MOX-Report No. 69/2020

FunCC: a new bi-clustering algorithm for functional data with misalignment

Galvani, M.; Torti, A.; Menafoglio, A.; Vantini S.

MOX, Dipartimento di Matematica
Politecnico di Milano, Via Bonardi 9 - 20133 Milano (Italy)

mox-dmat@polimi.it http://mox.polimi.it

FunCC: a new bi-clustering algorithm for functional data with misalignment

Marta Galvani[1,*] Agostino Torti[2,3,*]
Alessandra Menafoglio[2] Simone Vantini[2]

[1] Department of Mathematics, University of Pavia, Pavia, Italy

[2]MOX - Department of Mathematics, Politecnico di Milano

[3]Center for Analysis Decisions and Society, Human Technopole, Milano

Abstract

The problem of bi-clustering functional data, which has recently been addressed in literature, is considered. A definition of ideal functional bi-cluster is given and a novel bi-clustering method, called Functional Cheng and Church (FunCC), is developed. The introduced algorithm searches for non-overlapping and non-exhaustive bi-clusters in a set of functions which are naturally ordered in matrix structure through a non-parametric deterministic iterative procedure.
Moreover, the possible misalignment of the data, which is a common problem when dealing with functions, is taken into account. Hence, the FunCC algorithm is extended obtaining a model able to jointly bi-cluster and align curves.
Different simulation studies are performed to show the potential of the introduced method and to compare it with state-of-the-art methods. The model is also applied on a real case study allowing to discover the spatio-temporal patterns of a bike-sharing system infrastructure.

Keywords: Bi-clustering, Clustering, Functional Data, Curve alignment

[*] Both authors contributed equally to this work

1 Introduction

Many systems are able to collect information with high frequency, obtaining data streams collected in an almost continuous fashion. For this reason, in the last decades, lot of efforts have been put into the development of new statistical methods able to deal with this new type of data. In particular, functional data analysis (FDA) is the branch of statistics that deals with random variables taking values into an infinite dimensional functional space, see ([1]) and ([2]) for more details.
In this paper we consider the problem of bi-clustering functional data. While clustering methods are able to detect groups observing the similarity between rows or columns (usually observations and features) of a data matrix, a large number of algorithms have been proposed for multivariate data with the aim of performing simultaneous clustering on both dimensions of the data matrix, see ([3]) for a complete review. This is of particular interest when the data are intrinsically ordered in a matrix structure and the aim is to simultaneously group the rows and the columns of the data matrix. Bi-clustering methods allow to discover subgroubs of observations behaving in a similar way on a subset of features or vice-versa a subgroups of features behaving in a similar way only on a subset of observations, without constraining the rows (or the columns) of a data matrix to belong to only one group over all the features (or the observations) as in the classical clustering methods.
In this paper we consider the problem in which each cell of the data matrix is a function and we would like to perform a bi-clustering of these functions to obtain similarity subgroups of rows and columns. To this purpose classical multivariate bi-clustering methods should be extended to deal with functional data. Many different methods have been proposed in the literature for clustering functional data, considering dataset where each observation is a function, see ([4]) for a complete survey on these models. In the bi-clustering framework, ([5]) and ([6]) both proposed a procedure which generalizes the classical latent block model (([7])) for multivariate data to the functional setting. These procedures are model-based and assume the existence of a latent-block structure in the data-matrix. ([5]) and ([6]) assume respectively that the coefficients of the first functional PCA and the basis expansion coefficients of the functions in each block can be adequately described by an m-dimensional Gaussian distribution. These two models are therefore semi-parametric in nature and define as output an exhaustive bi-clustering of the data matrix. In addition, ([5]) considers only the coefficients of the first functional PCA to

represent each curve, therefore loosing information, while (6) allows only for basis expansions as a smoothing procedure, while many other methods can be taken into account considering the different nature of the data at hand.
The aim of this work is to present a new flexible algorithm for the bi-clustering of functional data called *Functional Cheng and Church* (FunCC) algorithm, extending the well known Cheng and Church algorithm, proposed by (8) for the bi-clustering of multivariate data. This novel method is completely non-parametric, thus no assumptions are made on the distribution generating the data. In addition, it gives free choice on the smoothing procedure to be applied on the data at hand. The output of the model is a non-exhaustive bi-clustering of the data matrix, thus, more realistically, assuming the existence of curves possibly not belonging to any bi-cluster.
When dealing with functional data, another problem that has to be taken into account is the possible misalignment of the data (see for example (9) and (10)) which acts as a confounding factor when trying to analyse the data. The problem of curve registration has been considered in literature by different authors. (11) considers self-modelling non-linear regression models to align curves, while (12) develops non-linear mixed effects models. Other works (see (13), (14) and (15)) proceed defining an appropriate similarity index among functions and try to find the best alignment optimizing this similarity measure. Following this latter approach, in the FunCC algorithm we also consider the introduction of warping functions for the curves alignment with the aim of maximizing a goodness measure of the found bi-clusters. Allowing for curves registration, indeed, we are able to bi-cluster functions with a similar behaviour despite of a misalignment of the data.
To show the benefits of the developed methodology, the FunCC algorithm is also applied on a real dataset, the bike sharing system (BSS) of Lyon. The aim is to provide useful information for the correct management of the service by discovering subgroups of stations and days with common operating patterns and highlighting potential issues of the BSS.
The paper is structured as follow: in Section 2 a novel definition of functional bi-cluster is given coupled with a measure of bi-cluster goodness of fit. The extension of the Cheng and Church algorithm for functional data is proposed in Section 3. In Section 4 the more general case which allows for the functions registration step is presented. Different simulation studies are performed in Section 5 to underline the potential of the introduced algorithm and to compare it with state-of-the-art methods. In Section 6 the algorithm is applied on the BSS of Lyon. Conclusion are presented in Section 7. In

appendix A the original Cheng and Church procedure is reported.

2 Functional Bi-clustering

Given a dataset of real numbers arranged in a matrix A composed by n rows and m columns, the aim of a bi-clustering technique is to find a submatrix $B(I, J) \in A$, corresponding to a subset of rows I and a subset of columns J, with a high similarity score (notice that the rows and the columns in the submatrix are not necessarily adjacent).
In the functional framework, suppose to have a sample of $n \cdot m$ continuous functions $f_{ij}(t)$ with $t \in T$ arranged in a matrix A composed by n rows and m columns, i.e. each element of the matrix A is a function $f_{ij}(t)$ with $i \in \{1, ..., n\}$ and $j \in \{1, ..., m\}$. Let $B(I, J)$ be a sub-matrix of A with set of rows $I \subseteq \{1, ..., n\}$ and set of columns $J \subseteq \{1, ..., m\}$ containing only the elements $f_{ij}(t)$ s.t $i \in I$ and $j \in J$.
Thus we can give the definition of a functional bi-cluster as:

Definition 2.1 *An ideal bi-cluster is a sub-matrix $B(I, J)$, s.t each element $f_{ij}(t)$ with $i \in I$ and $j \in J$ can be expressed as:*

$$f_{ij}(t) = \mu^{IJ}(t) + \alpha_i^{IJ}(t) + \beta_j^{IJ}(t) \;\; \forall i \in I \; , \; \forall j \in J \;\; and \;\; t \in T$$

with $\mu^{IJ}(t)$, $\alpha_i^{IJ}(t)$ and $\beta_j^{IJ}(t)$ defined for the bi-cluster $B(I, J)$ as:

- $\mu^{IJ}(t) = \frac{1}{|I||J|} \sum_{i=1}^{|I|} \sum_{j=1}^{|J|} f_{ij}(t)$
- $\alpha_i^{IJ}(t) = \frac{1}{|J|} \sum_{j=1}^{|J|} f_{ij}(t) - \mu(t)$
- $\beta_j^{IJ}(t) = \frac{1}{|I|} \sum_{i=1}^{|I|} f_{ij}(t) - \mu(t)$

Notice that, $\alpha_i^{IJ}(t)$ and $\beta_j^{IJ}(t)$ represent the rows/columns components, i.e. the functional residues of respectively rows and columns with respect to the average function $\mu^{IJ}(t)$ of the bi-cluster. Notice that both $\alpha_i^{IJ}(t)$, $\beta_j^{IJ}(t)$ and $\mu^{IJ}(t)$ are specific for each bi-cluster $B(I, J)$, hence different bi-clusters could have different row and column components. For simplicity of notation in the next sections we drop the apexes I and J from $\mu^{IJ}(t)$, $\alpha_i^{IJ}(t)$ and $\beta^{IJ}(t)$.
Starting from Definition 2.1 it is possible to obtain different kind of bi-clusters associated to different application contexts by setting differently the parameters. Setting, for instance, $\alpha_i = \beta_j = 0$ then the ideal bi-cluster is composed

by a group of functions all equal to the average function $\mu(t)$ in the bi-cluster. Considering non-null components $\alpha_i(t)$ and $\beta_j(t)$, other ideal bi-clusters can be obtained, discovering groups of functions that exhibit coherent variations on the rows or on the columns of the data matrix. Setting $\alpha_i = 0$ and taking only $\beta_j(t)$ into consideration, each bi-cluster is expressed by $\mu(t) + \beta_j(t)$. If forcing the column components to be equal to a constant value along the domain, i.e. $\beta_j(t) = c_j$ with $c_j \in \mathbb{R}$, then each element is represented by the average behaviour of the bi-cluster plus a constant function representing its column specific deviation. If instead $\beta_j(t)$ is not constrained as constant the bi-cluster is a sub-matrix where each column has a similar average behaviour except that for an additive functional components, giving more degrees of freedom to the model. Similar considerations can be done for $\alpha_i(t)$. Indeed, considering only $\alpha_i(t)$ and avoiding the usage of $\beta_j(t)$, each bi-cluster is expressed by $\mu(t) + \alpha_i(t)$ meaning that each element is equal to the average behaviour of the bi-cluster plus a function representing its row specific deviation. When considering all the elements introduced in the Definition 2.1 the found bi-clusters are based on equation $f_{ij}(t) = \mu(t) + \alpha_i(t) + \beta_j(t)$, giving the model different degrees of freedom, hence making more difficult the interpretation of the results.
Consistently with the Cheng and Church approach, we want to find bi-clusters $B(I, J)$ which minimize a specific objective function, hence defining a specific H-score which measures the deviation of the selected rows and columns from an ideal bi-cluster. The introduced H-score evaluates the mean squared residual obtained when representing each function with the estimated template $\mu(t)+\alpha_i(t)+\beta_j(t)$ of the bi-cluster to which the function is assigned to. We then define a new H-score for functional data as:

Definition 2.2 *Let $B(I, J)$ be a bi-cluster and $f_{ij}(t)$ each function belonging to it. The H-score of the functional bi-cluster $B(I, J)$ is defined as:*

$$H(I, J) = \frac{1}{|I||J|} \sum_{i=1}^{|I|} \sum_{j=1}^{|J|} \left\| f_{ij} - \bar{f}_{ij} \right\|_{L_2}^2$$

with

$$\bar{f}_{ij}(t) = \mu(t) + \alpha_i(t) + \beta_j(t) \tag{1}$$

being the template function of the bi-cluster.

