Computational Statistics
https://doi.org/10.1007/s00180-024-01490-5

SOFTWARE ARTICLE

FPDclustering: a comprehensive R package for probabilistic distance clustering based methods

Cristina Tortora[1] · **Francesco Palumbo**[2]

Received: 11 September 2023 / Accepted: 17 March 2024

Abstract
Data clustering has a long history and refers to a vast range of models and methods that exploit the ever-more-performing numerical optimization algorithms and are designed to find homogeneous groups of observations in data. In this framework, the probability distance clustering (PDC) family methods offer a numerically effective alternative to model-based clustering methods and a more flexible opportunity in the framework of geometric data clustering. Given n J-dimensional data vectors arranged in a data matrix and the number K of clusters, PDC maximizes the joint density function that is defined as the sum of the products between the distance and the probability, both of which are measured for each data vector from each center. This article shows the capabilities of the PDC family, illustrating the R package `FPDclustering`.

Keywords Probabilistic distance clustering · Soft clustering · Mixed-type data

1 Introduction

In an extended and inclusive definition, given a set of multivariate units, the final aim of any clustering approach is to recognize similar units and produce a meaningful grouping of objects in the data (Aggarwal 2014, Chap. 1). Data clustering is considered unsupervised learning and is formulated as a numerical optimization problem that exploits computers' computing power to maximize a given within-group homogeneity (or between-group heterogeneity) criterion. In many domains, the complexity of the data sets available has risen. This has devised a need for methods that do not rely on assumptions on the distribution of the data and can work on variables recorded

✉ Francesco Palumbo
fpalumbo@unina.it

Cristina Tortora
cristina.tortora@sjsu.edu

1 San José State University, San Jose, CA, USA

2 University of Naples Federico II, Naples, Italy

Published online: 15 May 2024

on diverse measurement scales (nominal, ordinal, and metric), namely mixed-type data (Megahed and Jones-Farmer 2015). This has generated a renewed interest in geometrical clustering approaches and new methods for the treatment of mixed-data (Ahmad and Khan 2019; Van de Velden et al. 2019; Mbuga and Tortora 2021), in particular, towards numerical optimization algorithms that can effectively exploit the last generation multicore CPUs (Oyewole and Thopil 2023; Tjur 2011).

Generally, the number K of groups is unknown a priori, and the choice of the *best* K remains an open issue. Because of the huge number of possible data partitions for any fixed K, even in the geometrical context and with a moderately large data set, finding the optimal solution that groups the data into homogeneous clusters remains a numerically untreatable problem, suggesting to focus on methods computationally feasible but that could lead to sub-optimal solutions. In this framework, partitioning methods are among the most efficient. They generally start from a given number of (randomly) selected centroids and, until convergence, iterate two steps: (i) (re)assigns the units to the closest centroids; (ii) updates the centroids. See Gordon (1999) for an exhaustive treatment.

Therefore, when dealing with mixed data types, with many variables and a huge number of units, namely complex data treating, partitioning algorithms remain more appealing, flexible, and likely the most broadly used. In this family of methods, beyond any doubt, the most known and widely used approach is the K-means algorithm. Interested readers may refer to Jain (2009) for a historical perspective of the K-means and to the more recent paper from Ikotun et al. (2022) that reconsiders the K-means algorithm in the era of Big Data. Several extensions of K-means clustering for mixed-type data have been proposed, the most popular being k-prototypes (Huang 1998) and KAy-means for MIxed LArge data (KAMILA) (Foss and Markatou 2018). A fuzzy version of k-means, where the cluster memberships are belonging probabilities, is called fuzzy k-means Bezdek (2013). Given the popularity of K-means clustering, the algorithm is available in most statistical software. In R (R Core Team 2016), the function `kmeans` is available in the core package `stats`, while some extensions that jointly combine dimensionality reduction and K-means, namely Factorial K-means (Vichi and Kiers 2001) and reduced K-means (De Soete and Carroll 1994) are implemented in the package `clustrd` (Markos et al. 2019). The extensions for mixed type data, k-prototypes, and KAMILA are available using the packages `clustMixType` (Szepannek 2018) and `kamila` (Foss and Markatou 2018), respectively. Fuzzy k-means is available in the package `fclust` (Ferraro et al. 2019).

Within the large family of geometric clustering algorithms and their applications to diverse data types, this work considers probabilistic distance (PD) clustering (Iyigun and Ben-Israel 2008; Ben-Israel and Iyigun 2008). PD clustering is a distribution-free, probabilistic clustering method that assigns the units to the given K clusters by solving a numerical optimization problem. The algorithm assigns the units to the clusters according to a K-dimensional membership array, whose values sum to one for each unit and may be interpreted as the probabilities of belonging to a cluster, where K indicates the number of clusters (defined a priori). It grounds on the following geometric assumption: given a point and K cluster centers lying in any J dimensional space, the product between the belonging probability and the distance for any given point to each cluster center is a constant. The value of the constant is inversely pro-

portional to the classifiability of the point; therefore, PD clustering aims to find those centers that maximize the overall classifiability. This is equivalent to minimizing the sum of the products between the distance from the cluster centers and the probability that the point belongs to the cluster computed for each point in the data. Because these two quantities are unknown and mutually dependent, the algorithm runs over two steps: alternately, it computes the distances of each point from all the centers and then updates the belonging probabilities and, consequently, the centers. PD clustering is computationally efficient and extremely flexible; moreover, its factor reformulation, proposed by Tortora et al. (2016), can treat many continuous variables.

This paper reviews the whole PD clustering family; it offers a perspective on PD clustering usability in various data contexts. To this end, it gives a detailed and illustrated introduction to the method by presenting the recently updated `FPDclustering` package functionalities (Tortora et al. 2024). The package is enriched with effective interpretation wizards, helping interpret the results that have never been systematically presented before. The paper allows the readers to appreciate the PD clustering capabilities under several conditions; three applications with real data sets are used.

The remainder of the paper is organized as follows: Sect. 2 contains an overview of the discussed methods, with appropriate references for interested readers. Section 3 details the algorithms used per each method while Sect. 4 gives an introduction of the package `FPDclustering`. Section 5 shows the package's functionalities on three real data sets, and section 6 concludes the paper.

