is the most likely to produce spurious clusters according to the gap test.

To estimate the number of clusters in $\mathcal{X}_{j,l} = \{X_{i_1}^l(t_j), \dots, X_{i_{n_r}}^l(t_j)\}$ with the gap statistic defined in (2), we first compute for $k = 1, \dots, K$ all partitions $\{C_{r_1}, \dots, C_{r_k}\}$ of C_r using the k -means algorithm and calculate W_k as shown in (1), where the boundary K must be fixed and is the maximally allowed number of clusters depending on the computational effort for each set of data. Then, we estimate $E_{n_r}^*(\log(W_k))$ for $k = 1, \dots, K$ via bootstrap resampling as follows:

- Generate B random samples of size n_r from a uniform distribution on the interval $[\min(\mathcal{X}_{j,l}), \max(\mathcal{X}_{j,l})]$, i.e., $X_{i_1,b}^{l*}(t_j), \dots, X_{i_{n_r},b}^{l*}(t_j)$ for $b = 1, \dots, B$.
- For each sample, compute $\log(W_{k,b}^*)$ for $k = 1, \dots, K$.
- Estimate $E_{n_r}^*(\log(W_k))$ as $(1/B) \sum_{b=1}^B \log(W_{k,b}^*)$ for $k = 1, \dots, K$.

The estimated gap statistic is given by

$$Gap_{n_r}(k) = \frac{1}{B} \sum_{b=1}^B \log(W_{k,b}^*) - \log(W_k), \quad \text{for } k = 1, \dots, K.$$

The estimated number of clusters in $\mathcal{X}_{j,l} = \{X_{i_1}^l(t_j), \dots, X_{i_{n_r}}^l(t_j)\}$ according to the gap statistic, $\hat{k}(j, l)$, is the smallest k such that

$$Gap_{n_r}(k) \geq Gap_{n_r}(k + 1) - n_{sd} \sqrt{1 + \frac{1}{B} sd(k + 1)}, \tag{3}$$

where $sd(k + 1)$ is the standard deviation of the B replicates $\log(W_{k+1,b}^*)$ and n_{sd} is a factor that measures the deviation (in terms of the standard deviation), which must be prespecified. Tibshirani et al. (2001) suggested $n_{sd} = 1$, but stronger rules can be considered.

2.2 Functional boxplots for outlier detection in local clusters

Sun and Genton (2011) proposed the functional boxplot (FB) as an extension of the classical one-dimensional boxplot for exploratory functional data analysis. For the construction of a functional boxplot, they used the typical tools of descriptive statistics, adapted to the particularities of functional data. A functional boxplot consists of an envelope representing the central 50% of the functions, the median function and the maximum non-outlying envelope. Finding the central region and the median function requires a definition for the ordering of functions. Although there is no complete order in any space of functions, depth measures are appropriate for ordering functions from the center outward. Sun and Genton suggested using the band depth, which is a depth measure for functional data that was introduced by Lopez-Pintado and Romo (2009).

As in the one-dimensional case, functional boxplots may highlight possible atypical functions. This property is used in each splitting step of the DivClusFD method to detect whether any function $\{X_{i_1}^l, \dots, X_{i_{n_r}}^l\}$ from C_r is incorrectly classified with regard to the \hat{k} clusters given by the partition $\{C_{r_1}, \dots, C_{r_{\hat{k}}}\}$ of C_r . A function is considered to be misclassified when it looks like the other functions in its own cluster at $t_{\hat{j}}$ but is more similar to the functions from a different cluster in the rest of the domain.

To simplify the notation, we here describe the construction of a functional boxplot for a generic sample of n functions $\{Y_1, \dots, Y_n\}$. The detection of potential outliers using functional boxplots proceeds as follows:

- Compute the band depths for the sample functional data Y_1, \dots, Y_n and sort the functions in decreasing order. Let $Y_{[1]}, \dots, Y_{[n]}$ denote the functions ordered according to decreasing band-depth, where $Y_{[1]}$ is the median (deepest) function and $Y_{[n]}$ is the most outlying function (with the lowest band depth).
- Estimate the central 50% region for the functions as the band (in \mathbb{R}^2) delimited by the functions $Y_{[1]}, \dots, Y_{[M]}$, where $M = \lceil \frac{(n+1)}{2} \rceil$ is the smaller integer not less than $n/2$; this band is given by

$$C_{0.5} = \{(t, y(t)) : t \in [a, b], \min_{r=1, \dots, M} Y_{[r]}(t) \leq y(t) \leq \max_{r=1, \dots, M} Y_{[r]}(t)\}.$$

This region is analogous to the inter-quantile range in the univariate case, and the edges form the envelope of $C_{0.5}$, which corresponds to the box in the functional boxplot.

- Any function that, for at least one t , lies outside the inflated envelope obtained by multiplying the range of $C_{0.5}$ by 1.5 is usually considered a potential outlier. In practice, the factor 1.5 can be adjusted to a higher value to be more conservative with the non-outlying null hypothesis.

In each splitting step of the DivClusFD method, we consider any function $X_u^{\hat{l}}$, where $u \in C_{r_s} \subset \{C_{r_1}, \dots, C_{r_r}\}$, to be potentially incorrectly classified if it is identified as a potential outlier in the functional boxplot computed for all functions with indices in C_{r_s} , using an inflation factor of 3 for the envelope of $C_{0.5}$, following the classical rule to identify severe outliers with outer fences (Tukey 1977).

In the case that $X_u^{\hat{l}}$ is a potential outlier, we compute the \hat{k} different functional boxplots for the functions with indices in $\{C_{r_1}, \dots, C_{r_k}\}$, and we finally assign $X_u^{\hat{l}}$ to the cluster for which it is most often inside the corresponding inflated envelope of $C_{0.5}$.

2.3 Algorithm for the division of clusters in the DivClusFD method

The DivClusFD method starts with a single group that includes all functions with indices in $C = \{1, \dots, n\}$, and it iteratively looks for all possible divisions and subdivisions necessary for identifying the complete cluster structure in a functional data set. The splitting process terminates when there is no evidence of any further cluster structure within any group.