The defined H-score is an index of the functions similarity in each bi-cluster. The introduced H-score is based on the L^2 distance between the observed

functional data in the bi-cluster and its template, however, the definition can be readily generalized introducing other metrics among curves to any other functional Hilbert space.
It has to be noticed that every data matrix contains submatrices with the perfect score $H(I, J) = 0$ which corresponds to the degenerative bi-cluster $B(I, J)$ having $|I| = 1$ and/or $|J| = 1$, following the Definition 2.1 of an ideal bi-cluster. Using other definitions of ideal bi-clusters, by setting differently the parameters $\alpha_i(t)$ and $\beta_j(t)$, other limiting cases can be obtained. Indeed, when setting $\alpha_i = \beta_j = 0$ the degenerative bi-clusters with $H(I, J) = 0$ are all the submatrices of dimension one. If instead we consider the row components $\alpha_i(t)$ and we set $\beta_j = 0$ the degenerative bi-clusters are all the submatrices with $|I| \geq 1$ and $|J| = 1$, as opposite considering $\beta_j(t)$ and setting $\alpha_i = 0$ the degenerative bi-clusters are all the submatrices with $|I| = 1$ and $|J| \geq 1$. As these bi-clusters are limiting cases in which the H-score is zero by definition, in the algorithm introduced in the next section we impose some constrains on the bi-clusters dimensions according to the chosen parameters setting. Specifically, if $\alpha_i = \beta_j = 0$ each bi-cluster $B(I, J)$ should have $|I| > 1$ and $|J| > 1$, if we consider the column component $\beta_j(t)$ and $\alpha_i = 0$ then $B(I, J)$ should have $|I| > 1$ and $|J| \geq 1$, if we consider the row component $\alpha_i(t)$ and $\beta_j = 0$ then $B(I, J)$ should have $|I| \geq 1$ and $|J| > 1$.

3 Functional Cheng and Church algorithm

As proven in (8), the problem of finding a bi-cluster is NP-hard, therefore a greedy procedure is employed to find an approximate solution. Hence, to find the set of bi-clusters a deterministic and greedy algorithm is defined as in Algorithm 1, following the main structure of the Cheng and Church procedure (see the A for original Cheng and Church algorithm details).
The algorithm starts considering the whole dataset and proceeds iteratively removing and adding elements to find the biggest bi-cluster with an H-score lower then a given threshold δ. The Multiple Node Deletion phase allows for a faster but rougher procedure trying to remove at the same time groups of rows or columns with scores bigger than the H-score scaled by a parameter $\theta \geq 1$. The lower is θ the faster is the algorithm and more rows or columns are removed at the same time. After this phase, a Single Node Deletion phase is performed until the H-score is lower than a threshold δ; at each iteration

the row or column with the biggest score is removed and the H-score of the new obtained matrix is updated. In the functional case the rows/columns scores are estimated extending the rows/columns scores introduced by (8). Specifically, we evaluate respectively the row and the column scores of a submatrix $B(I, J)$ as:

$$d_{iJ} = \frac{1}{|J|} \sum_{j=1}^{|J|} \left\| f_{ij} - \bar{f}_{ij} \right\|_{L_2}^2 \quad \forall i \in I$$

$$d_{jI} = \frac{1}{|I|} \sum_{i=1}^{|I|} \left\| f_{ij} - \bar{f}_{ij} \right\|_{L_2}^2 \quad \forall j \in J$$

with $\bar{f}_{ij}(t)$ as defined in (1). At the end of this phase the algorithm tries to add removed rows or columns in order to make the bi-cluster as big as possible without increasing the H-score. The Multiple and Single Node Deletion steps follow the same procedure as in the original Cheng and Church algorithm (See A for more details). In the original Node Addition Steps by Cheng and Church also inverted rows are considered (see A), this because the algorithm was developed for gene expression data and an anti-correlated or inverted row may indeed represent a negatively regulated genes that is of interest when finding a bi-cluster. In our case, we do not consider the inverted rows as they are not of interest when defining a bi-cluster in a general framework.

After finding a new bi-cluster, in the original Cheng and Church algorithm, before searching for a new one, a masking procedure is performed on the assigned elements substituting them with numbers from a random distribution (see A). This procedure reduces the probability of those elements to be assigned to another bi-clusters in the following iterations, but it does not ensure that, at the end of the algorithm, each element belongs to at most one bi-cluster. For this reason, in our algorithm, we decide to avoid this masking procedure. Therefore, as replacement, after a new bi-cluster is found, our algorithm proceeds by looking for new bi-clusters avoiding to consider the already assigned elements. An example of the used procedure is illustrated in Figure 1. In details, after finding a new bi-cluster, the algorithm looks for all the biggest submatrices contained in a binary matrix A' where each element $a'_{ij} = 1$ if the function $f_{ij}(t)$ has not been assigned yet to any bi-cluster and 0 otherwise. To do that, we employ the Bimax Bi-clustering, based on

the framework by (16), which searches for the biggest submatrices of ones in a logical matrix. So, at each time a new bi-cluster is found, the set of biggest submatrices on which a new bi-cluster can be found is updated and the algorithm proceeds looking for a new bi-cluster in the biggest of these submatrices. The procedure stops when the number of iterations exceeds a fixed $maxIter$ value or when no more bi-clusters are found in no one of the submatrices of not assigned elements. Note that, in Figure 1, bi-clusters are represented as blocks of adjacent rows and columns just for ease of explanation. Indeed, the algorithm looks for bi-clusters that can reconstruct a sub-matrix by means of a permutations of rows and columns.

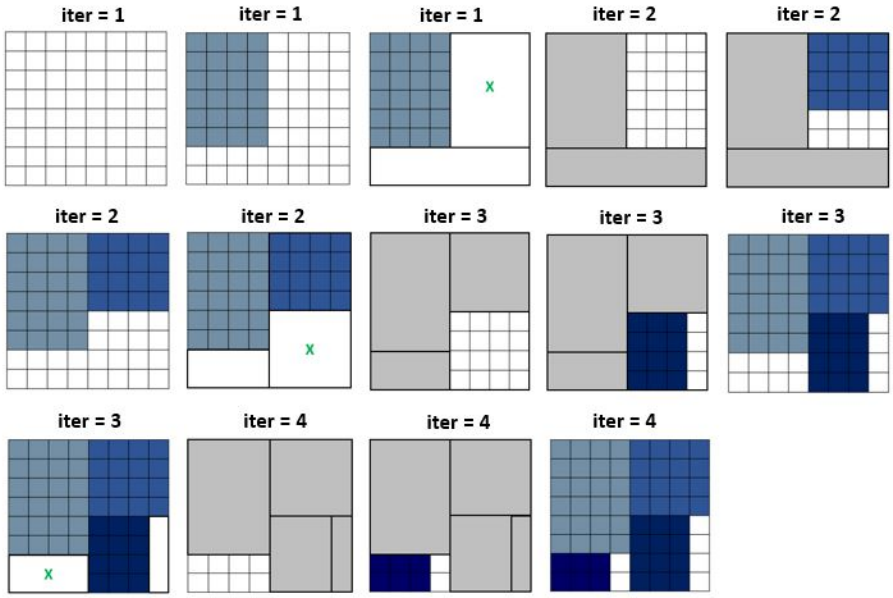

Figure 1: Illustration of the iterative procedure of the Functional Cheng and Church algorithm.

Algorithm 1: Functional Cheng and Church algorithm

Input: (n, m) matrix A whose elements are functions $f_{ij}(t)$
$\delta \geq 0$ the maximum acceptable H-score
$\theta \geq 1$ a parameter for Multiple Node Deletion
$maxIter$ the maximum number of allowed iterations

Result: A set $\mathbb{B}$ of Bi-clusters $B(I, J)$ with H-score$< \delta$

Set:

- $\mathbb{B} = \{\}$ set of bi-clusters found
- $(I^{iter}, J^{iter}) = \{(i, j) : i \in 1...n, j \in 1...m\}$ set of not assigned elements
- $\mathbb{M} = \{A\}$ set of biggest submatrices and M=A

while *iter* $< maxIter$ *and* $\mathbb{M} <> \{\}$ **do**

On M do:

1. **Multiple Node Deletion**: remove a group of rows/columns with score greater than $\theta \cdot H$-score.
2. **Single Node Deletion**: remove the rows/columns that reduce H-score the most while H-score$> \delta$.
3. **Node Addition**: add rows/columns that do not increase the H-score.

if *A new bi-cluster* $B(I, J)$ *is found* **then**

- $\mathbb{B} = \mathbb{B} \cup B(I, J)$
- $(I^{iter+1}, J^{iter+1}) = (I^{iter}, J^{iter})/(I, J)$
- update $\mathbb{M}$ applying Bmax algorithm on A' with $A'(I^{iter+1}, J^{iter+1}) = 1$ and 0 elsewhere and select $M \in \mathbb{M}$ the biggest submatrix

else

select the following biggest $M \in \mathbb{M}$

end

end

Parameters Selection

As in the classical Cheng and Church algorithm, there are two important parameters, δ and θ, that need to be set before the algorithm running. The parameter δ influences the number of the obtained bi-clusters and generally a small value of δ is better because it defines the quality of the bi-clusters. However, a too low value would imply a really large number of bi-clusters or even the impossibility to find a bi-cluster with score smaller than δ. By contrast, a too high value of δ would imply a unique big bi-cluster that corresponds to the whole data matrix. Hence, a balance value should be found for this parameter before running the algorithm. To tune this δ parameter, we perform a sensitivity analysis on the number of obtained bi-clusters and the number of not assigned observations, observing how these two values change when varying the parameter δ. Then, following the same approach used for many other clustering techniques as the classical k-means, we choose the value of δ where an evident change of slope (i.e. an elbow) in the observed values is present.
The second important parameter that has to be set is θ, which is used in the multiple node deletion phase of the algorithm and directly influences the algorithm speed. A too high value of θ would make impossible to pass through the multiple node deletion step forcing the algorithm to apply the slower single node deletion step, while with a too low value of the parameter a high number of rows and columns would be removed, thus following a too raw procedure. Therefore θ is selected as big as possible, while still maintaining low the computational time. To tune this parameter we make a sensitivity analysis on the computational time requested to run the algorithm, taking δ as fixed. Notice that, following (8), θ is taken greater or equal than unity to guarantee the decreasing of the H-score along iterations.
An example of this parameters selection procedure is explained in Section 6.