2 Statistical methods

2.1 Probabilistic distance clustering

Let $\mathbf{X}$ be a data matrix with n units, J variables, and generic row vector $\mathbf{x}_i$ $(i = 1, \ldots, n)$, given K clusters that are assumed not to be empty having as center the row vectors $\mathbf{c}_k$ $(k = 1, \ldots, K)$, PD-clustering (Ben-Israel and Iyigun 2008) criterion uses two quantities: the distance, $d(\mathbf{x}_i, \mathbf{c}_k)$, of each data point $\mathbf{x}_i$ from the K cluster centers $\mathbf{c}_k$, and the probabilities, $p(\mathbf{x}_i, \mathbf{c}_k)$, of each point belonging to a cluster. Then, the probability of any generic point belonging to each cluster $p(\mathbf{x}_i, \mathbf{c}_k)$ is assumed to be inversely proportional to the distance from the centers of the clusters $d(\mathbf{x}_i, \mathbf{c}_k)$. The relation between them is the basic principle of the method. Formally, replacing $p(\mathbf{x}_i, \mathbf{c}_k)$ with p_{ik} and $d(\mathbf{x}_i, \mathbf{c}_k)$ with d_{ik} for the sake of compactness the following equation

$$d_{ik}\, p_{ik} = F(\mathbf{x}_i), \tag{1}$$

holds for any $i = 1, \ldots, n$ and any set of $k = 1, \ldots, K$ centers. Equation 1 states that the product between the distances and the probabilities is a constant depending only on $\mathbf{x}_i$. Therefore, when d_{ik} decreases, then the probability p_{ik} of the point $\mathbf{x}_i$ belonging to the cluster k increases. The distance d_{ik} can be computed according to different criteria; the squared norm is one of the most commonly used

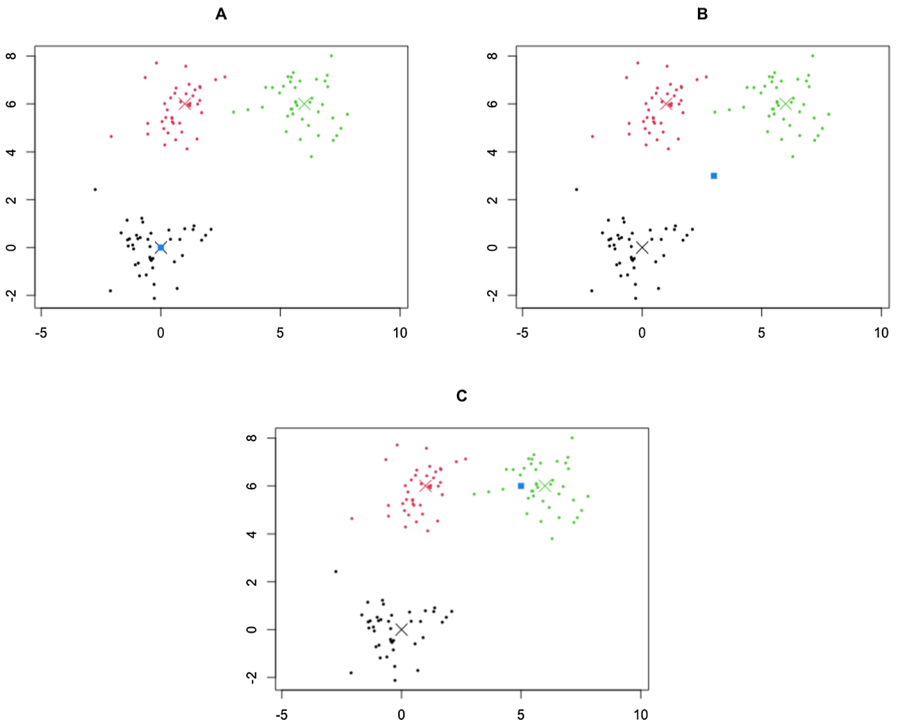


Fig. 1 Toy data set to illustrate PDC. The dots are the data colored based on cluster membership, the X symbols are the centers of the clusters, and the blue square is the point considered for clustering

Table 1 Values of d_{ik}, p_{ik}, and $F(\mathbf{x}_i)$ corresponding to the blue squared points in Fig. 1 A, B, and C

	A (0, 0)			B (3, 3)			C (5, 6)		
	k=1	k=2	k=3	k=1	k=2	k=3	k=1	k=2	k=3
d_{ik}	0.00	6.08	8.49	4.24	3.61	4.24	7.81	4.00	1.00
p_{ik}	1.00	0.00	0.00	0.31	0.37	0.31	0.09	0.18	0.73
$F(\mathbf{x}_i)$	0.00	0.00	0.00	1.34	1.34	1.34	0.73	0.73	0.73

The coordinates of the points are in parentheses

$$d_{ik} = ||\mathbf{x}_i - \mathbf{c}_k||,$$

and $F(\mathbf{x}_i)$ measures the classifiability of the point $\mathbf{x}_i$ with respect to the cluster centers $\mathbf{c}_k$, $k = 1, \ldots, K$.

Summing $F(\mathbf{x}_i)$ over n defines the classification criterion, namely the *Joint Distance Function* (JDF); it measures the closeness of $\mathbf{x}_i$ from all the cluster centers. If a point coincides with one cluster center $F(\mathbf{x}_i) = 0$; if all the distances between the point $\mathbf{x}_i$ and the centers of the clusters are equal, then $F(\mathbf{x}_i) = d_{ik}/k$ and all the probabilities of belonging to each cluster are equal. The smaller the JDF value, the higher the point classifiability to one cluster. The toy example in Fig. 1 shows three simple scenarios.

The dots are the data colored based on cluster membership, and the symbols X in the plot are the centers of the clusters, i.e., (0,0), (1,6), and (6,6). The blue square is the point considered for clustering of coordinates (0,0) in Figure A, (3,3) in Figure B, and (5,6) in Figure C. The values of d_{ik}, p_{ik}, and $F(\mathbf{x}_i)$ are shown in Table1. In Figure A, the point coincides with the center of cluster number 1; therefore, the distance from that cluster is 0, and the probability of belonging to cluster 1 is 1. In scenario B, the point is approximately at the same distance from all the centers, while in scenario 3, the point is closer to cluster 3. By construction, for each point, the value of $F(\mathbf{x}_i)$ is the same for each cluster; it is 0 in scenario A, of best classifiability, and highest in scenario B, of worst classifiability.

Using the fact that $p_{ik}d_{ik}$ is a constant, the equation for the probabilities can be obtained as

$$p_{ik} = \frac{\prod_{m \neq k} d_{im}}{\sum_{m=1}^{K} \prod_{r \neq m} d_{ir}}.$$

The clustering problem is solved by minimizing the sum of $F(\mathbf{x_i})$ over i; this summation does not depend on k.

To ensure convexity, Ben-Israel and Iyigun (2008) suggest using p_{ik}^2 instead of p_{ik}, then the clustering problem solved by minimizing the JDF becomes:

$$\text{JDF} = \sum_{i=1}^{n} \sum_{k=1}^{K} ||\mathbf{x}_i - \mathbf{c}_k|| p_{ik}^2. \tag{2}$$

The summation over k is introduced to avoid the effect of rounding errors. The row vectors $\mathbf{c}_k$ that minimize the JDF are

$$\mathbf{c}_k = \sum_{i=1}^{n} \left(\frac{u_k(\mathbf{x}_i)}{\sum_{j=1}^{n} u_k(\mathbf{x}_j)} \right) \mathbf{x}_i, \tag{3}$$

where

$$u_k(\mathbf{x}_i) = \frac{p_{ik}^2}{d_{ik}}.$$

Each unit is assigned to the k^{th} cluster according to the highest probability.