Each splitting step of the DivClusFD method is executed by the following algorithm, which runs in two phases. In the first phase, a local search for clusters is performed (local phase), and in the second phase, the functions are assigned to the clusters based on the observed functions or their derivatives throughout the entire domain (global phase). The algorithm can be applied for splitting the cluster $C_r = \{i_1, \dots, i_{n_r}\}$.

Local phase (cluster-structure search):

- Define a grid of discrete points in the interval $[a, b]$, $a = t_1 < t_2 < \dots < t_{N-1} < t_N = b$.
- Evaluate all functions and first and second derivatives with indices in C_r at the grid points to obtain $X_{i_s}^l(t_j)$ for $s = 1, \dots, n_r$, $j = 1, \dots, N$ and $l = 0, 1, 2$.
- For each possible univariate sample $\mathcal{X}_{j,l} = \{X_{i_1}^l(t_j), \dots, X_{i_{n_r}}^l(t_j)\}$, where $j = 1, \dots, N$ and $l = 0, 1, 2$, estimate the number of clusters $\tilde{k}(j, l)$ using the gap statistic as explained in Sect. 2.1.
- Estimate the number of clusters in $\{X_{i_1}^l, \dots, X_{i_{n_r}}^l\}$ as $\hat{k} = \max_{j,l} \tilde{k}(j, l)$.
- Find \hat{j} and \hat{l} such that $\tilde{k}(\hat{j}, \hat{l}) = \hat{k}$. In the case of a tie, choose the values of j and l that maximize $\log(W_{\hat{k}}(j, l)) - \log(W_{\hat{k}+1}(j, l))$, where $W_{\hat{k}}(j, l)$ is given in (1).
 - If $\hat{k} = 1$, then the group $C_r = \{i_1, \dots, i_{n_r}\}$ is considered to form a single cluster.

- If $\hat{k} \geq 2$, use the constructed \hat{k} subclusters for the following global phase that builds the final clusters.

Global phase (assignment of functions to clusters):

- Let $\{C_{r_1}, \dots, C_{r_{\hat{k}}}\}$ denote the partition of C_r into \hat{k} clusters obtained by applying the k -means algorithm to the univariate sample $\mathcal{X}_{\hat{j}, \hat{l}} = \{X_{i_1}^{\hat{l}}(t_{\hat{j}}), \dots, X_{i_{n_r}}^{\hat{l}}(t_{\hat{j}})\}$. Assign the functions $\{X_{i_1}^{\hat{l}}, \dots, X_{i_{n_r}}^{\hat{l}}\}$ to the \hat{k} clusters based on their indices in the partition $\{C_{r_1}, \dots, C_{r_{\hat{k}}}\}$.
- Compute the \hat{k} functional boxplots, $FB_{r_1}, \dots, FB_{r_{\hat{k}}}$, for those functions among $\{X_{i_1}^{\hat{l}}, \dots, X_{i_{n_r}}^{\hat{l}}\}$ whose indices are in $C_{r_1}, \dots, C_{r_{\hat{k}}}$, respectively, following [Sun and Genton \(2011\)](#) and as summarized in Sect. 2.2.
- For each $s = 1, \dots, \hat{k}$, check for potential outliers among all functions with indices $u \in C_{r_s} \subset \{C_{r_1}, \dots, C_{r_{\hat{k}}}\}$.
 - If $X_u^{\hat{l}}$ is a potential outlier in FB_{r_s} as explained in Sect. 2.2, let $FB_{r_{s'}}$ denote the functional boxplot for which $X_u^{\hat{l}}$ most often lies inside the corresponding inflated envelope of $C_{0.5}$.
 - If $s' \neq s$, remove u from C_{r_s} and add it to the class $C_{r_{s'}}$.
 - Otherwise, u remains in C_{r_s} .
 - If $X_u^{\hat{l}}$ is not a potential outlier in FB_{r_s} , u remains in C_{r_s} .

The application of this algorithm to the functional data $\{X_{i_1}, \dots, X_{i_{n_r}}\}$ provides estimates of the number of clusters \hat{k} and the partition $\{C_{r_1}, \dots, C_{r_{\hat{k}}}\}$ of $\{i_1, \dots, i_{n_r}\}$ with the indices for assigning the functions to the \hat{k} clusters. Additionally, the algorithm also provides information that may be useful for interpreting the clusters: the point $t_{\hat{j}}$ and the feature \hat{l} (functions, first or second derivatives) for which the clusters are best separated.

2.4 Issues regarding the practical implementation of the DivClusFD method

Considering again the example shown in Fig. 1, we find that the DivClusFD method first separates the third group using the derivatives. In the second iteration, it separates the first two groups using the trajectories, and it then ends the splitting process because no further groups are found. The DivClusFD method succeeds in finding the cluster structure of the three groups, and all functions are correctly classified. [Ieva et al. \(2013\)](#) proposed a k -means clustering method for functional data that is also based on information about the functions and their derivatives. Instead of using the classical L^2 norm, their method works in the Sobolev H^1 space with the ordinary norm. Upon applying the method of [Ieva et al. \(2013\)](#) to the same example and estimating the number of clusters using either the gap statistic or the Calinski–Harabasz approach, which are among the best-known procedures for estimating the number of clusters, it is observed that neither of them detects the correct number of clusters.

In more realistic problems, the computation of the first two derivatives is not always easy. This is the case when, in a set of functional data, the functions are observed at a

number of consecutive points that are not equidistant or are separated with different frequencies for each function. When the functions are observed without sampling errors, they can be interpolated. Otherwise, it is necessary to use smoothing techniques to transform the observations into functions that can be evaluated at any point t_j . Ramsay et al. (2009) discussed the properties of the main smoothing techniques for functional data. Then the derivatives can be easily computed through differentiation in both cases.