4 Functional bi-clustering with alignment

As pointed out in Section 1, a problem often encountered in functional data analysis is the misalignment of the curves (or registration problem). Indeed, the misalignment may act as a confounding factor when analysing the data. In the case of bi-clustering, allowing for curves registration, we are grouping functions with a similar behaviour with respect to the bi-cluster template de-

spite of a misalignment along the domain. To this purpose, we have to handle with the problem of aligning each function to its template when defining a bi-cluster.
In a general framework, given a set of functions $\mathbb{F}$, aligning a function $\mathbf{f}$ to another function $\mathbf{g}$ (that in our case is the template function) means finding a warping function $w(t) : \mathbb{R} \to \mathbb{R}$ of the abscissa parameter t such that $\mathbf{f} \circ w$ and $\mathbf{g}$ are less dissimilar than $\mathbf{f}$ and $\mathbf{g}$ themselves, according to a defined dissimilarity score $\epsilon(.,.) : \mathbb{F} \times \mathbb{F} \to \mathbb{R}$. More precisely, given a dissimilarity measure ϵ between curves and a class $\mathbb{W}$ of warping functions such that $\mathbf{f} \circ w \in \mathbb{F}, \forall \mathbf{f} \in \mathbb{F}$ and $\forall w \in \mathbb{W}$, to align $\mathbf{f}$ and $\mathbf{g}$ one needs to find the w^* which minimizes $\epsilon(\mathbf{f} \circ w, \mathbf{g})$. The couple $(\epsilon, \mathbb{W})$ should satisfy some minimal requirements ((14),(17)):

1. The dissimilarity index ϵ has a lower bound in 0 and respects the classical properties of distance measures, i.e. it is *reflexive, symmetric, transitive*;

2. The class of warping functions $\mathbb{W}$ is a convex vector space and has a group structure with respect to function composition $\circ$;

3. The couple $(\epsilon, \mathbb{W})$ is consistent in the sense that given two functions, $\mathbf{f}$ and $\mathbf{g}$, we have that $\forall w \in \mathbb{W}$:

$$\epsilon(\mathbf{f} \circ w, \mathbf{g} \circ w) = \epsilon(\mathbf{f}, \mathbf{g}).$$

From 2 and 3 we obtain that for all w_1 and $w_2 \in \mathbb{W}$:

$$\epsilon(\mathbf{f} \circ w, \mathbf{g} \circ w) = \epsilon(\mathbf{f} \circ w \circ w^{-1}, \mathbf{g}) = \epsilon(\mathbf{f}, \mathbf{g} \circ w \circ w^{-1}).$$

Property 3 highlights the importance of a careful and consistent choice of the couple $(\epsilon, \mathbb{W})$, as these requirements concern ϵ and $\mathbb{W}$) jointly (e.g. (18) used the nonparametric form of the Fisher-Rao metric for this purpose).
In our case, the considered dissimilarity index is the squared L^2 distance between two functions, evaluated in the H-score as defined in Definition 2.2. Therefore we consider the class of warping functions $\mathbb{W}$ as:

$$\mathbb{W} = \{w : w(t) = t + q \quad with \quad q \in \mathbb{R}\},$$

i.e. the group of shift transformations, which were shown to fulfill properties 1-3 in (14).
Thus we can give the definition of functional bi-cluster with alignment as:

Definition 4.1 *An ideal bi-cluster is a sub-matrix $B(I, J)$, s.t each element $f_{ij}(t)$ with $i \in I$ and $j \in J$ can be expressed as:*

$$(f_{ij} \circ w_{ij}^{IJ})(t) = \mu^{IJ}(t) + \alpha_i^{IJ}(t) + \beta_j^{IJ}(t)$$

with $\mu^{IJ}(t)$, $\alpha_i^{IJ}(t)$ and $\beta_j^{IJ}(t)$ defined for the bi-cluster $B(I, J)$ as:

- $\mu^{IJ}(t) = \frac{1}{|I||J|} \sum_{i=1}^{|I|} \sum_{j=1}^{|J|} (f_{ij} \circ w_{ij}^{IJ})(t)$
- $\alpha_i^{IJ}(t) = \frac{1}{|J|} \sum_{j=1}^{|J|} (f_{ij} \circ w_{ij}^{IJ})(t) - \mu(t)$
- $\beta_j^{IJ}(t) = \frac{1}{|I|} \sum_{i=1}^{|I|} (f_{ij} \circ w_{ij}^{IJ})(t) - \mu(t)$

and $w_{ij}^{IJ}(t)$ being a warping function in $\mathbb{W}$ the class of shift transformations.

For simplicity of notation in the remaining part of the paper we drop I and J as apexes in $\mu^{IJ}(t)$, $\alpha_i^{IJ}(t)$, $\beta^{IJ}(t)$ and $_{ij}^{IJ}(t)$.
The introduced warping function allows for an additional degree of freedom in the definition of an ideal bi-cluster, by defining a shift on the domain specific for each function. Setting differently the parameters $\alpha_i(t)$ and $\beta_j(t)$ and considering or not the warping function $w_{ij}(t)$, many different types of ideal bi-cluster can be obtained. As already underlined in Section 2, how to set the parameters is application specific and depends on the problem at hand.
The new H-score for a functional bi-cluster $B(I, J)$ with alignment is hence defined as:

Definition 4.2 *Let $B(I, J)$ be a bi-cluster and $f_{ij}(t)$ each function belonging to it. The H-score of the functional bi-cluster $B(I, J)$ is defined as:*

$$H_{IJ} = min_{\{w_{ij}, i \in I, j \in J\} \subset W} \frac{1}{|I||J|} \sum_{i=1}^{n} \sum_{j=1}^{m} \left\| f_{ij} \circ w_{ij} - \bar{f}_{ij} \right\|_{L_2}^2$$

where

$$\bar{f}_{ij}(t) = \mu(t) + \alpha_i(t) + \beta_j(t) \tag{2}$$

with $\mu(t)$, $\alpha_i(t)$ and $\beta_j(t)$ as defined in Defintion 4.1 and $w_{ij}(t)$ being a warping function in $\mathbb{W}$ the class of shift transformations.

An alignment procedure is also introduced when evaluating the row and the column scores of a submatrix $B(I, J)$ as:

$$d_{iJ} = min_{\{w_{ij}, i\in I, j\in J\}\subset W} \frac{1}{|J|} \sum_{j=1}^{|J|} \left\| f_{ij} \circ w_{ij} - \bar{f}_{ij} \right\|_{L_2}^2 \quad \forall i \in I$$

$$d_{Ij} = min_{\{w_{ij}, i\in I, j\in J\}\subset W} \frac{1}{|I|} \sum_{i=1}^{|I|} \left\| f_{ij} \circ w_{ij} - \bar{f}_{ij} \right\|_{L_2}^2 \quad \forall j \in J$$

where $\bar{f}_{ij}(t)$ as in (2) and $w_{ij}(t)$ being a warping function in $\mathbb{W}$ the class of shift transformations.
In this case, the algorithm follows the same procedure as in Algorithm 1 except for, before evaluating the H-score of a bi-cluster $B(I, J)$ and the rows and columns scores, an alignment step is introduced. In details, in order to find the warping functions $w_{ij}(t)$, specific for each function $f_{ij}(t) \in B(I, J)$, a two steps iterative procedure is implemented as follow:

- Alignment of the functions: each function $f_{ij}(t)$ inside the sub-matrix $B(I, J)$ is aligned to the template function $\bar{f}_{ij}(t)$ determining a warping function
$$w_{ij}^* = argmin_{w_{ij}\in W} \left\| f_{ij} \circ w_{ij} - \bar{f}_{ij} \right\|_{L_2}^2 \tag{3}$$
with $\mathbb{W}$ being the class of shift transformations;

- Identification of the new template: the new template function $\bar{f}_{ij}(t)$ of the sub-matrix $B(I, J)$ is estimated using the aligned functions $(f_{ij} \circ w_{ij}^*)(t)$ as in (2).

These two steps are iterated until convergence, e.g. no more improvement in minimizing the distance between aligned functions in the bi-cluster and the template function are achieved. The aligned functions and the new template are then used to estimate the H-score, as in Definition 4.2, and the rows and columns scores. The iterative alignment procedure is shown in Algorithm 2. The alignment procedure is introduced in Algorithm 1 in the Multiple and Single node deletion step and in the Node Addition, i.e. each time the H-score or the rows and column scores of a bi-cluster $B(I, J)$ are evaluated.

Algorithm 2: Alignment procedure

Input: A sub-matrix $B(I, J)$ where each element is a function f_{ij}

Result: Warping functions $w^*_{ij}(t) \in \mathbb{W}$ for each function $f_{ij}(t)$ with $i \in I$ and $j \in J$

while *No more improvements in minimizing the distance between aligned functions and the template function are achieved* **do**

1. **Alignment of the functions**:
   for each function $f_{ij}(t)$ in the sub-matrix $B(I, J)$ the warping function $w^*_{ij}(t)$, which minimizes the distance of the function $f_{ij}(t)$ to the template function $\bar{f}_{ij}$, is determined;

2. **Identification of the new template**:
   evaluate the new template function $\bar{f}_{ij}(t)$ of the sub-matrix $B(I, J)$ using the aligned functions $(f_{ij} \circ w^*_{ij})(t)$.

end

5 Simulation study

In this section we illustrate the potential of the FunCC algorithm, described in the previous sections, through different simulation studies. In details, in case A, we simulate a non-exhaustive bi-cluster structure showing the potential of the algorithm in determining only the true bi-clusters and leaving all the other elements as not assigned. In case B, we show the importance of considering the rows (and/or the columns) components. In case C, we show the performance of our algorithm in the case of misaligned functions. The FunCC algorithm is then compared with the state-of-the-art methods for functional bi-clustering in case D.

5.1 Case A: non-exhaustive bi-cluster

We simulate a data matrix A of dimensions 30 x 7 with two bi-clusters. Each bi-cluster is defined considering different prototype curves: $g_1(t) = [t^4 - t^3 - 19t^2 - 11t + 81]/10$, $g_2(t) = [-(t^4 - t^3 - 19t^2 - 11t) - 100]/10$ with

$t \in [0, 5]$. The data matrix is generated as follows:

$$g_{ij}(t) = \begin{cases} g_1(t) + \varepsilon_{ij}(t) & \forall \ (i,j) \in [1:15, 1:4] \\ g_2(t) + \varepsilon_{ij}(t) & \forall \ (i,j) \in [17:30, 5:7] \end{cases}$$

where the errors $\varepsilon_{ij}(t)$ are from a Gaussian process with zero mean and $E(\varepsilon_{ij}(t)\varepsilon_{ij}(s)) = e^{-(|t-s|)}$. All other elements in A are i.i.d. noisy data such that $g_{ij}(t) = 5\varepsilon_{ij}(t)$. The simulated curves are displayed in Figure 2.