The most used fuzzy clustering method is fuzzy k-means (FKM) (Bezdek 2013), which minimizes $\sum_{i=1}^{n} \sum_{k=1}^{K} u_{ik}^{v} ||\mathbf{x}_i - \mathbf{c}_k||$, where u_{ik} is a positive membership grade with $\sum_{k=1}^{K} u_{ik} = 1$, and v is a fuzzier. Although setting $v = 2$ makes the equation equal to Formula 2, the lack of the constraint on the product between distance and probability impacts the formulas for computing the centers and the probabilities. Figure 2 shows an example of the application of PDC and FKM on a toy data set generated by merging two bivariate data sets pseudo-randomly generated from two normal distributions with different means. The size of the points is proportional to the probability of belonging to the cluster to which they are assigned.

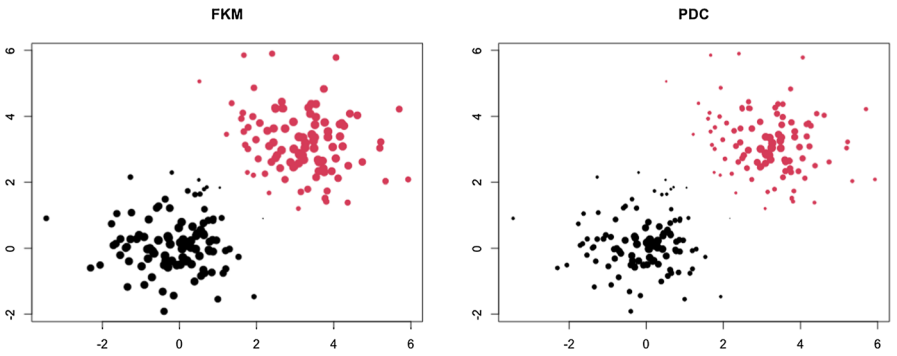


Fig. 2 Toy data set to illustrate the difference between FKM and PDC. The data are colored based on cluster membership; the size of the points is proportional to the belonging probability to the cluster they are assigned to

Focusing on FKM, Fig. 2 shows that the sizes of the points, i.e., the cluster belonging probabilities, are smaller in between clusters. Points in between two clusters have smaller probabilities of belonging to either cluster and, consequently, a smaller weight when computing the centers. In PDC, the probabilities are lower as the points get further from the center in any direction, making it robust by down-weighting.

2.2 PDQ: PDC adjusted for cluster size

When the true data structure consists of clusters of different sizes, as PDC does not consider the cluster size, it may fail to identify the most appropriate cluster memberships. To overcome this issue, Iyigun and Ben-Israel (2008) proposed PDC adjusted for cluster size (PDQ). PDQ relies on the following adjusted definition of $F(\mathbf{x}_i)$

$$\frac{p_{ik}^2 d_{ik}}{q_k} = F(\mathbf{x}_i), \tag{4}$$

where q_k is the cluster size, with the constraint that $\sum_{k=1}^{K} q_k = n$. Then, p_{ik} can be computed according to the following expression:

$$p_{ik} = \frac{\prod_{m \neq k} d_{im}/q_m}{\sum_{m=1}^{K} \prod_{r \neq m} d_{ir}/q_r}.$$

The cluster size is considered a variable, and the value of q_k that minimizes (4) is

$$q_k = n \frac{\left(\sum_{i=1}^{n} d_{ik} p_{ik}^2\right)^{1/2}}{\sum_{k=1}^{K} \left(\sum_{i=1}^{n} d_{ik} p_{ik}^2\right)^{1/2}},$$

for $k = 1, \ldots, K-1$, and $q_K = n - \sum_{k=1}^{K-1} q_k$.

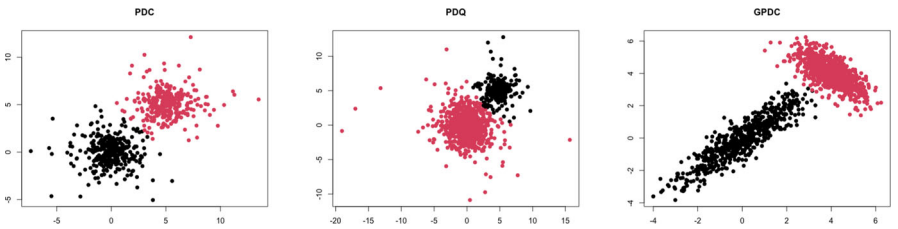


Fig. 3 Toy data sets to illustrate with which kind of cluster shape PDC, PDQ, and GPDC perform best. The data are colored based on cluster membership

2.3 Flexible PDQ

In its original formulation, PD clustering does not account for the correlation between variables and is sensible to outliers. Tortora et al. (2020) proposed a kernel version of the PD clustering to increase the method flexibility to data sets characterized by correlated variables and outliers, based on a symmetric unimodal density function family $f(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\theta})$, with location parameter vector $\boldsymbol{\mu}$ and parameter vector $\boldsymbol{\theta}$, which specify the function. As the PD clustering cannot be separated from a definition of a dissimilarity or a distance measure, Tortora et al. (2020) proposed the following dissimilarity measure

$$d_{ik} = \log(M_k f(\boldsymbol{x}_i; \boldsymbol{\mu}_k, \boldsymbol{\theta}_k)^{-1}), \tag{5}$$

where M_k is defined as $M_k = \max\{f(\boldsymbol{x}; \boldsymbol{\mu}_k, \boldsymbol{\theta}_k)\}$, for any $k = 1 \ldots, K$. According to the dissimilarity measure in 5, two PD clustering variants were developed, Gaussian PDC (GPDC) and Student-t PDC (TPDC), based on the multivariate Gaussian and Student-t distributions, respectively.

Figure 3 shows three fictitious situations in which PDC, PDQ, and GPDC perform best, respectively.

2.4 PDC for frequency data

The function `PDQ` includes an option to accommodate frequency data. Let $\mathbf{F}$ be a contingency table, i.e., the data represent frequencies, with n rows and J columns, having general term f_{ij}. The χ^2 distance is defined as follows:

$$d_{ii'} = \sqrt{\sum_{j=1}^{J} \frac{1}{f_{\cdot j}} \left(\frac{f_{ij}}{f_{i\cdot}} - \frac{f_{i'j}}{f_{i'\cdot}} \right)^2},$$

where $f_{i\cdot}$ and $f_{\cdot j}$ are the generic row marginal and column marginal frequencies, respectively.