When the functions are observed without sampling errors, they can be interpolated. Otherwise, it is necessary to use smoothing techniques to transform the observations into functions that can be evaluated at any point t_j . Ramsay et al. (2009) discussed the properties of the main smoothing techniques for functional data. Then the derivatives can be easily computed through differentiation in both cases.

The estimation of the number of clusters during the local phase of the algorithm presented in Sect. 2.3 can be computationally expensive because for each discrete point on the grid (t_1, \dots, t_N) and each feature of the functions ($l = 0, 1, 2$), the limits of the uniform distributions to be generated for calculating the gap statistics using (3) are likely to be different. Thus, the finer the grid is, the more computationally expensive the local phase will be. Without modifying the grid, we can make the algorithm faster by generating the uniform samples more efficiently. For each univariate sample $\mathcal{X}_{j,l} = \{X_{i_1}^l(t_j), \dots, X_{i_{n_r}}^l(t_j)\}$, we suggest rescaling the data to the interval $[0, 1]$ as follows:

$$Z_{i_s}^l(t_j) = \frac{X_{i_s}^l(t_j) - \min\{\mathcal{X}_{j,l}\}}{\max\{\mathcal{X}_{j,l}\} - \min\{\mathcal{X}_{j,l}\}}, \quad \text{for } s = 1, \dots, n_r.$$

Now, all uniform samples will be generated in the interval $[0, 1]$. This means that the bootstrap resampling of Sect. 2.1 must be performed only once in each splitting step; however, the same resampling cannot be used in all splitting steps because the size of the uniform sample depends on n_r , which changes with each new cluster subdivision.

3 Simulation study

To illustrate the performance of the DivClusFD method, we conducted a simulation study using various artificial data sets that have been previously presented in the literature for the clustering of functional data. In all cases, we report the number of times that DivClusFD selects the correct number of groups. For these successful examples, the correct classification rates (CCRs) are estimated with respect to the known partitions.

Before applying the DivClusFD method, it is necessary to fix the values of some parameters related to the gap statistic. In general, we followed the suggestions in Tibshirani et al. (2001), except for n_{sd} in (3). In each splitting step, we set $n_{sd} = 3$ because we prefer to be more conservative, with a null hypothesis of a smaller number of clusters. Because DivClusFD iteratively searches for the cluster structure, we prefer to underestimate the number of clusters in each step because if a cluster division is missed in one step, it will be found in future iterations. It is not necessary to use more than 500 bootstrap samples to compute (3). The maximum number of partitions

calculated using the k -means algorithm is $K = 5$. This default option can be increased until (3) is satisfied.

In the global phase for the assignment of functions to clusters, a minimum of 10 functions is required to compute a functional boxplot. The minimum cluster size to continue dividing a cluster was also set to 10 functions. We used the same values for all simulations and real-data examples.

3.1 Sampling errors in the functions

Sangalli et al. (2010a) introduced three models for curve generation. We followed their models for data generation with the aim of examining the performance of the Div-ClusFD method on cluster structures with different amplitudes and curve registration problems.

Model A: Two clusters of $n/2$ functions each, generated as follows:

$$\begin{aligned} X_i(t) &= (1 + \epsilon_{1i}) \sin(\epsilon_{3i} + \epsilon_{4i}t) \\ &\quad + (1 + \epsilon_{2i}) \sin\left(\frac{(\epsilon_{3i} + \epsilon_{4i}t)^2}{2\pi}\right) \\ &\quad \text{for } i = 1, \dots, n/2, \end{aligned} \quad (4)$$

$$\begin{aligned} X_i(t) &= (1 + \epsilon_{1i}) \sin(\epsilon_{3i} + \epsilon_{4i}t) \\ &\quad - (1 + \epsilon_{2i}) \sin\left(\frac{(\epsilon_{3i} + \epsilon_{4i}t)^2}{2\pi}\right) \\ &\quad \text{for } i = n/2 + 1, \dots, n. \end{aligned} \quad (5)$$

Model B: Two clusters of $n/2$ functions each, with the first group generated following (4) and the second group generated as follows:

$$\begin{aligned} X_i(t) &= (1 + \epsilon_{1i}) \sin\left(\epsilon_{3i} + \epsilon_{4i} \left(-\frac{1}{3} + \frac{3}{4}t\right)\right) \\ &\quad - (1 + \epsilon_{2i}) \sin\left(\frac{(\epsilon_{3i} + \epsilon_{4i} \left(-\frac{1}{3} + \frac{3}{4}t\right))^2}{2\pi}\right) \\ &\quad \text{for } i = n/2 + 1, \dots, n. \end{aligned} \quad (6)$$

Model C: Three clusters of $n/3$ functions each, with the first group generated following (4), the second group generated following (5) and the third group generated following (6).

We simulated 200 data sets of size $n = 90$ for each model. All errors $\epsilon_{1i}, \dots, \epsilon_{4i}$ were independent and normally distributed with a mean of 0 and a standard deviation of 0.05. Figure 2 shows three examples of sets of 90 curves generated using models A, B and C.

For each model, Table 1 reports the percentage of data sets for which DivClusFD finds the number of clusters indicated in the first column. In almost all cases, Div-

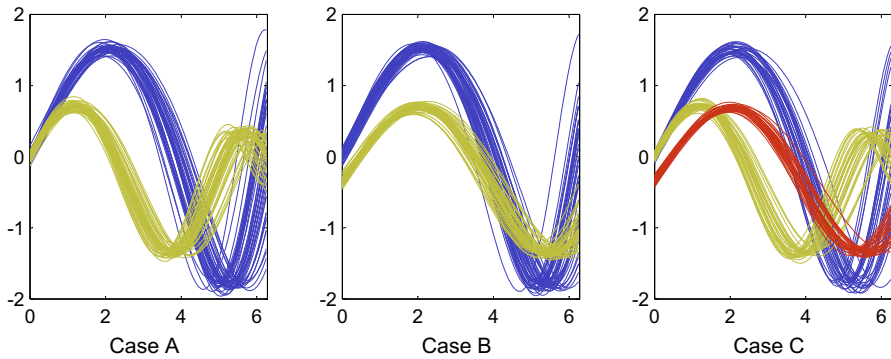


Fig. 2 Examples of data sets consisting of functions simulated using models A, B and C

Table 1 Distributions of the number of groups found by DivClusFD and the corresponding mean CCRs

Number of clusters	Model A	Model B	Model C
2	88	99.5	0
3	12	0.05	75.5
4	0	0	23.5
5	0	0	1
Mean CCR	99.67	99.96	99.45

ClusFD finds the correct number of clusters. The result is poorer for model C, which also presents the most challenging problem, with 75.5% of the identifications being successful. When the number of clusters is not identified correctly, an additional cluster is found in most cases. We observed that in these cases, one of the original clusters is erroneously divided into two and the other clusters are identified without error. Table 1 also presents the mean CCRs, calculated using only data sets for which the number of clusters is correctly identified. It is clear that DivClusFD demonstrates outstanding performance in these cases.