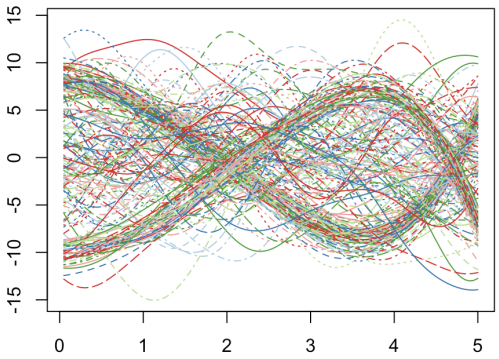

Figure 2: Simulated curves (Case A)

The FunCC algorithm, with parameters $\delta = 2$, $\theta = 1$, $\alpha_i = 0 \ \ i = 1, ..., n$ and $\beta_j = 0 \ \ j = 1, ..., m$, can easily reconstruct the bi-clustering structure: it finds two bi-clusters, which come from templates $g_1(t)$ and $g_2(t)$ respectively, leaving all the other elements as not included in any bi-cluster. Results are shown in Figure 3, where bi-cluster 0 represents the artificial bi-cluster containing the not assigned elements.

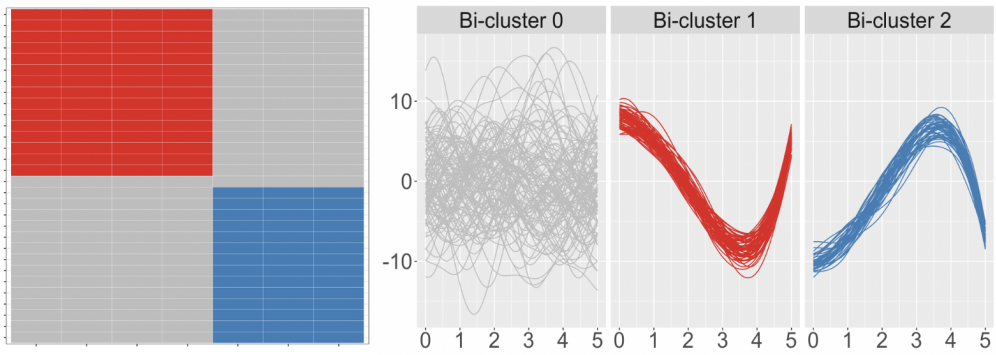

Figure 3: Obtained results applying FunCC algorithm to the simulated data (Case A): resulting matrix (left) and assigned functions to each bi-cluster (right). The not assigned elements are artificially assigned to bi-cluster 0.

5.2 Case B: the rows components

We simulate a data matrix A of dimensions 30 x 7 with three bi-clusters. Each bi-cluster is defined considering as prototype curves: $g_1(t) = t^4 - 9t^2$, $g_2(t) = -(t^4 - 9t^2) + 5$ and $g_3(t) = 0$ with $t \in [-3.5, 3.5]$.
We define each entry $g_{ij}(t)$ of the data matrix A as the introduced prototype curves with an additional small error $\varepsilon_{ij}(t)$. Random row components are also added as $\alpha_i(t) = c_i \sim U[0,1]$ $i = 1, ..., n, t \in [0, 5]$.
Specifically the data matrix is generated as follows:

$$g_{ij}(t) = \begin{cases} g_1(t) + \alpha_i(t) + \varepsilon_{ij}(t) & \forall\ (i,j) \in [1:14, 1:5] \\ g_2(t) + \alpha_i(t) + \varepsilon_{ij}(t) & \forall\ (i,j) \in [15:30, 1:5] \\ g_3(t) + \alpha_i(t) + \varepsilon_{ij}(t) & \forall\ (i,j) \in [1:30, 6:7] \end{cases}$$

where the errors $\varepsilon_{ij}(t)$ are from a Gaussian process with zero mean and $E(\varepsilon_{ij}(t)\varepsilon_{ij}(s)) = e^{-(|t-s|)}$. The simulated curves are displayed in Figure 4.

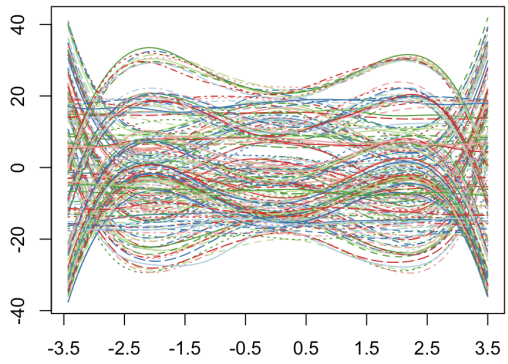

Figure 4: Simulated curves with row components (Case B)

The FunCC algorithm, with parameters $\delta = 10$, $\theta = 1$, $\alpha_i(t) = c \in \mathbb{R} \;\; i = 1, ..., n, t \in [0, 5]$ (i.e. constrained as constant) and $\beta_j = 0 \quad j = 1, ..., m, t \in [0, 5]$, finds three bi-clusters whose results are reported in Figure 5.

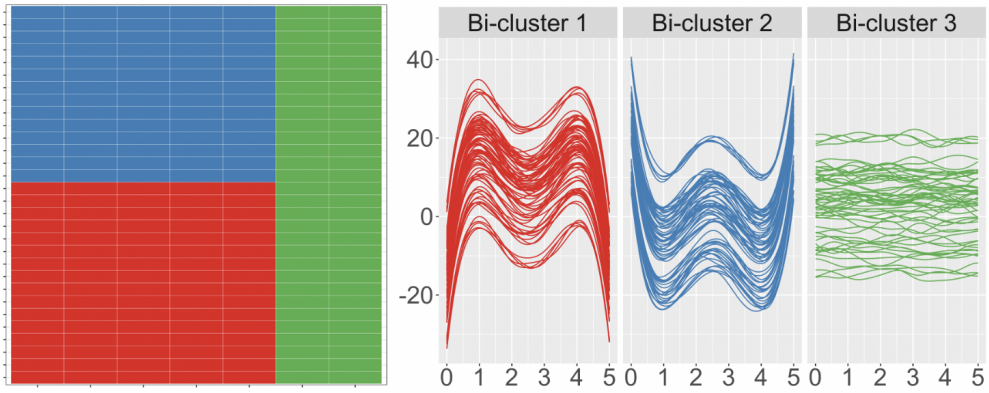

Figure 5: Obtained results applying FunCC algorithm to the simulated data (Case B): resulting matrix (left), assigned functions to each bi-cluster (right)

It is immediate to observe that the algorithm is able to perfectly reconstruct the generated data structure assigning each element to the corresponding bi-cluster. If not considering any row components when searching for the bi-clusters, the algorithm is not able to detect the three bi-clusters and identifies an higher number of bi-clusters just taking into account the average

functional behaviour in each bi-cluster. If instead $\alpha_i(t)$ is not constrained to be a constant component, then bi-cluster 1 and bi-cluster 2 are joined together, since each element can be described by the average function plus a column specific functional component.

5.3 Case C: the shift alignment

Consider the prototype curve $c(t) = 2sin(2t)$ with $t \in [0, 2\pi]$. We simulate a data matrix A of dimensions 15 x 20 whose entries are functions coming from the introduced prototype curve with a random translation along the domain from a uniform distribution between 0 and 2π, plus an additional small error. Specifically,

$$g_{ij}(t) = 2sin(2t + u) + \varepsilon_{ij}(t) \;\; u \sim U[0, 2\pi] \;\; i = 1, ..., n \;\; j = 1, ..., m,$$

where the errors $\varepsilon_{ij}(t)$ are Gaussian process with zero mean and $E(\varepsilon_{ij}(t)\varepsilon_{ij}(s)) = 0.1e^{-(|t-s|)}$. The 300 simulated curves are displayed in Figure 6.

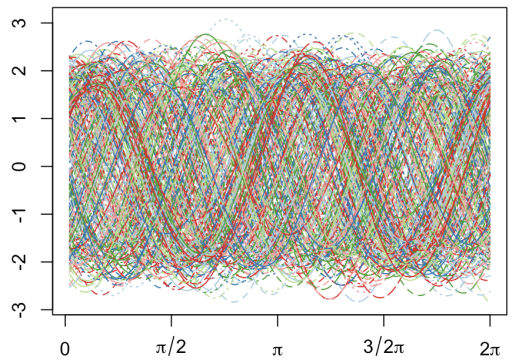

Figure 6: Simulated curves with shift alignment (Case C)

The FunCC algorithm, with parameters $\delta = 0.1$, $\theta = 1$, $\alpha_i(t) = 0 \;\; i = 1, ..., n, t \in [0, 2\pi]$, $\beta_j(t) = 0 \;\; j = 1, ..., m, t \in [0, 2\pi]$, finds a unique bi-cluster that covers the entire data matrix whose results are reported in Figure 7.

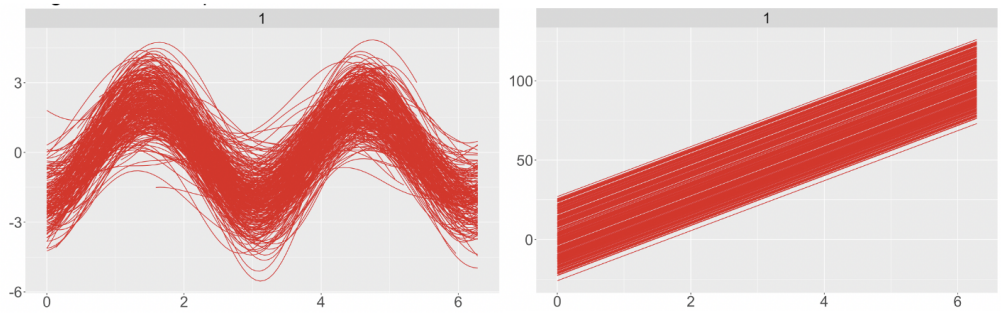

Figure 7: Obtained results applying FunCC algorithm to the simulated data (Case C): aligned functions in the bi-cluster (left) and warping functions for each function (right).

The alignment step is performed allowing for a maximum shift of 30% of the domain. If not considering the alignment phase the algorithm does not find a single bi-cluster, i.e. it does not recognize that all the functions come from the same distribution and try to group them in different bi-clusters.