2.5 PDC for mixed-type data

When **X** is characterized by mixed-type data, the function `PDQ` using the Gower's distance should be used. A different kind of distance is used for different types of variables, and the centers are updated using equation (3) for continuous variables, while the mode is used for categorical and binary variables. More details about PDQ for mixed-type data can be found in Tortora and Palumbo (2022).

2.6 High dimensional data sets

The package also includes Factor PDC (FPDC) for data sets with many variables. The iterative algorithm alternately performs a linear transformation of original variables into a reduced number of orthogonal ones (factors), and the PD clustering onto the reduced space minimizes the JDF at each step (Tortora et al. 2013; Tortora 2011).

Tortora (2011) and Tortora et al. (2013) formally demonstrated the accordance between the Tucker3 method (Kroonenberg 2008) with the PD clustering JDF criterion, and empirical evidence demonstrates that the algorithm ensures convergence to a local minimum, at least. Specifically, let us define with **G** a 3-way $n \times J \times K$ distance array of general elements $g_{ijk} = |x_{ij} - c_{kj}|$, where n is the number of units, J the number of variables, and K the number of clusters. The Tucker3 method decomposes the three-way array **G** in three matrices, one for each way, in a full core array plus an error term. The generic term g_{ijk} is then approximated into a lower dimensional space according to the following expression

$$g^*_{ijk} = |x^*_{ij} - c^*_{kj}| = \sum_{r=1}^{R}\sum_{q=1}^{Q}\sum_{s=1}^{S} \lambda_{rqs}(u_{ir}b_{jq}v_{ks}),$$

where λ_{rqs} is a weighting parameter, R, Q, and S are the number of retained components (factors) for each of the three ways of **G**.

The algorithm solves the optimization problem by alternatively computing the coordinates of the observations in the space of variables obtained through Tucker3 decomposition and then running the PD clustering until the JDF

$$JDF = \sum_{i=1}^{n}\sum_{q=1}^{Q}\sum_{k=1}^{K} p^2_{ik}|x^*_{iq} - c^*_{kq}|,$$

no longer decreases, where x^*_{iq} and c^*_{kq} are the data and the centers' projections in the reduced subspace.

As in all factor methods, factor axes in the Tucker3 model are sorted according to explained variability. The first factor axes explain the greatest part of the variability. Therefore, the choice of the parameters R, Q and S is a ticklish issue, as the aim is to minimize the number of factors retaining the significant value of the explained variability according to the specific context. The package `FPDclustering` has a tool to help select the best number of factors.

3 PDC algorithm

All the described PD clustering-based algorithms are iterative methods that get the solution through successive approximations. The PD clustering algorithm requires a minimum set of input parameters that consist of the $n \times J$ data matrix $\mathbf{X}$ and the number of clusters K. PDC, PDQ, GPDC, and TPDC common core procedure can be summarized in the following steps:

0. center initialization;
1. computation of distance matrix;
2. computation of belonging probabilities array and other parameters where appropriate;
3. update of the centers,
4. iteration of steps 1–3 until convergence is reached.

For PD clustering, the function `PDC` sets the first two centers at the combination of all minimum and maximum values of the input variables in $\mathbf{X}$, respectively. If $K > 2$, then `PDC` randomly chooses the other $K - 2$ centers.

The convergence is reached when, between two successive iterations, the JDF or the centers' matrix variation is in the order ≤ 0.001. The algorithm also stops and returns a warning if the convergence is not reached after 1000 iterations. Given the vectors of the centers $\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_K$, points are assigned to the clusters according to the highest probability. For PDQ, the function `PDQ` initializes the centers using the different options available for the parameter `ini`. Available options are `random`, `kmedoids` (Van der Laan et al. 2003), and `center` (the user provides the matrix of the center starts in the function input parameters). When choosing the option `kmedoids`, if the data are mixed-type, the function automatically uses the k-modes algorithm (Chaturvedi et al. 2001) for the categorical variables. `PDQ` also has another optional input, `dist`, to select the distance measure used (options available are Euclidean (‘`euc`’), Gower (‘`gower`’), and chi-square (‘`chi`’)). If the data are of mixed type, the following inputs need to be used to specify the column indices corresponding to each variable type, specifically: `ord` for ordinal variables, `cat` for categorical variables, `bin` for binary variables, and `cont` for continuous variables. The optional input `w`, a numerical vector the same length as the columns of the data, can be used to give different weights to each variable.

The functions `GPDC` and `TPDC` also have the option to change initialization criteria using the parameter `method`; available options are `random`, `kmedoids`, and `PDC`. If random is chosen, the user can use `nr` to specify the number of random starts. The number of iterations until convergence for `GPDC` and `TPDC` is set to 100 and can be changed using the parameter `iter`.

The FPDC algorithm has a higher complexity; see Tortora et al. (2016) for details. Moreover, it requires two more input parameters than PDC: the number of factors for variables and the number of factors for units. The option `TuckerFactors`, available in the `FPDclustering`, allows calling a graphical procedure to help the user to choose heuristically. In particular, the plot represents the explained variance at several combinations of retained factors for the two ways (units and variables, with

Table 2 Summary of the functions contained in the `FPDclustering` package version 2.3.1 available at the CRAN (The Comprehensive R Archive Network) http://cran.r-project.org/

Function	Data type	Description	Citations
`PDC`	Continuous	PD clustering	Ben-Israel and Iyigun (2008)
`PDQ`	Continuous /mixed type	PD clustering adjusted for cluster size	Ben-Israel and Iyigun (2008) Tortora and Palumbo (2022)
`GPDC`	Continuous	Gaussian PD-Clustering, accounts for correlation and variance	Tortora et al. (2020)
`TPDC`	Continuous	Student-t PD-Clustering, accounts for correlation, variance, and outliers	Tortora et al. (2020)
`FPDC`	Continuous/high dimensions	Factor PD-clustering dimension reduction and clustering	Tortora et al. (2016)
`TuckerFactors`	Continuous/high dimensions	Choice of the number of Tucker 3 factors for FPDC	Tortora et al. (2016)
`Silh`	Any	Densitybased silhouette plot	Tortora et al. (2013)
`summary`	FPDclustering object	Summary Depends on function used	
`plot`	FPDclustering object	Plots depend on function used	

occasions being set to $K - 1$) on a Cartesian space. More details and an example are available in the following sections.

4 FPDclustering package overview

The `FPDclustering` package contains the functions listed in Table 2.

The following sections are devoted to presenting one example per data type. The results are displayed by using the commands `plot` and `summary`. They recognize the functions' output format and produce a different output depending on the function used, the data type, and the dimension of the data set. In all the following sections, the phrase "*belonging probability*" refers to the belonging degree of a certain unit to each cluster, namely the membership function; the unit is assigned to a cluster in accordance with the maximum degree.