Finally, we compare our procedure with other clustering methods for functional data that are available in R, namely, funclust (Jacques and Preda 2013), multivariate funclust (Jacques and Preda 2014a), funHDDC (Bouveyron and Jacques 2011) and kmeans.fd, which is a *k*-means procedure for functional data that is included in the R package *fda.usc* (Febrero-Bande et al. 2012). In all cases, the number of clusters must be specified beforehand. Table 2 presents the mean CCRs for these four procedures.

Table 2 Mean CCRs for funclust, multivariate funclust, kmeans.fd and funHDDC

	Model A	Model B	Model C
funclust	75.62	73.57	48.48
multivariate funclust	84.29	81.23	60.9
kmeans.fd	67	76.77	68.45
funHDDC	85.81	47.39	37.86

It is clear that DivClusFD outperforms all four clustering strategies, as it has a higher CCR for each model.

3.2 Sampling errors in the observations

We simulated 200 data sets, each consisting of four clusters that were generated using a model similar to that introduced by Serban and Wasserman (2005) . Each cluster contains 150 functions, generated by

$$X_{ij}(t) = f_j(t) + \epsilon_i(t),$$

for $t \in [0, 1], i = 1, \dots, 150$ and $j = 1, \dots, 4,$

where

$$f_1(t) = \min \left(\frac{2 - 5t}{2}, \left(\frac{2 - 5t^2}{2} \sin \left(\frac{5\pi t}{2} \right) \right) \right),$$

$$f_2(t) = -f_1(t), \quad f_3(t) = \cos(2\pi t) \quad \text{and} \quad f_4(t) = -f_4(t).$$

Serban and Wasserman (2005) assume independent errors, while we assume correlated errors from a Gaussian process. For all the functions, $\epsilon(t)$ is normally distributed with mean of 0.4, a standard deviation of 0.9 and a covariance structure given by

$$\rho(s, t) = 0.3 \exp \left(-\frac{(s - t)^2}{0.3} \right), \quad \text{for } s, t \in [0, 1].$$

Figure 3a, b show one of the generated data sets and the corresponding data set obtained after smoothing with *B*-splines, respectively. Each color represents one theoretical cluster. Figure 3c, d display the first two derivatives of the functions. To avoid boundary effects, we reflected one third of the observations at the beginning and end

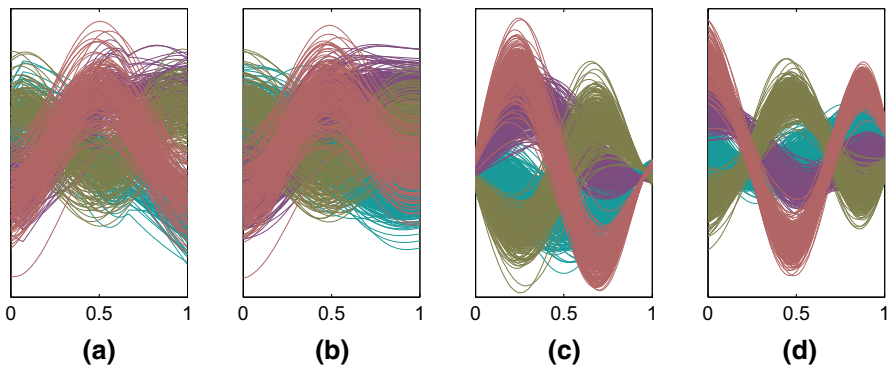


Fig. 3 **a** A simulated data set with sampling errors in the observations; **b** the smoothed data set; **c** the first derivatives; and **d** the second derivatives

Table 3 Mean CCRs for funclust, multivariate funclust, kmeans.fd, funHDDC and DivClusFD

Clustering procedure	Mean CCR
funclust	40.05
multivariate funclust	43.74
kmeans.fd	60.12
funHDDC	63.21
DivClusFD	99.85

of each function. We again compare our method with other clustering procedures for functional data: funclust, multivariate funclust, funHDDC and kmeans.fd. For these other methods, the number of clusters was given as an input. Table 3 presents the mean CCRs for the 200 data sets. It is important to emphasize that DivClusFD always identifies four groups and has a much higher mean CCR than the other procedures, achieving perfect classification for most of the replicates.

4 Real-data examples

4.1 Berkeley Growth Study data

The Berkeley Growth Study is one of the best-known long-term development investigations ever conducted, and the height growth data set introduced by [Tuddenham and Snyder \(1954\)](#) has been used as a reference to evaluate various methods of functional data analysis. This data set, consisting of the heights of 54 girls and 39 boys measured between 1 and 18 years of age at 31 unequally spaced time points, is considered to be one of the most challenging data sets for clustering purposes. More measurements were taken during the later years of childhood and adolescence, when growth was more rapid, and fewer during the early years, when growth was more stable. Consequently, to compute the first two derivatives, we needed to transform the observations into functions that could be evaluated at any point in time. We employed a monotonic cubic regression spline smoothing, as suggested by [Ramsay et al. \(2009\)](#). Figure 4a shows all of the functions without gender identification to offer a clear idea of the difficulties that any clustering method will face when confronted with this data set. The derivatives, shown in Fig. 4b, do not present a more tractable scenario.