5.4 Case D: Comparison with state-of-the-art method

This last numerical study aims to compare the FunCC algorithm with ther state-of-the-art method in the functional bi-clustering literature, namely with the FunLBM algorithm by ([6]). The two methods, even if they have the same purpose of finding common groups of rows and columns, have many differences by construction. First of all, the FunCC method, as natural extension of the Cheng and Church model ([8])), looks for non exhaustive bi-clusters, while the FunLBM, as natural extension of the Latent Block Model (LBM, ([7])), looks for exhaustive bi-clusters able to reconstruct a checkboard structure by means of a permutation of rows and columns. Regarding the model themselves, the FunCC algorithm is deterministic and non-parametric while the FunLBM algorithm is semi-parametric since it assumes a Gaussian mixture distribution on the basis coefficients of the functional data. In this regard, being non-parametric, the FunCC procedure is expected to perform better than the FunLBM procedure in the cases in which the bi-clusters are not gaussian. Moreover, while the FunLBM algorithm works on the basis expansion of the functional data performing a basis decomposition inside the algorithm, the FunCC algorithm allows for different smoothing choices since

it does not rely on any specific smoothing procedure because it works on the functions themselves rather than on the coefficients of a basis expansion. Furthermore we have to underline that the FunLBM model does not deal with row or column components, neither with curves registration problems that may be crucial when identifying bi-clusters. Thus, to directly compare the two models, we simulate a data matrix with a checkboard structure, without assuming any row or column components in the data, neither introducing a misalignment. This is a case that can be properly managed also by the LBM ([6]).
The simulated data matrix A has dimension 30 x 7 with nine bi-clusters, whose means are defined by nine different functions $f_1(t), ..., f_9(t)$ with $t \in [0, 1]$, as shown in Figure 8.

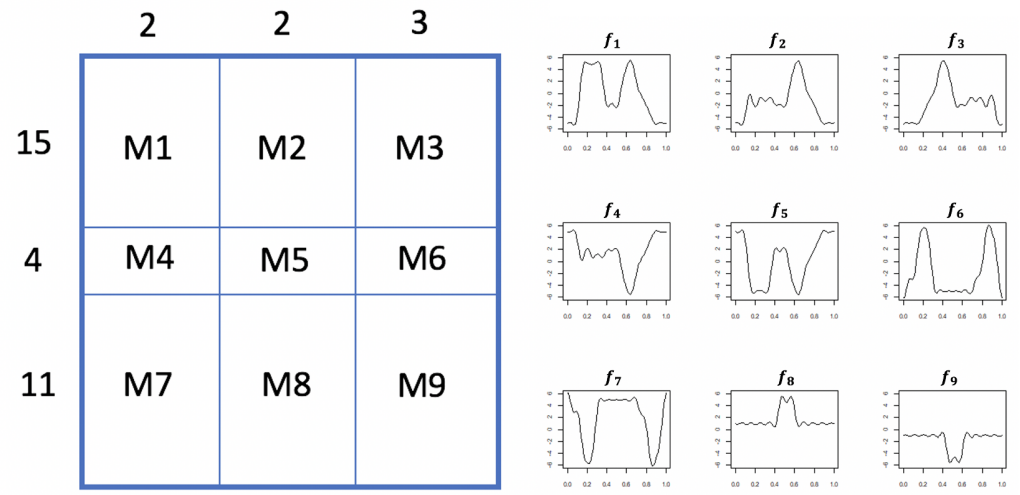

Figure 8: The simulated data matrix structure (left) and the nine functional means used in the simulations (right).

All curves are sampled as follows:

$$g_{ijk}(t) = N(f_k(t), 0.1^2) \;\; i = 1, ..., 30 \;\; j = 1, ..., 7 \;\; t \in [0, 1]$$

In Figure 9 obtained results for both FunLBM and FunCC algorithm are reported.

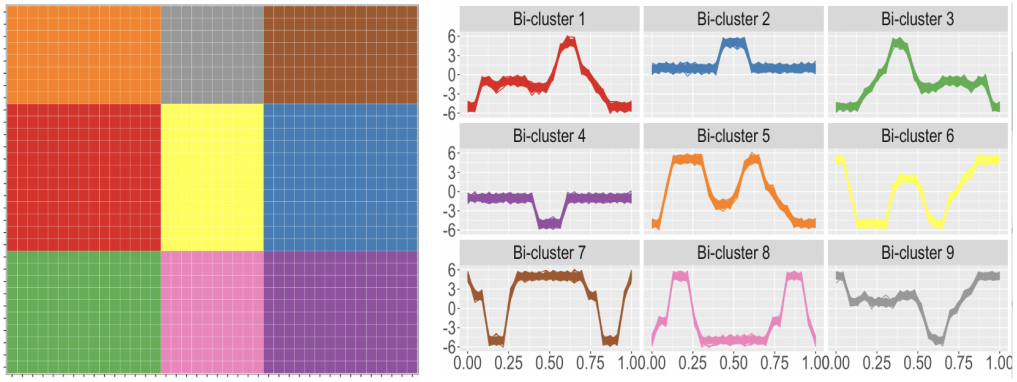

Figure 9: Obtained results applying both FunLBM and FunCC algorithm to the simulated data (Case D). Resulting biclusters (left) and assigned functions to each bi-cluster (right)

In details, FunLBM algorithm is run with parameters $L = 3$, $K = 3$ (representing respectively the number of rows and columns clusters) and Fourier basis expansion with 18 basis; FunCC algorithm is run with parameters $\delta = 0.6$, $\theta = 1$, $\alpha_i(t) = 0, \quad i = 1, ..., 30 \ t \in [0, 1]$, $\beta_j(t) = 0, \quad j = 1, ..., 7 \ t \in [0, 1]$. The same smoothing technique as in the FunLBM procedure is applied. It is possible to observe that both methods are able to detect the different subgroups of the simulated data matrix.

6 Case Study

The new algorithm presented in Section 2 has been applied on a real case study to underline its potential on a real dataset. We focus on the Bike Sharing System (BSS) of Lyon, called Vélo'v, with the aim of providing useful information for the correct management of the service by highlighting specific spatio-temporal patterns in the bike stations usage profiles.
The analysed dataset contains the loading profiles of the 345 bike stations in Lyon over one week in March 2014 and it is available at `https://developer.jcdecaux.com/` trough an api key. This dataset has been first used in (19) that aimed at identifying common operating patterns and highlight potential issues of the BSS. In this work the authors treated the data as functional, due to their continuous dependence on time, and performed a cluster analysis on the leading profiles of each station looking at them as a single curve along the

entire week. By doing so, they discovered stations with a common behaviour during the whole week, but at the expense of the differences between days. It may indeed happen that some stations have a similar behaviour during some specific days but a different one in other days, as usually a different behaviour is observed among working days and during Saturday or Sunday. To underline these patterns, a bi-clustering approach is necessary, since it allows to look at the same time at two dimensions, i.e. the stations and the days, identifying subgroups of stations with the same behaviour in a subgroup of days. Following this idea, in our analysis we define a function for each station and for each day, arranging the functional data in a matrix whose rows are stations and columns are days. We define a bike station loading profile during an entire day as a continuous functional datum representing the number of available bikes divided by the total number of bike docks at each timestamp. In details, a kernel density estimation smoothing procedure is applied on the functions, with a tricube kernel function, a bandwidth equal to 0.5 and a numerical estimation grid of 240 points (see (20) for more details on smoothing procedures). The final data matrix is composed by 2415 curves $f_{ij}(t)$, i.e. 345 stations (i.e. rows) per 7 days (i.e. columns), and the resulting functions are shown in Figure 10.

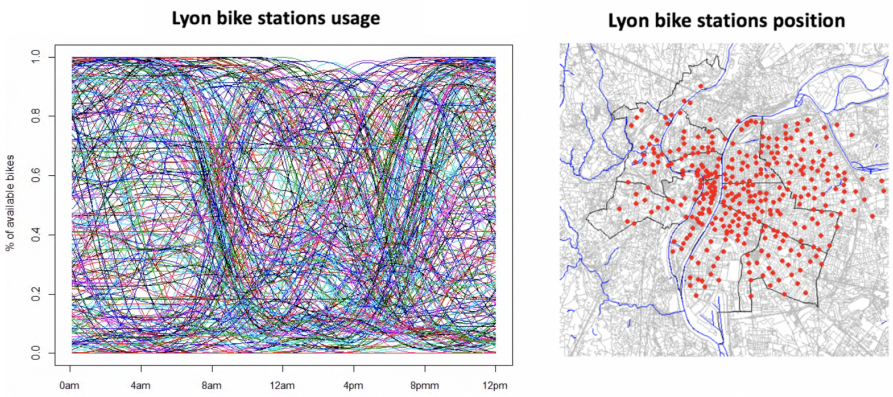

Figure 10: Left: a sample of 300 curves randomly extracted from the functional data matrix. Right: the position of the 345 bike stations in Lyon.

The FunCC algorithm, presented is Section 2, is then applied on this dataset with the aim of finding sub-groups of bike stations and days with

a similar behaviour. To this purpose, no alignment is considered and both $\alpha_i(t)$ and $\beta_j(t)$ are set equal to zero for all i and j. In this way, each bi-cluster is represented only by its average behaviour and the ideal bi-cluster is characterised by a group of functions all equal to each other. A sensitivity analysis is performed to set the two hyper-parameters δ and θ, as explained is Section 2. First, a sensitivity analysis is performed to choose the threshold parameter δ maintaining θ fixed at high value as to not perform the multiple node deletion. In Figures 11 the different number of obtained bi-clusters and the number of observations not assigned to any bi-cluster obtained varying the parameter are reported.

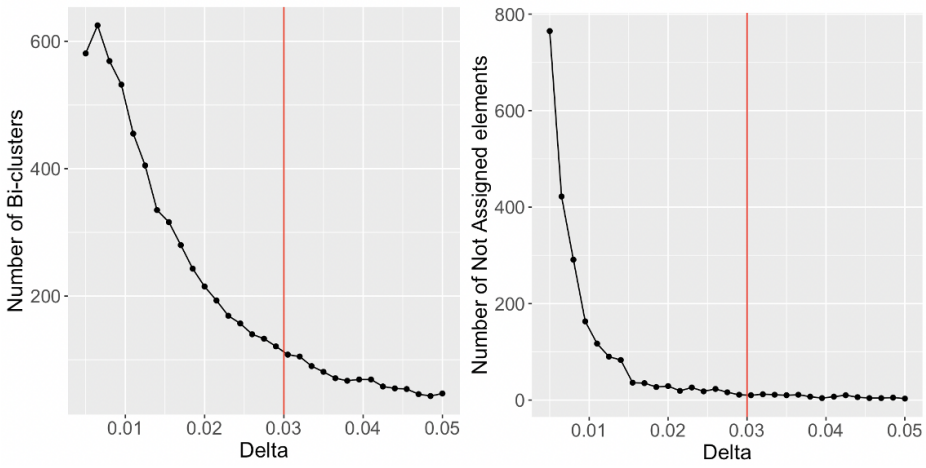

Figure 11: The different number of obtained bi-clusters (left) and the number of observations not assigned to any bi-cluster (right) varying the parameter δ.