4.1 Visualization

The package also provides a graphical approach to assess the algorithm's overall ability to find well-separated and homogeneous groups. Inspired by the Density-based Silhouette, first proposed by Menardi (2011) as a parametric adaption of the well-known Rousseeuw silhouettes (Rousseeuw 1987), the Density-based Silhouette plot

represents a helpful tool to evaluate *ex-post* the final result. The equation (6) defines the dbs_i for the generic observation $\mathbf{x}_i$

$$dbs_i = \frac{\log\left(\frac{p_{ik}}{p_{il}}\right)}{\max\limits_{i=1,\ldots,n}\left|\log\left(\frac{p_{ik}}{p_{il}}\right)\right|}, \tag{6}$$

where k is such that $\mathbf{x}_i$ belongs to cluster k, i.e. p_{ik} is the biggest probability given i, and l is such that p_{il} is maximum for $l \neq k$ (Tortora et al. 2013). Clusters are ordered in decreasing order according to the dbs index and then plotted on a bar chart. The input function parameter, called `Silh`, is the probability $n \times K$ matrix $\mathbf{P}$, and the output is the average dbs.

The `FPDclustering` package also includes a `plot` function that produces a silhouette plot and a scatter plot with different colors per cluster, as well as a parallel coordinate plot.

5 Applications

In this section, we applied the described functions to three different data sets.

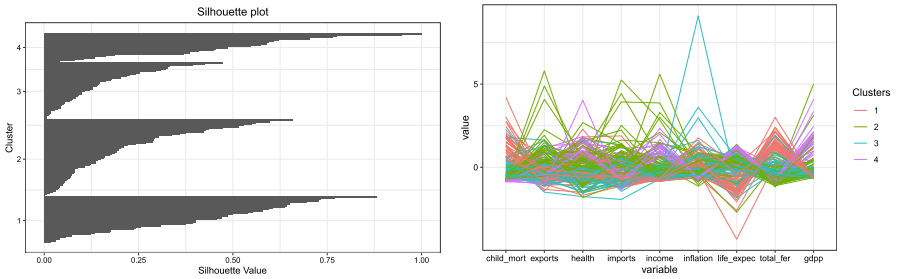
5.1 Continuous data

The data set `Unsupervised Learning on Country Data` contains ten variables recorded on 167 countries (Kokkula 2022). The goal is to categorize the countries using socio-economic and health indicators that determine the country's overall development. The data set has been donated by the HELP International organization, an international humanitarian NGO that needs to identify the countries that need aid and asked the analysts to categorize the countries. The data set does not require data cleaning. Since the units are different per column, the data need to be scaled, and the name of Micronesia needs to be converted to the standard name fsm. We fixed the number of clusters equal to 4 and used the default kmedoids to start the algorithms.

```
library(FPDclustering)
Country_data[102,1]='fsm'
data=scale(Country_data[,2:10])
####results
resPD=PDQ(data,k=4)
resT=TPDC(data,k=4)
resG=GPDC(data,k=4)
```

The results of the three methods implemented in the package for continuous data, PDQ, GPDC, and TPDC are quite different.

To compare the clustering results among the methods, we use the Adjusted Rand Index (ARI) (Sundqvist et al. 2023); the ARI expected value is zero under random classification, and it is equal to one when there is a perfect class agreement. The ARI



(a) Silhouette plot (b) Parallel coordinate plot

Fig. 4 Output of the function plot on PDQ output, silhouette plot and parallel coordinate plot where each color represents a cluster

can be obtained using the `ARI()` function of the `MixGHD` package (Tortora et al. 2019).

```
MixGHD::ARI(resPD$label,resG$label)
[1] 0.1023866
MixGHD::ARI(resPD$label,resT$label)
[1] 0.1320793
MixGHD::ARI(resG$label,resT$label)
[1] 0.5817766
```

Using the function `Silh` we obtain the average silhouette value, which is a measure of the quality of the partition.

```
Silh(resPD$probability)
[1] 0.2706263
Silh(resG$probability)
[1] 0.444655
Silh(resT$probability)
[1] 0.3188897
```

As expected, the ARI shows that PDQ gives different results from GPDC, which is more similar to TPDC. The function `Silh` also shows the silhouette plot; an example using PDQ can be seen in Fig. 4a. In this case, clusters 1 and 4 have some countries with very high belonging probability, and cluster 3 has a lower probability for all the points. The plot also emphasizes some borderline countries with lower belonging probability. This can also be observed in the scatter plot matrix in Fig. 5, where the clusters clearly overlap.

In this example, GPDC gives the best result using the silhouette value as a quality measure. Two functions can help in interpreting the results, `summary` and `plot`, which give the plots in Figs. 4a, b, and 5.

Fig. 5 Output of the function plot on PDQ output, scatter plot where each color represents a different cluster

```
summary(resPD)
 Cluster N. of elements
1       1              37
2       2              61
3       3              46
4       4              23
plot(resPD)
```

The summary shows the number of elements per cluster, and the plot shows the silhouette plot, a parallel coordinate plot, and a scatter plot matrix; in the latter two plots, each cluster corresponds to a different color.

To visualize the differences among the algorithms, Figs. 6, 7, and 8 show the world map colored according to the clustering partition and the corresponding parallel coordinate plot for PDQ, GPDC, and TPDC, respectively. The map is obtained using the package `rworldmap` (South 2011). The following code shows how to create the map using the results from PDQ; the other two maps can be created analogously.

```
# Prepare the data
dFPD=cbind(country=Country_data[,1],label=resPD$label)
sPDFPD <- rworldmap::joinCountryData2Map( dFPD ,joinCode
= "NAME", nameJoinColumn = "country",verbose=T)
# Make the map
mapPD=rworldmap::mapCountryData(sPDFPD,catMethod=
'categorical',
mapTitle ='', aspect='variable', addLegend='FALSE',
nameColumnToPlot='label',
colourPalette=c('brown1','chartreuse4','cyan3',
'darkorchid1'))
```

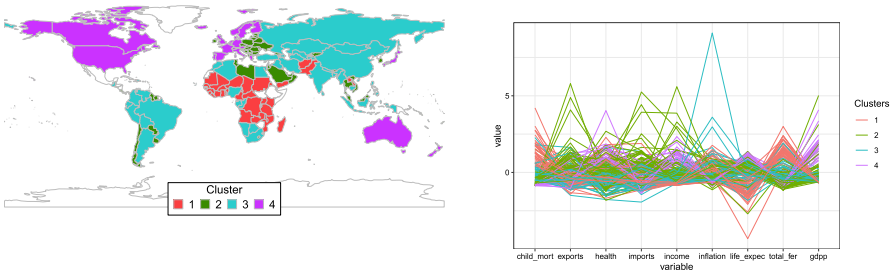


Fig. 6 Map and parallel coordinate plot where each color represents a cluster obtained using PDQ