Our first objective was to test the effectiveness of the DivClusFD method in determining the number of groups in this data set without regard to gender information. Second, we used the output of DivClusFD to classify the children. The information obtained from the splitting steps provides guidance for understanding the differences among the clusters.

Using the same parameters as in the simulation study, the DivClusFD method identifies two clusters. In the local phase, the algorithm presented in Sect. 2.3 finds the evidence in the first derivative (growth rate) at an age of 14, which is coincident with the end of puberty in girls but not in boys. In the final classification of the data, one cluster is coincident with boys (orange curves in Fig. 4c–f) and the other with girls (green curves in Fig. 4c–f). Only the 10 functions highlighted with thicker lines are

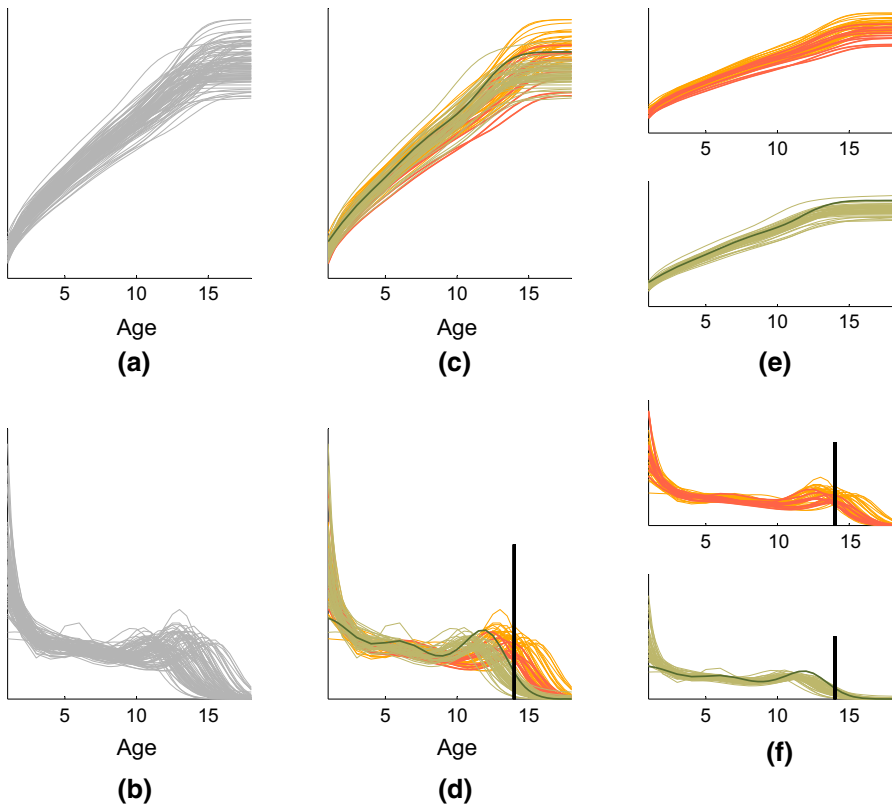


Fig. 4 **a, b** Height curves and their derivatives, respectively, of 54 girls and 39 boys measured between 1 and 18 years of age; **c, d** original and derivative functions, respectively, of the two clusters identified using the DivClusFD method (the *curves* in the cluster of “mostly boys” are shown in *orange*, and the *curves* in the cluster of “mostly girls” are shown in *green*); **e, f** original and derivative functions, respectively, of the two clusters that include most of the boys (*top*) and most of the girls (*bottom*). The *thicker lines* represent curves that are misclassified by DivClusFD (color figure online)

misclassified. They correspond to nine girls with late puberty and one boy with early maturity. The CCR is 89.25%.

This data set has recently been used by Jacques and Preda (2014a) to compare several clustering methods for functional data. All analyzed methods make use of the prior information that the number of clusters is two, which could suggest that a priori, DivClusFD should be less competitive. However, the CCRs for the other methods are not always superior: funclust-CCR = 69.89% (Jacques and Preda 2014a), multivariate funclust-CCR = 53.76% (Jacques and Preda 2014a), funHDDC-CCR = 96.77% (Bouveyron and Jacques 2011), fclust-CCR = 69.89% (James and Sugar 2003), and kCFC-CCR = 93.55% (Chiou and Li 2011). The results of DivClusFD are closer to those of the most successful methods (funHDDC and kCFC) than to funclust and fclust. In addition, the output of DivClusFD also includes an estimate of the number of groups and provides assistance in interpreting the clusters.

Sangalli et al. (2010b) also analyzed this data set. Although they found more evidence for the existence of a single cluster, they analyzed the case of two clusters, for which the best CCR obtained was 88.17%.

4.2 ECG200 data

The 200 electrocardiograms in the ECG200 data set can be found on the UCR Time Series Classification and Clustering website (Chen et al. 2015). The data set consists of two groups, one with 133 and the other with 67 electrocardiograms, each one recorded at 96 equally spaced instants. The left side of Fig. 5 shows the electrocardiograms (f) and their first (f') and second (f'') derivatives for both clusters (orange and blue functions). This data set has been analyzed by Jacques and Preda (2014a), among others, using the same clustering procedures as in the previous example except for kCFC. The results are funclust-CCR = 84%, multivariate funclust-CCR = 60%, funHDDC-CCR = 75%, and fclust-CCR = 74.5%.