Looking at the trend of the number of not assigned elements we notice that for a δ bigger than 0.015 the curve seems not to decrease (an elbow is evident). In addition, observing the number of obtained bi-clusters, we notice that with δ around 0.03 the descent is gentle and after this value the trend does not change essentially. A δ equal to 0.03 is chosen as threshold for the H value of each bi-cluster. After setting parameter δ, the computational requested time to run the algorithm is evaluated when varying θ in the interval $[1, 3]$ (Figure 12).

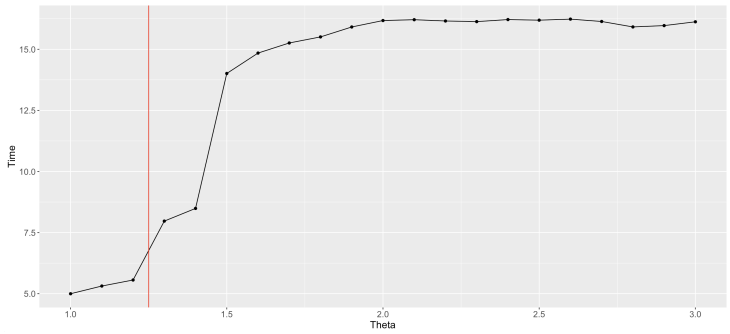

Figure 12: Computational requested time (in minutes) to run the algorithm varying θ.

It can be noticed that, with values bigger than 1.8, the computation time converges. Recall that, when choosing θ, the aim is to maintain low the computational time while being precise in finding a bi-cluster, i.e. removing only one row/column at time. For this reason we decide to proceed with $\theta = 1.25$ setting a computation time as short as possible, while maximizing θ to be more precise when removing elements from bi-clusters.
Results obtained with $\delta = 0.03$ and $\theta = 1.25$ are shown in Figure 13 and Figure 14.

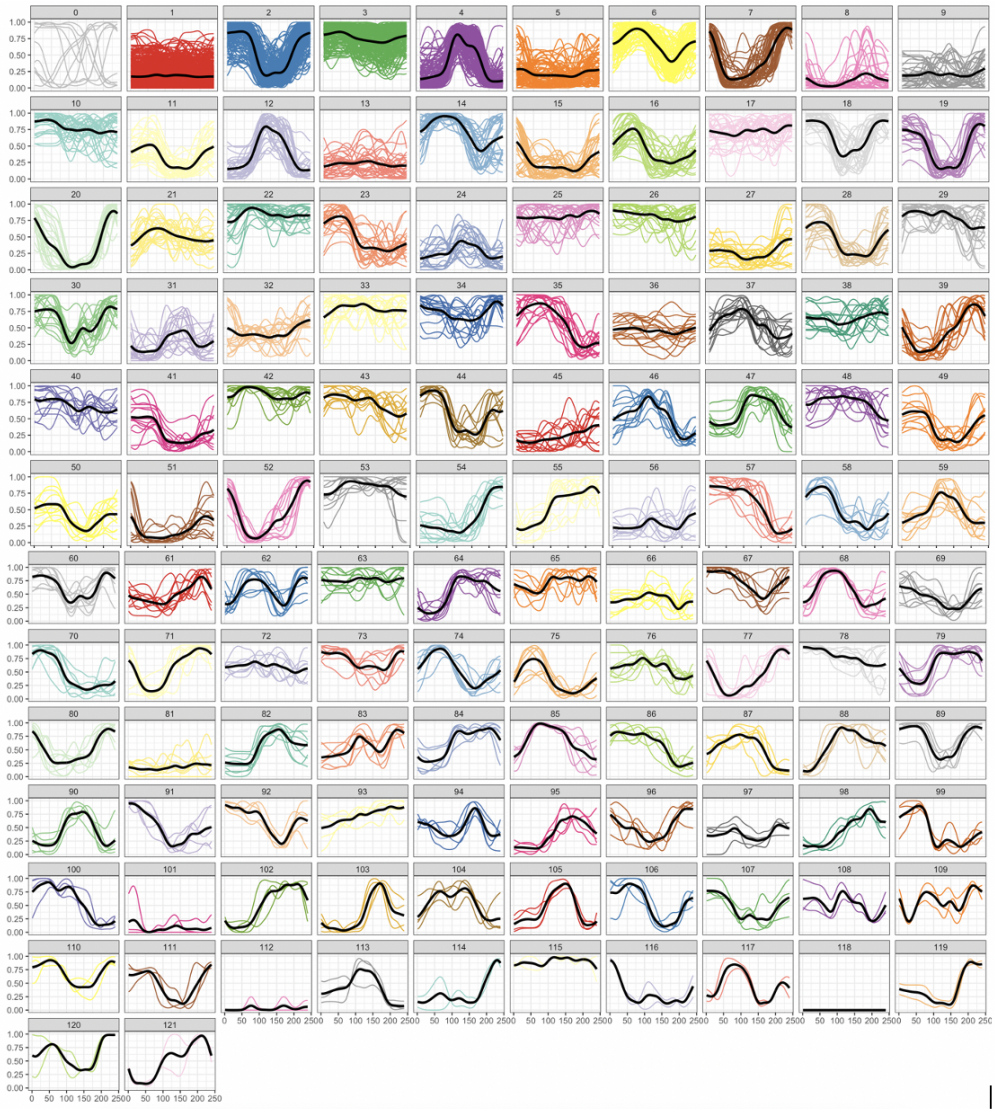

Figure 13: Functions belonging to each bi-cluster with template functions in black

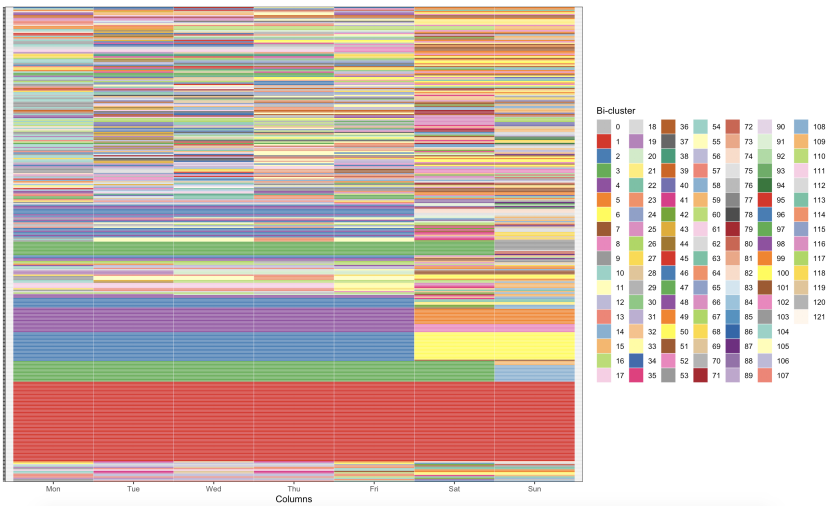

Figure 14: Data matrix showing the membership of each element to the respective bi-cluster.

A total of 121 bi-clusters is found, while observations not assigned to any bi-clusters are artificially assigned to bi-cluster 0, which is coloured in grey. Among the found bi-clusters, 54 have both the number of covered rows and columns bigger than one, while respectively one and 66 are composed by a singular row or a singular column, thus representing the specific behaviour of a specific bike stations along different days and the specific behaviour of a group of bike stations along one day.
For each found bi-cluster all the functions contained in that bi-cluster are shown together with their bi-cluster template, i.e. their average function. Figure 14 shows the membership to a bi-cluster of each element of the functional data matrix. Note that the found bi-clusters have been ordered from the biggest one to the smallest one, considering the number of included elements. Figure 15 shows the coverage of each bi-cluster in terms of percentage of contained elements.
The first bi-cluster is the bi-cluster 0, i.e. the artificial bi-cluster containing the not assigned elements. We can notice that the obtained results are able to explain the 99% of the data, while the 0.4% of the functions are not assigned to any bi-cluster. With the first ten bi-clusters we are able to explain more than the 50% of the data, while with the first 30 we are over the 70% of coverage.
Evaluating the percentage of working and weekend days for each bi-cluster,

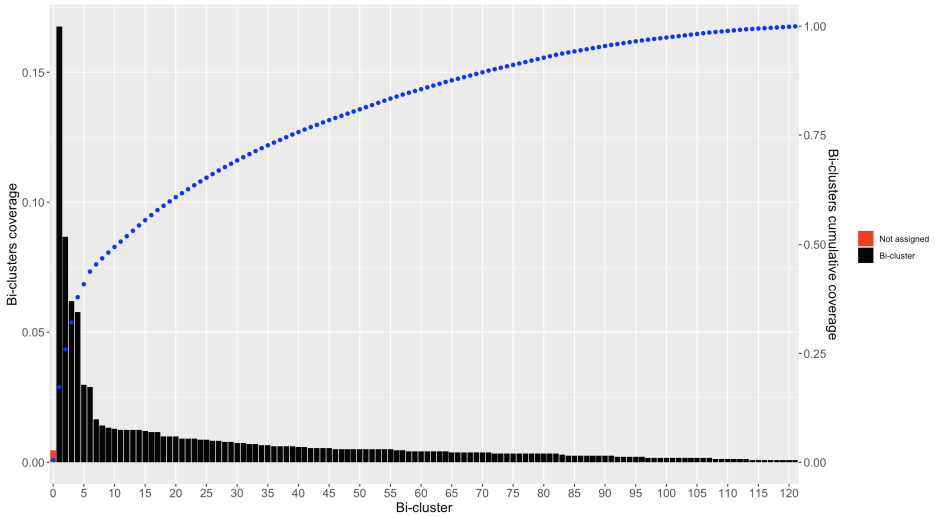

Figure 15: Coverage of each bi-cluster in terms of percentage of contained elements.

we notice that some bi-clusters cover specific patterns of the working days (e.g. bi-clusters 2 and 4) or of the weekends (e.g. bi-clusters 6 and 7), while other bi-clusters consider stations that have the same pattern during both working days and Saturday or Sunday (e.g. bi-cluster 1 and 3).
Observing the found bi-clusters and their associated functions of loading profiles, it is possible to identify different activity areas in the city according to the day of the week. In particular, among the 121 found bi-clusters, different main groups can be identified: the *constant profiles*, the *residential profiles*, the *working profiles* and the *weekend profiles*.
Bi-clusters showing a *constant profile* of usage during the whole day can underlay a no usage or a continuous replacement of bikes. Among these, some bi-clusters represent the almost-always-full stations (e.g. bi-clusters 3 and 17) and others the almost-always-empty stations (e.g. bi-clusters 1 and 9). These bi-clusters are important as they include stations in which it is not possible to drop-off or pick-up a bike respectively, thus implying users dissatisfaction.
Bi-clusters underlying *working profiles* (e.g. bi-clusters 4 and 12) coupled with bi-clusters representing *residential profiles* (e.g. bi-clusters 2 and 19) are mainly centered during working days. These two different groups show an opposite behaviour, while the first one contains stations which respectively

fill up in the morning and empty out in the evening, the stations in the second one empty out in the morning and fill up in the evening. Therefore, these two groups reveal a clear commuting behaviour of the bike sharing users which move during working days in the morning and evening rush hours. As explanatory example of this behaviour we can observe bi-clusters 2 and 4 (see Figure 16 to observe all the functions belonging to the bi-clusters with the bi-cluster template (in black), the corresponding days and bike stations location).