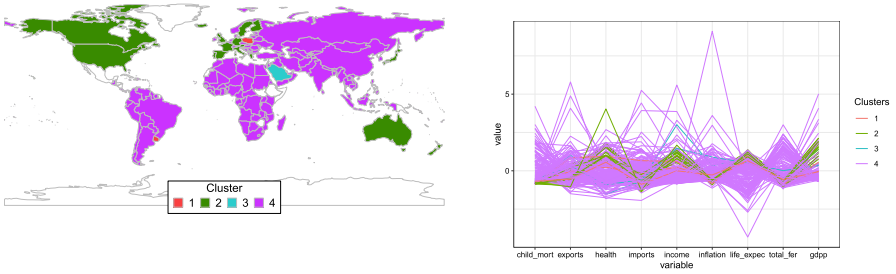


Fig. 7 Map and parallel coordinate plot where each color represents a cluster obtained using GPDC

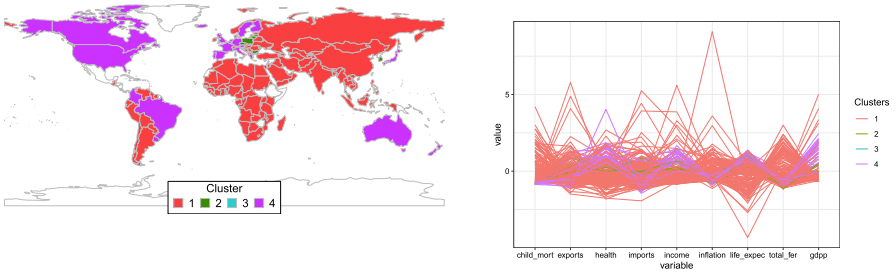


Fig. 8 Map and parallel coordinate plot where each color represents a cluster obtained using TPDC

```
# Add the legend
do.call( rworldmap::addMapLegendBoxes, c(mapPD,x='
bottom',
title="Cluster",horiz=TRUE))
```

PDQ does not account for the clusters' differences in the correlations between variables nor the variability within the clusters and, therefore, partitions the countries into four relatively balanced clusters. Cluster 1 mainly contains African countries with higher child mortality and fertility but lower life expectancy. Cluster 2 mainly contains Eastern European countries, and it is primarily characterized by higher exports and imports but lower fertility. Units in Cluster 3 are mostly from Asia and South America; the cluster exhibits average values in most categories. Cluster 4 contains countries from North America, Western Europe, Australia and Japan, and it is the only cluster

identified by all three different algorithms. On average, the countries in this cluster have higher *per capita* income, imports, and life expectancy. The obtained partitions are different considering the cluster variability and correlation among variables, i.e., using GPDC and TPDC. In both cases, there is a cluster, cluster 3 for GPDC and 4 for TPDC, which contains the countries with higher income, imports, and life expectancy values on average.

A few countries, like Brazil, can be seen as outliers, with only TPDC putting it in the same cluster as North America and Western Europe. The other major difference between those two methods is the smaller clusters. The GPDC algorithm finds that Croatia, Poland, Slovenia, and Uruguay form a cluster with average variables. Bahrain, Kuwait, and Saudi Arabia form another cluster with average values but lower health and higher income. TPDC, instead, finds that Bulgaria, Croatia, Czech Republic, Latvia, Lithuania, Poland, and South Korea form a cluster with overall average values but lower fertility. Barbados and El Salvador are their own cluster.

This example emphasizes how PDQ finds clusters based on a different criterion compared to GPDC and TPDC. The correlations among variables have a big impact on GPDC and TPDC. Therefore, the resulting clusters may have observations with different values per variable in terms of magnitude but a similar correlation structure. PDQ, instead, does not use this information, it is mainly based on each variable's average value. According to the goal, a different method can be a better fit.

More details on the clusters can be obtained by looking at the estimated parameters; here below, we show merely, for example, the first two columns of centers of the clusters obtained using TPDC, i.e. *child_mort* and *exports*.

```
resT$centers[,1:2]
     child_mort       exports
[1,]  0.2786718 -0.102614777
[2,] -0.2937409  0.209532791
[3,] -0.1382427  0.007063895
[4,] -0.5488449 -0.189211879
```

When available, the centers, the sigma values, and degrees of freedom (*df*) can be used for interpretation. It complements what is already shown in the plots, for example, that, when using TPDC, cluster 1, on average, has higher child mortality and lower exports.

5.2 Mixed-type data

To show the usage of PDQ with mixed-type data, we use a data set collected in 2022 that contains ten variables recorded on a convenience sample of 253 students enrolled in the first year at the University of Naples Federico II and attending an introductory Statistics course,[1] Gender (male, female) and prior knowledge of Statistics (yes, no) are binary variables; High School type (lyceum, technical, vocational), course

[1] The authors are indebted with Dr. Rosa Fabbricatore for the data (Fabbricatore 2023). The study was approved by the Ethical Committee of Psychological Research of the University of Naples Federico II (protocol number 26/2022).

modality of attendance[2] (in presence, at home, mixed), and parents' education degree are categorical variables; the others are continuous and refer to five scales assessing: the statistical anxiety (Chiesi et al. 2011, SAS), the mathematical prerequisites for psychometric (Galli et al. 2008, PMP), the relative autonomy index (Alivernini and Lucidi 2008, RAI), the self-efficacy (Bonanomi et al. 2018, S_EFF), and the cognitive competence (Chiesi and Primi 2009, COG). These variables were centered and rescaled to variance equal to one. The package requires specifying the type of data in each column using the position as in the following example.

```
bin=c(1,3)
cat=c(2,4,5)
con=6:10
```

Binary variables are in columns 1 and 3, categorical variables are in columns 2, 4, and 5, and continuous variables are in columns 6 to 10. Binary and categorical variables are defined as factor types, while continuous variables (numerical) must be standardized.

```
data=data.frame(Students)

##Scaling numerical variables
data[,con]=apply(data[,con],2,scale)

##Binary and categorical variables need to be coded as
 factors for(i in 1:5){
data[,i]=factor(data[,i])}
```

We can now apply the `PDQ` function. Continuous variables are standardized, and their variance is 1, the variability of the binary and categorical variables is higher. To avoid over-weighting the non-continuous variables, specific weights can be assigned to each variable using the input parameter `w`. Choosing the most appropriate weights is a matter that interested several authors; however, a definitive answer does not exist (Foss et al. 2019). In this case, the weight of each noncontinuous variable is set to 0.033 based on the variability measured using the normalized Gini heterogeneity index. As the number of clusters is unknown, we explore the results with 2 and 3 clusters and then choose the most appropriate by observing the Silhouette plots in Fig. 9. The distance parameter is set to `gower` since the data are of mixed-type.

```
##Assign weights to account for standardization
weights=c(0.033,0.033,0.033,0.033,0.033,1,1,1,1,1)

res2=PDQ(data,k=2,dist='gower',cat=cat,bin=bin,cont=con,
w=weights)
Silh(res2$probability)
[1] 0.2756792
```

[2] During the first post-COVID-19 academic year (2021/2022), courses were provided in presence and broadcasted through a learning platform to allow remote attendance. Students chose the most convenient modality according to their personal and health conditions.