The tree structure in the central part of Fig. 5 shows the results of the splitting steps followed by the DivClusFD method to find the cluster structure, using the same parameters as in the simulation study. In the first splitting step, DivClusFD detects two clusters using the information provided by the electrocardiograms at $t = 43$. In the second splitting step, the maximum number of clusters estimated in the first cluster is one; therefore, it is a terminal node. The majority (35) of the functions in this cluster are blue, and 19 functions are orange. In the next splitting step, the other cluster is divided into two clusters at $t = 41$, based on the information from the first derivatives. One cluster is a terminal node with 99 orange functions and only 7 blue; the latter may be misclassifications. In the last splitting step, the remaining functions

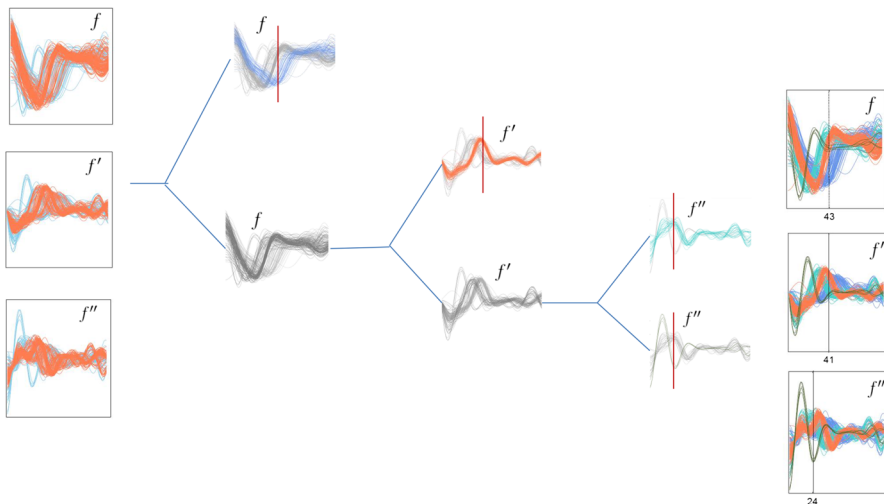


Fig. 5 On the *left panel*, the electrocardiograms (f) and their first (f') and second (f'') derivatives for the two clusters (*blue and orange*) of the ECG200 data set; in the *central panel*, the results of the splitting steps performed by the DivClusFD method; on the *right panel*, the final classification obtained by DivClusFD (color figure online)

are separated into two clusters at $t = 24$, based on the information from the second derivatives. The green functions are a terminal cluster with 21 blue functions and 15 orange. Finally, the last four functions cannot be separated because of the small size of the group (under 10). When we analyze these functions, we find that they can be considered outliers. Some authors eliminate these electrocardiograms at the beginning of the analysis; see, for instance, [Jacques and Preda \(2014a\)](#).

The right side of Fig. 5 shows f , f' and f'' for the final three clusters and the four outliers. The four outliers appear in dark green. Although this data set contains only two clusters and we identify three, it can be seen from the results that the cluster found in the second splitting step can be considered the same as the original cluster of orange curves, with 7 functions that are misclassified. Under the assumption that the original cluster of blue electrocardiograms is the union of the other two (blue and green), the CCR is 79.08%. For this calculation, we have excluded the four outliers to compare the results of DivClusFD with those of [Jacques and Preda \(2014a\)](#).

The classification obtained by DivClusFD is very good in comparison with those of the other methods. Although DivClusFD-CCR is not the highest CCR, note that the other methods start from an advantageous position by assuming the known number of clusters. DivClusFD is outperformed only by *funclust*.

It is important to remark that if the global phase is ignored, the clustering structure changes as follows: (1) in the first splitting step the terminal node has one more function; (2) in the second splitting step, the terminal node has 22 additional functions; (3) the final clustering structure is different and the CCR of the procedure ignoring the global phase decrease to 77.5%.

4.3 Italy Power Demand data

The Italy Power Demand data set can also be found on the UCR Time Series Classification and Clustering website ([Chen et al. 2015](#)). This data set contains two clusters: cluster 1, with 513 power demand functions (see Fig. 6a), and cluster 2, with 516 functions (see Fig. 6b). Note that there are two different patterns of consumption in each cluster. All functions were observed at 24 equally spaced time points throughout a day. The data were smoothed with six equally spaced knot splines (choosing more knots led to similar results). To prevent boundary effects, we reflected one third of the observations at the beginning and end of each function. DivClusFD was executed with the same parameters as in the previous examples.

The DivClusFD method finds, through two splitting steps, the four clusters shown in the tree in Fig. 7, identifying both the original clusters and the two different patterns of consumption within them. All partitions are based on information provided by the first derivatives, at $t = 7$, $t = 23$ and $t = 9$. The final classification is shown on the right side of Fig. 7. The orange functions constitute cluster 1, whereas the green functions make up cluster 2. The darker green and orange functions are misclassified power demand functions. Upon collapsing the four clusters into only two, we can calculate a CCR of 93.49%.

From the hours and the function feature by which the clusters are identified, it can be observed that the clusters are characterized by the different rates at which consumption

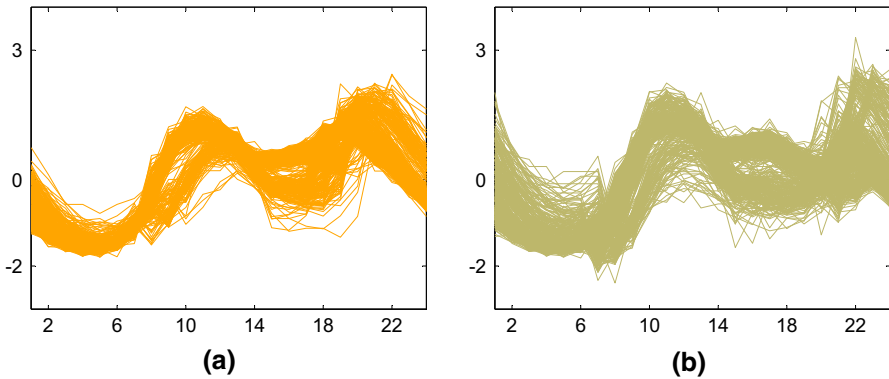


Fig. 6 Power demand functions throughout a day in Italy: **a** the 513 functions in cluster 1 and **b** the 516 functions in cluster 2

increases in the morning (7–9 h) and decreases at the end of the day (23 h). Similar patterns are observed in data from home consumers of electric power in the City of Buenos Aires, Argentina (see [Fraiman et al. 2008](#)).