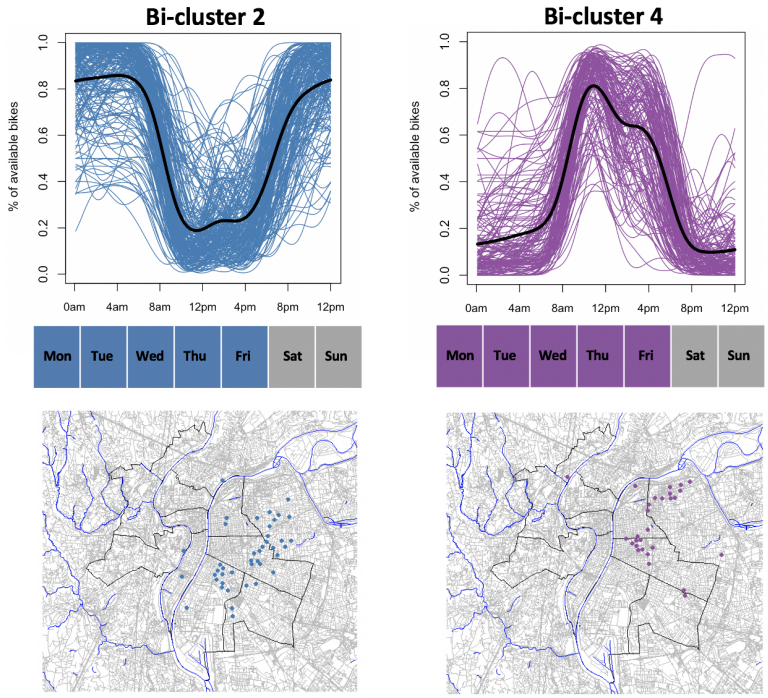

Figure 16: Functions belonging to the bi-clusters with the bi-cluster template in black (top), the corresponding days (center) and bike stations location (bottom) respectively of bi-cluster 2 (left) and 4 (right).

In details, bi-cluster 2 is composed by 42 stations and five days (from Monday to Friday). Loading profiles belonging to it represent full stations before 8a.m. and after 8p.m. and empty stations during the rest of the day. Observing the map, it is possible to notice that these stations are mostly

located in residential areas in the East of the city. Bi-cluster 4 (Figure 16(right)) is composed by 28 stations on five working days from Monday to Friday. This bi-cluster is characterized by stations which are full between 8a.m.-8p.m. and empty in the rest of the day, showing an opposite behaviour with respect to bi-cluster 2. Observing the map, it is possible to observe that these stations are mainly located in parts of the city with many companies where people are probably used to commute during the day, thus explaining this peculiar loading profile.
Another small group of bi-clusters contains bi-clusters almost covering weekend days, thus showing what we can call a *weekend profile*.

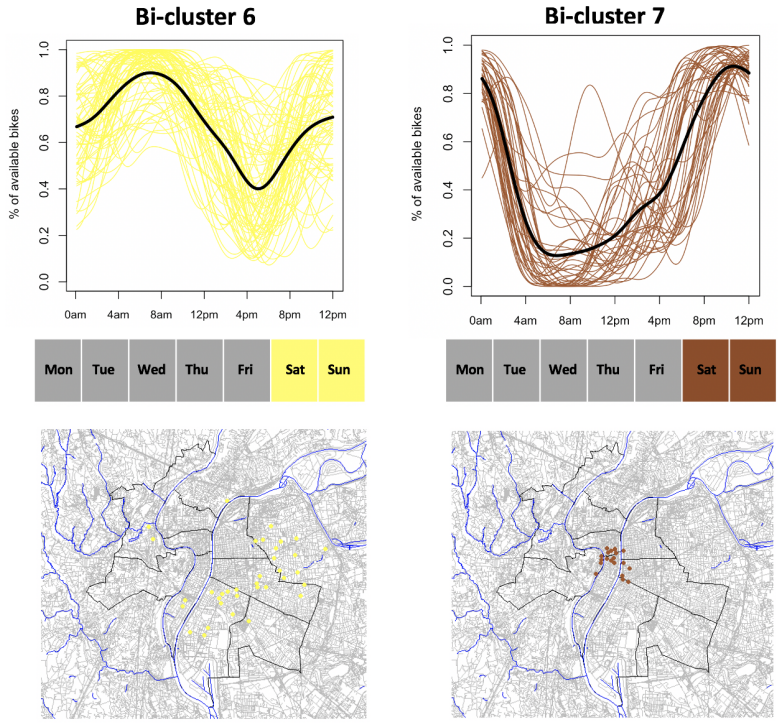

Figure 17: Functions belonging to the bi-clusters with the bi-cluster template in black (top), the corresponding days (center) and bike stations location (bottom) respectively of bi-cluster 6 (left) and 7 (right).

Focusing for example on bi-cluster 6, Figure 17(left), we can notice that this bi-cluster covers the loading profiles of 35 stations which are mainly a

subgroup of stations belonging to bi-cluster 2, i.e. residential stations, but during Saturday and Sunday. These stations are full until 8a.m., then they slowly empty out, even if not completely, until 4p.m. and finally they again refill. Bi-cluster 7 shows instead an opposite behaviour, Figure 17(right), covering 20 bike stations which slowly fill up during evening midnight and then slowly empty out during the night on Saturday and Sunday. We can explain this particular behaviour observing from the map that these stations are mainly located in the city center, very closed to River Sāone banks, where there are many shops and bars, therefore they are probably used by people going out clubbing and then coming back home late at night.

7 Conclusion

In this paper a new bi-clustering technique for functional data is presented to group simultaneously rows and columns of a data matrix whose elements are functions on a continuous domain. The presented algorithm is non parametric and very flexible, allowing to discover different bi-clustering depending on the problem at hand. In details a bi-cluster can be defined as group of functions with similar average behaviour considering or not the rows/columns components. The aim is to have a method which can be applied to real functional dataset, in which not all the elements belong to a bi-cluster and where data are not necessarily Gaussian, thus modelling assumptions on the data are hardly verified. Moreover, since another common problem in real functional datasets is the misalignment of the data, in the FunCC algorithm an alignment procedure is also implemented. Therefore, compared to other methods presented in the literature, the FunCC algorithm presents some advantages being totally non parametric, non exhaustive (thus not forcing all elements in the data matrix to be in one bi-cluster) and allowing for a registration step. Empirical simulations are performed, clearly showing the potential of the FunCC method.
The algorithm is also applied on real dataset, the bike sharing system of Lyon, with the aim of providing useful information for the correct management of the service. Through our functional bi-clustering technique we discover subgroups of stations and days with similar behaviour. Clear patterns of usage are revealed, allowing to segment the bike stations into different usage profiles (for example the residential and industrial profiles) according to the days of the week, identifying when the bike demand is higher. Moreover, groups

of stations always full or always empty are highlighted, revealing some criticalities in the service.
The algorithm proposed in this work is implemented in the R package *FunCC*, available at `https://cran.r-project.org/web/packages/FunCC/index.html`.

References

[1] J. O. Ramsay, Functional data analysis, Encyclopedia of Statistical Sciences 4.

[2] F. Ferraty, P. Vieu, Nonparametric functional data analysis: theory and practice, Springer Science & Business Media, 2006.

[3] B. Pontes, R. Giráldez, J. S. Aguilar-Ruiz, Biclustering on expression data: A review, Journal of biomedical informatics 57 (2015) 163–180.

[4] J. Jacques, C. Preda, Functional data clustering: a survey, Advances in Data Analysis and Classification 8 (3) (2014) 231–255.

[5] Y. B. Slimen, S. Allio, J. Jacques, Model-based co-clustering for functional data, Neurocomputing 291 (2018) 97–108.

[6] C. Bouveyron, L. Bozzi, J. Jacques, F.-X. Jollois, The functional latent block model for the co-clustering of electricity consumption curves, Journal of the Royal Statistical Society: Series C (Applied Statistics) 67 (4) (2018) 897–915.

[7] G. Govaert, M. Nadif, Co-clustering: models, algorithms and applications, John Wiley & Sons, 2013.

[8] Y. Cheng, G. M. Church, Biclustering of expression data., in: Ismb, Vol. 8, 2000, pp. 93–103.

[9] J. O. Ramsay, X. Li, Curve registration, Journal of the Royal Statistical Society: Series B (Statistical Methodology) 60 (2) (1998) 351–363.

[10] G. M. James, et al., Curve alignment by moments, The Annals of Applied Statistics 1 (2) (2007) 480–501.

[11] N. Altman, J. Villarreal, Self-modelling regression for longitudinal data with time-invariant covariates, Canadian Journal of Statistics 32 (3) (2004) 251–268.

[12] M. J. Lindstrom, D. M. Bates, Nonlinear mixed effects models for repeated measures data, Biometrics (1990) 673–687.

[13] D. Kaziska, A. Srivastava, Gait-based human recognition by classification of cyclostationary processes on nonlinear shape manifolds, Journal of the American Statistical Association 102 (480) (2007) 1114–1124.

[14] L. M. Sangalli, P. Secchi, S. Vantini, A. Veneziani, A case study in exploratory functional data analysis: geometrical features of the internal carotid artery, Journal of the American Statistical Association 104 (485) (2009) 37–48.