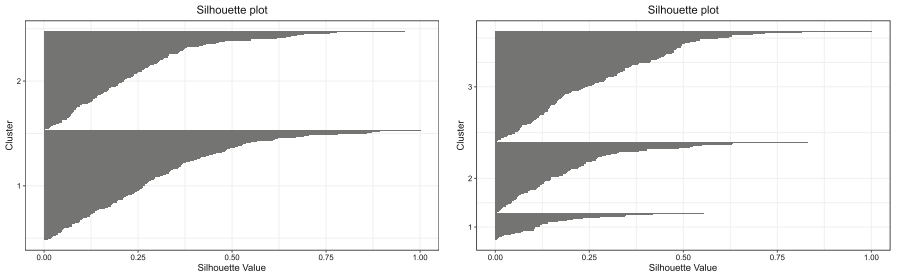


Fig. 9 Silhouette plots for 2 and 3 clusters on the Student data

```
res3=PDQ(data,k=3,dist='gower',cat=cat,bin=bin,cont=con,
w=weights)
Silh(res3$probability)
[1] 0.2321261
```

The average silhouette values are very similar, slightly lower for more clusters, which is expected. Looking at the shape of the plots, the 3 cluster plot has less steep slopes, which means more points with higher silhouette values; we, therefore, choose 3 clusters.

Since no initialization criterion is specified, the default one is used: *i.e.*, `kmedoids` for continuous variables, `kmodes` for categorical variables. The summary function displays the number of elements per cluster.

```
summary(res3)
 Cluster N. of elements
1       1             32
2       2             86
3       3            135
plot(res3)
```

The function plot produces Figs. 10 and 11. Categorical variables can be described using bar plots; see Fig. 12, the following code shows how to obtain the plot for the variable Sex using the package `ggplot2` Wickham (2016). The code can be easily extended to plot the other variables.

```
desc=cbind(data,Cluster=as.factor(res3$label))
ggplot(desc, aes( Sex))+
 geom_bar(aes(fill = Cluster),
    color = "white",position='fill')+ theme_bw()
```

The parallel coordinate plot in Fig. 10 is a powerful tool for interpreting a cluster analysis output with continuous variables. Each line corresponds to a statistical unit: a bundle of the same color lines represents a group. Therefore, looking at the plot, we pinpoint that Cluster 1, the smallest one, contains "average" students for most variables. Cluster 2 includes the students who expressed high values of anxiety for learning Statistics (SAS) and low values of Self-efficacy. The normalized barplots in Fig. 12 show the distributions of the clusters within the binary and nominal variable

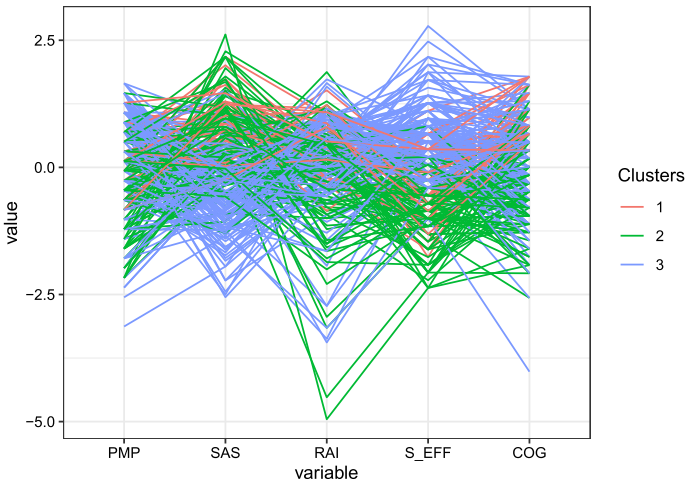


Fig. 10 Parallel coordinate plot of the numerical variables of the Student data set, each color represents a cluster

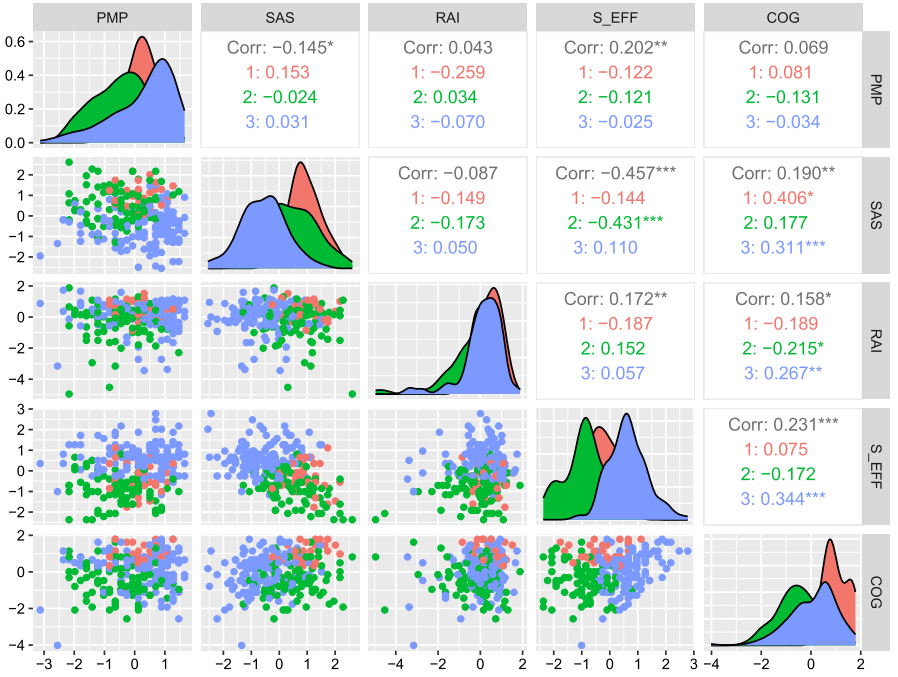


Fig. 11 Scatter plot matrix of the numerical variables of the Student data set, each color represents a cluster