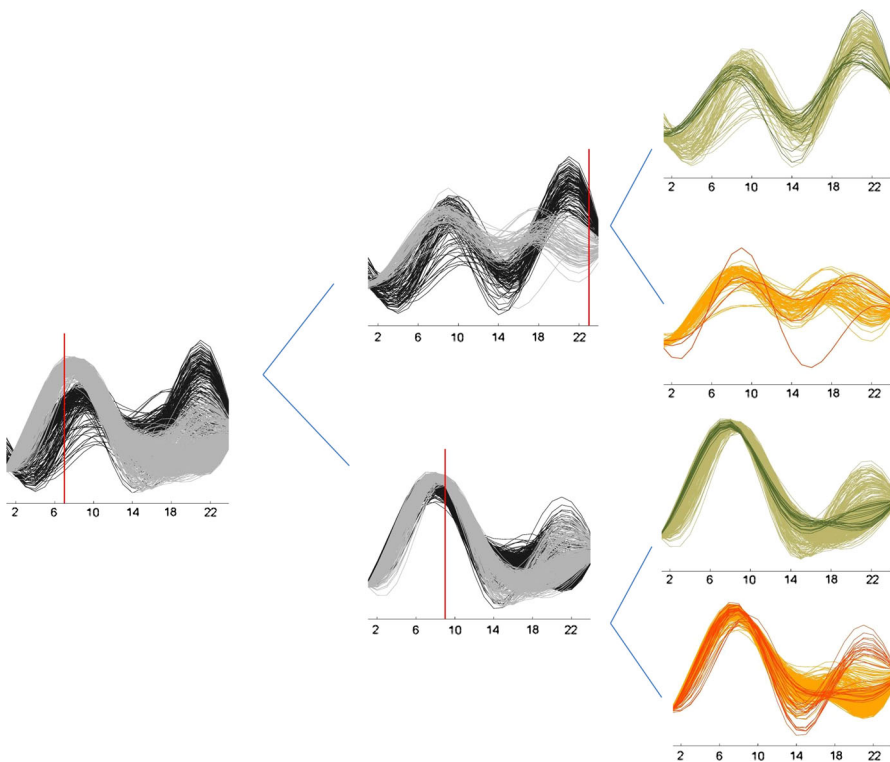


Fig. 7 Results of the two splitting steps performed by DivClusFD to classify functions describing power demand in Italy. The first derivatives are shown for all iterations

Finally, we challenged our procedure with the same clustering procedures that we considered in Sect. 3. The results are $\text{funclust-CCR} = 51.02\%$, $\text{multivariate funclust-CCR} = 51.6\%$, $\text{funHDDC-CCR} = 53.16\%$, and $\text{kmeans.fd-CCR} = 55.2\%$.

5 Conclusion

In this paper, we propose the DivClusFD method, a new divisive hierarchical clustering method for functional data. It simultaneously provides the number of clusters and the classification of the observed functions. Because we are interested in identifying cluster structures that are related to not only the levels of the functions but also their shapes, the DivClusFD method also analyzes the first and second derivatives of the functions. DivClusFD is a divisive procedure that iteratively splits a sample into clusters by searching for the points on a grid, defined during the local phase of the clustering algorithm for the functions and their derivatives, that offer the highest clustering capability according to the gap statistic criterion. Functional boxplots for each identified group are used for the reallocation of possibly misclassified data. Moreover, DivClusFD provides helpful information regarding the group structure.

Although DivClusFD is based on the one-dimensional gap statistic, alternative methods of estimating the number of groups could be considered. Similarly, the functional boxplots could be adapted to consider different definitions of depth for functional data. Several proposals can be found in Mosler (2013).

The algorithm could easily be extended to the multivariate case, in which each datum is a vector of functions (see Berrendero et al. 2011). In the local phase, any clustering method could be considered, and in the global phase, the functional boxplots could be designed using the band depth proposed by Lopez-Pintado et al. (2014) for multivariate functional data. The efficiency of the method is expected to decrease with increasing dimensionality of the data.

When we compare the results of DivClusFD with those of other methods on real examples, we observe that DivClusFD obtains solutions of very similar quality for different sets of data, whereas well-known methods such as funclust and funHDDC exhibit somewhat irregular behavior. For instance, in the growth example FunHDDC is the best algorithm, while in EGC200 is almost the worst, and the opposite happens with funclust.

Finally, the output of the algorithm includes the number of groups and the clustering allocation. In addition, it provides information about the *key* points $t \in [a, b]$ and whether the cluster divisions are based on functions or their derivatives (or both). This information yields a better understanding of the cluster structure.

Acknowledgements We wish to thank the editors and four anonymous referees who have carefully reviewed the paper. Their suggestions and comments have helped us to improve the quality of this paper.

References

Abraham C, Cornillon PA, Matzner-Lber E, Molinari N (2003) Unsupervised curves clustering using B-Splines. *Scand J Stat* 30:581–595. doi:[10.1111/1467-9469.00350](https://doi.org/10.1111/1467-9469.00350)