[15] V. Vitelli, L. M. Sangalli, P. Secchi, S. Vantini, Functional clustering and alignment methods with applications, Communications in Applied and Industrial Mathematics 1 (1) (2010) 205–224.

[16] A. Prelić, S. Bleuler, P. Zimmermann, A. Wille, P. Bühlmann, W. Gruissem, L. Hennig, L. Thiele, E. Zitzler, A systematic comparison and evaluation of biclustering methods for gene expression data, Bioinformatics 22 (9) (2006) 1122–1129.

[17] S. Vantini, On the definition of phase and amplitude variability in functional data analysis, Test 21 (4) (2012) 676–696.

[18] A. Srivastava, W. Wu, S. Kurtek, E. Klassen, J. S. Marron, Registration of functional data using fisher-rao metric, arXiv preprint arXiv:1103.3817.

[19] C. Bouveyron, E. Côme, J. Jacques, et al., The discriminative functional mixture model for a comparative analysis of bike sharing systems, The Annals of Applied Statistics 9 (4) (2015) 1726–1760.

[20] T. Hastie, R. Tibshirani, J. Friedman, The Elements of Statistical Learning, Springer Series in Statistics, Springer New York Inc., New York, NY, USA, 2001.

A Appendix section

Referring to the Cheng and Church model ([8]), given a data matrix A composed by n rows and m columns, a bi-cluster $B(I, J)$ is a set of rows I and a set of columns J such that each element a_{ij} in the bi-cluster can be expressed as: $a_{ij} = \mu + \alpha_i + \beta_j$ with $i \in I$ and $j \in J$, where μ is the average value in the bi-cluster and α_i and β_j are respectively the residue value between row and column average value and the bi-cluster total average value a_{IJ}. In details:

- $\mu = \frac{1}{|I||J|} \sum_{i \in I, j \in J} a_{ij}$
- $\alpha_i = \frac{1}{|J|} \sum_{j \in J} a_{ij} - a_{IJ}$
- $\beta_j = \frac{1}{|I|} \sum_{i \in I} a_{ij} - a_{IJ}$

The mean squared residue score of a bi-cluster $B(I, J)$ is expressed as:

$$H(I, J) = \frac{1}{|I||J|} \sum_{i \in I, j \in J} (a_{ij} - \bar{a}_{ij})^2$$

where $\bar{a}_{ij} = \mu + \alpha_i + \beta_j$. To find a set of bi-clusters in the data, a deterministic and greedy approach is employed returning as output bi-clusters having the maximal dimension in terms of number of rows and columns according to the minimization of a mean squared residue score H. Specifically, a sub-matrix $B(I, J)$ is a bi-cluster if its H-score is smaller than a given threshold δ taken as input parameter by the algorithm. The aim of the algorithm is to find a maximal submatrix with a low H score. The Cheng and Church algorithm can be considered as a three steps procedure as expressed in Algorithm 3. Initially a nodes deletion step is performed removing rows and columns of the data matrix to minimize the mean squared residue H. Then the result of the deletion is modified by adding nodes which do not impact on the score, obtaining as output the maximal bicluster below the chosen threshold δ is identified. Details of Multiple and Single Node Deletion and Nodes Addition procedures are reported in Algorithm 4, 5 and 6. In details for the Multiple Node Deletion step, Algorithm 4, a new parameter θ is considered and the rows and columns with scores beyond a value identified by the threshold

$\theta \cdot H(I, J)$ are removed. This procedure is very fast but may return too much shrunk matrices. The Single Node Deletion step, in Algorithm 5, is instead a lower procedure deleting one node at the time. While $H(I, J)$ is bigger than the threshold δ and other rows and columns can be removed, the algorithm proceeds evaluating the rows and columns scores and removing the row or the column with the higher score, i.e. the one which largely contribute to the score H.

After the deletion phase, the resulting bi-cluster may not be maximal; hence an addition step is performed, trying to add all the rows and the columns that do not increase the score H. In this step the procedure tries to add also the anti-correlated or inverted rows. This step was introduced because the original Cheng and Church algorithm was developed for gene expression data. An anti-correlated or inverted row in a gene expression data may indeed represent a negatively regulated genes that is of interest when finding a bi-cluster.

Finally the algorithm is iterated without considering the results already found; to do so a masking procedure is performed. This masking procedure consists in replacing all the elements in the matrix already assigned to one bi-cluster with random values. This makes quite unlikely that already assigned elements would be reassigned to other bi-clusters, but it does not ensure it.

Algorithm 3: Cheng and Church algorithm

Input: (n, m) matrix A whose elements are numbers a_{ij}
$\delta \geq 0$ the maximum acceptable mean residue score
$\theta \geq 1$ a parameter for multiple node deletion

Result: A set $\mathbb{B}$ of Bi-clusters $B(I, J)$ with mean residue score lower than δ

Set:

- $\mathbb{B} = \{\}$ set of bi-clusters found
- $(I, J) = \{(i, j) : i \in 1...n, j \in 1...m\}$ set of not assigned elements

while *A new bi-cluster* $B(I, J)$ *is found* **do**

On A do:

1. apply Algorithm 4 to perform **Multiple Node Deletion**.
2. apply Algorithm 5 to perform **Single Node Deletion**.
3. apply Algorithm 6 to perform **Node Addition**.

if *A new bi-cluster* $B(I, J)$ *is found* **then**

- $\mathbb{B} = \mathbb{B} \cup B(I, J)$
- mask the assigned data

end

end

Algorithm 4: Multiple Node Deletion

Input: (n, m) matrix A whose elements are numbers a_{ij}
$\delta \geq 0$ the maximum acceptable mean residue score
$\theta \geq 1$ a parameter for multiple node deletion

Result: A submatrix $A(I, J)$ of A with a score no larger than δ

Set:

- $A(I, J)$=A
- $(I, J) = \{(i, j) : i \in 1...n, j \in 1...m\}$ set elements in $A(I, J)$
- evaluate $H(I, J)$

while $H(I, J) > \delta$ **do**

On $A(I, J)$ do:

1. Evaluate $H(I, J)$
2. Remove the rows $i \in I$ with $\frac{1}{|J|} \sum_{j \in J} (a_{ij} - \bar{a}_{ij})^2 > \theta H(I, J)$
3. Evaluate $H(I, J)$
4. Remove the columns $j \in J$ with $\frac{1}{|I|} \sum_{i \in I} (a_{ij} - \bar{a}_{ij})^2 > \theta H(I, J)$

if *No deletion is performed* **then**
STOP
end
end

Algorithm 5: Single Node Deletion

Input: (n, m) matrix A whose elements are numbers a_{ij}
$\delta \geq 0$ the maximum acceptable mean residue score

Result: A submatrix $A(I, J)$ of A with a score no larger than δ

Set:

- $A(I, J)$=A
- $(I, J) = \{(i, j) : i \in 1...n, j \in 1...m\}$ set elements in $A(I, J)$
- evaluate $H(I, J)$

while $H(I, J) > \delta$ **do**

On $A(I, J)$ do:

1. Find the row $i \in I$ with the largest $d_{iJ} = \frac{1}{|J|} \sum_{j \in J} (a_{ij} - \bar{a}_{ij})^2$
2. Find the column $j \in J$ with the largest $d_{Ij} = \frac{1}{|I|} \sum_{i \in I} (a_{ij} - \bar{a}_{ij})^2$

if $d_{iJ} > d_{Ij}$ **then**
  delete row i
else
  delete column j
end

end

Algorithm 6: Node Addition

Input: (n, m) matrix A whose elements are numbers a_{ij}
I and J representing the submatrix $A(I, J)$

Result: I' and J' such that $I' \subset I$ and $J' \subset J$ with the property that $H(I', J') \leq H(I, J)$

Set:

- evaluate $H(I, J)$

while *A row or a column is added* **do**

On $A(I, J)$ do:

1. Add the columns $j \notin J$ with $\frac{1}{|I|} \sum_{i \in I} (a_{ij} - \bar{a}_{ij})^2 \leq H(I, J)$
2. evaluate $H(I, J)$
3. Add the rows $i \notin I$ with $\frac{1}{|J|} \sum_{j \in J} (a_{ij} - \bar{a}_{ij})^2 \leq H(I, J)$
4. For each row $i \notin I$ add its inverse if $\sum_{j \in J} (-a_{ij} - (a_{IJ} - \alpha_i + \beta_j))^2 \leq H(I, J)$

end

MOX Technical Reports, last issues

Dipartimento di Matematica
Politecnico di Milano, Via Bonardi 9 - 20133 Milano (Italy)

67/2020 Caramenti, L.; Menafoglio, A.; Sgobba, S.; Lanzano, G.
Multi-Source Geographically Weighted Regression for Regionalized Ground-Motion Models

68/2020 Galvani, M.; Torti, A.; Menafoglio, A.; Vantini S.
FunCC: a new bi-clustering algorithm for functional data with misalignment

66/2020 Didkovsky, O.; Ivanov, V.; Papini, M.; Longoni, L.; Menafoglio, A.
A comparison between machine learning and functional geostatistics approaches for data-driven analyses of solid transport in a pre-Alpine stream

65/2020 Di Gregorio, S.; Vergara, C.; Montino Pelagi, G.; Baggiano, A.; Zunino, P.; Guglielmo, M.; Fu
Prediction of myocardial blow flow under stress conditions by means of a computational model

64/2020 Fiz, F.; Viganò, L.; Gennaro, N.; Costa, G.; La Bella, L.; Boichuk A.; Cavinato, L.; Sollini, M.;
Radiomics of Liver Metastases: A Systematic Review

63/2020 Tuveri, M.; Milani, E.; Marchegiani, G.; Landoni, L.; Torresani, E.; Capelli, P.; Sperandio, N.;
HEMODYNAMICS AND REMODELING OF THE PORTAL CONFLUENCE IN PATIENTS WITH CANCER OF THE PANCREATIC HEAD: A PILOT STUDY

62/2020 Massi, M. C.; Ieva, F.
Representation Learning Methods for EEG Cross-Subject Channel Selection and Trial Classification

61/2020 Pozzi, S.; Redaelli, A.; Vergara, C.; Votta, E.; Zunino, P.
Mathematical and numerical modeling of atherosclerotic plaque progression based on fluid-structure interaction

60/2020 Lupo Pasini, M; Perotto, S.
Hierarchical model reduction driven by a Proper Orthogonal Decomposition for parametrized advection-diffusion-reaction problems

59/2020 Massi, M.C.; Franco, N.R; Ieva, F.; Manzoni, A.; Paganoni, A.M.; Zunino, P.
High-Order Interaction Learning via Targeted Pattern Search