- Alonso AM, Casado D, Romo J (2012) Supervised classification for functional data: a weighted distance approach. *Comput Stat Data Anal* 56:2334–2346. doi:[10.1016/j.csda.2012.01.013](https://doi.org/10.1016/j.csda.2012.01.013)
- Berrendero JR, Justel A, Svarc M (2011) Principal components for multivariate functional data. *Comput Stat Data Anal* 55:2619–2634. doi:[10.1016/j.csda.2011.03.011](https://doi.org/10.1016/j.csda.2011.03.011)
- Bouveyron C, Jacques J (2011) Model-based clustering of time series in group-specific functional subspaces. *Adv Data Anal Classif* 5:281–300. doi:[10.1007/s11634-011-0095-6](https://doi.org/10.1007/s11634-011-0095-6)
- Chen Y, Keogh E, Hu B, Begum N, Bagnall A, Mueen A, Batista G (2015) The UCR time series classification archive. http://www.cs.ucr.edu/~eamonn/time_series_data/
- Chiou JM, Li PL (2011) Functional clustering and identifying substructures of longitudinal data. *J R Stat Soc B* 69:679–699. doi:[10.1111/j.1467-9868.2007.00605.x](https://doi.org/10.1111/j.1467-9868.2007.00605.x)
- Febrero-Bande M, Oviedo de la Fuente M (2012) Statistical computing in functional data analysis: the R Package *fda.usc*. *J Stat Softw* 51(4):1–28. doi:[10.18637/jss.v051.i04](https://doi.org/10.18637/jss.v051.i04)
- Fraiman R, Justel A, Svarc M (2008) Selection of variables for cluster analysis and classification rules. *J Am Stat Assoc* 103:1294–1303. doi:[10.1198/016214508000000544](https://doi.org/10.1198/016214508000000544)
- Ieva F, Paganoni AM, Pigoli D, Vitelli V (2013) Multivariate functional clustering for the morphological analysis of electrocardiograph curves. *J R Stat Soc Ser C (Appl Stat)* 62:401–418. doi:[10.1111/j.1467-9876.2012.01062.x](https://doi.org/10.1111/j.1467-9876.2012.01062.x)
- Jacques J, Preda C (2013) Funclust: a curves clustering method using functional random variable density approximation. *Neurocomputing* 171:112–164. doi:[10.1016/j.neucom.2012.11.042](https://doi.org/10.1016/j.neucom.2012.11.042)
- Jacques J, Preda C (2014a) Model-based clustering for multivariate functional data. *Comput Stat Data Anal* 71:92–106. doi:[10.1016/j.csda.2012.12.004](https://doi.org/10.1016/j.csda.2012.12.004)
- Jacques J, Preda C (2014b) Functional data clustering: a survey. *Adv Data Anal Classif* 8(3):231–255. doi:[10.1007/s11634-013-0158-y](https://doi.org/10.1007/s11634-013-0158-y)
- James G, Sugar C (2003) Clustering for sparsely sampled functional data. *J Am Stat Assoc* 98:397–408. doi:[10.1198/016214503000189](https://doi.org/10.1198/016214503000189)
- Keogh E, Pazzani M (2001) Dynamic time warping with higher order features. In: First SIAM international conference on data mining (SDM'2001), Chicago, IL, USA. doi:[10.1007/s10618-015-0418-x](https://doi.org/10.1007/s10618-015-0418-x)
- López-Pintado S, Romo J (2009) On the concept of depth for functional data. *J Am Stat Assoc* 104:718–734. doi:[10.1198/jasa.2009.0108](https://doi.org/10.1198/jasa.2009.0108)
- López-Pintado S, Sun Y, Lin JK, Genton MG (2014) Simplicial band depth for multivariate functional data. *Adv Data Anal Classif* 8:321–338. doi:[10.1007/s11634-014-0166-6](https://doi.org/10.1007/s11634-014-0166-6)
- Mosler K (2013) Depth statistics. In: Becker C, Fried R, Kuhnt S (eds) *Robustness and complex data structures*. Springer, Berlin, pp 17–34. doi:[10.1007/978-3-642-35494-6_2](https://doi.org/10.1007/978-3-642-35494-6_2)
- Ray S, Mallick B (2006) Functional clustering by Bayesian wavelet methods. *J R Stat Soc Ser B Stat Methodol* 68(2):305–332. doi:[10.1111/j.1467-9868.2006.00545.x](https://doi.org/10.1111/j.1467-9868.2006.00545.x)
- Ramsay J, Hooker G, Graves S (2009) *Functional data analysis with R and Matlab*. Springer, Berlin. doi:[10.1007/978-0-387-98185-7](https://doi.org/10.1007/978-0-387-98185-7)
- Ramsay J, Silverman BW (2005) *Functional data analysis*, 2nd edn. Springer, New York. doi:[10.1111/j.1541-0420.2007.00743_1.x](https://doi.org/10.1111/j.1541-0420.2007.00743_1.x)
- Sangalli LM, Secchi P, Vantini S, Vitelli V (2010a) *k*-means alignment for curve clustering. *Comput Stat Data Anal* 54:1219–1233. doi:[10.1016/j.csda.2009.12.008](https://doi.org/10.1016/j.csda.2009.12.008)
- Sangalli LM, Secchi P, Vantini S, Vitelli V (2010b) Functional clustering and alignment methods with applications. *Commun Appl Ind Math* 1:205–224. doi:[10.1685/2010CAIM486](https://doi.org/10.1685/2010CAIM486)
- Serban N, Wasserman L (2005) CATS: cluster analysis by transformation and smoothing. *J Am Stat Assoc* 100:990–999. doi:[10.1198/016214504000001574](https://doi.org/10.1198/016214504000001574)
- Sun Y, Genton MG (2011) Functional boxplots. *J Comput Graph Stat* 20:316–334. doi:[10.1198/jcgs.2011.09224](https://doi.org/10.1198/jcgs.2011.09224)
- Tarpey T, Kinatered KJ (2003) Clustering functional data. *J Classif* 20:93–114. doi:[10.1007/s11634-013-0158-y](https://doi.org/10.1007/s11634-013-0158-y)
- Tibshirani R, Walther G, Hastie T (2001) Estimating the number of data clusters via the gap statistic. *J R Stat Soc B* 63:411–423. doi:[10.1111/1467-9868.00293](https://doi.org/10.1111/1467-9868.00293)
- Tokushige S, Yadohisa H, Inada K (2007) Crisp and Fuzzy *k*-means clustering algorithms for multivariate functional data. *Comput Stat* 21:1–16. doi:[10.1007/s00180-006-0013-0](https://doi.org/10.1007/s00180-006-0013-0)
- Tuddenham RD, Snyder MM (1954) Physical growth of California boys and girls from birth to eighteen years. Tech. Rep. 1, University of California Publications in Child Development
- Tukey J (1977) *Exploratory data analysis*. Addison-Westley, Boston. doi:[10.1002/bimj.4710230408](https://doi.org/10.1002/bimj.4710230408)