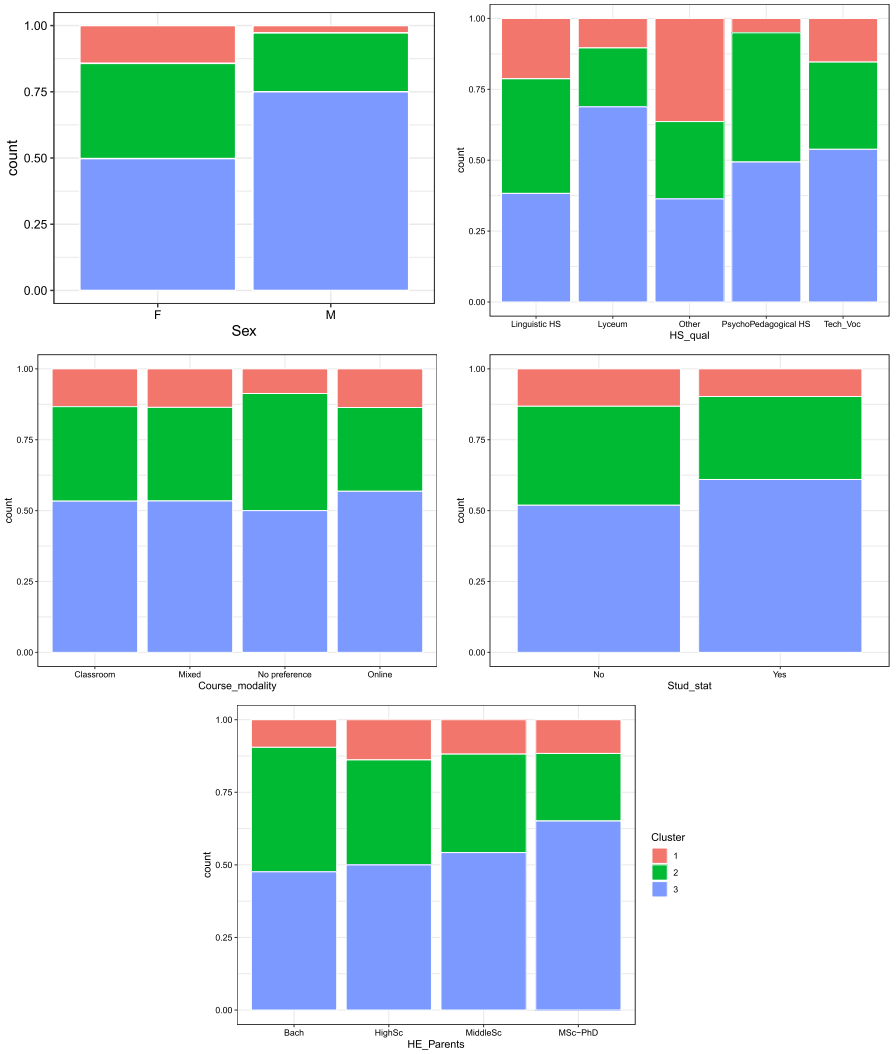


Fig. 12 Bar plots of the categorical variables of the Student data set

categories. It is worth noting that most of the students in Cluster 2 hold a diploma from a psycho-pedagogical, linguistic or technical high school address. The biggest cluster, Cluster 3, includes less anxious students with high self-efficacy. Most students in the cluster hold a Lyceum diploma. This example shows how all the categorical, binary, and continuous variables have contributed to the cluster partition.

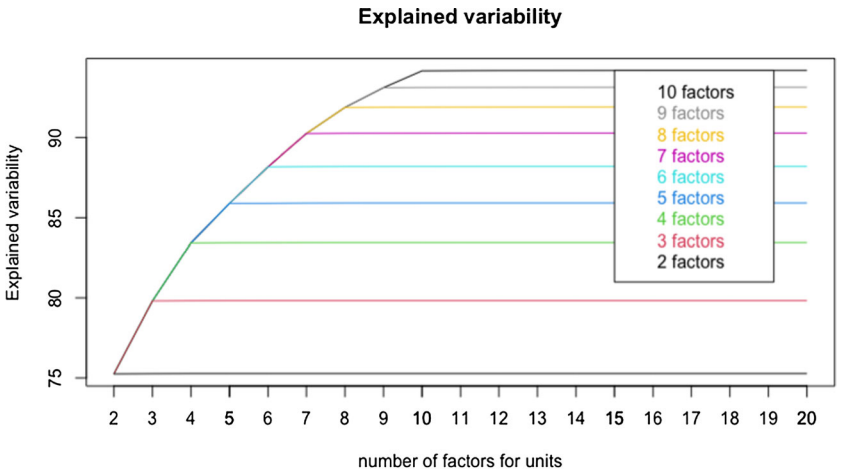


Fig. 13 TuckerFactor output: explained variability per combination of Tucker3 factors of the wdbc data set

5.3 High-dimensional data

In this section, we will use the Wisconsin diagnostic breast cancer (WDBC) data (Mangasarian et al. 1995) available in the R package `mclust` (Scrucca et al. 2016). The data set contains data for 569 patients, and there are 30 variables corresponding to the features of the cell nuclei obtained from a digitized image of a fine needle aspirate (FNA) of a breast mass. For each patient, the cancer was diagnosed as malignant or benign; therefore, we assume there are two clusters. The quite large number of variables recommends reducing the dimensionality. Factor PD clustering exploits the Tucker3 proprieties to reduce the whole problem dimensionality. Tortora et al. (2016) demonstrated the formal consistency between Tucker3 and the PD clustering. Therefore, the function input list of parameters must also include the number of factors to retain. The function `TuckerFactors` can guide the user through the choice. The function produces a table with the explained variability per combination of factors and the plot in Fig. 13.

```
data("wdbc",package='mclust')
library(FPDclustering)
TF=TuckerFactors(scale(wdbc[,3:32]),2)
1 % of the process completed
...
94 % of the process completed
```

The plot suggests four factors for the variables and four factors for the units since, after that, the increase in explained variability is smaller. We can now run the function `FPDC`.

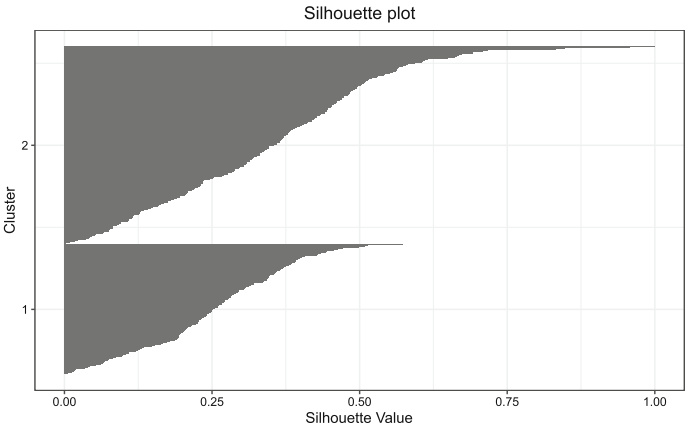


Fig. 14 Silhouette plot obtained clustering wdbc data set using FPDC

```
fpdc_class=FPDC(scale(wdbc[,3:32]),2,4,4)
Iteration number 1
Iteration number 2
Iteration number 3
Iteration number 4

#Results
table(fpdc_class$label,wdbc[,2])
     B   M
 1  31 195
 2 326  17
```

Using the function `table`, we can see that the method identifies malignant and benign tumors with an error rate of 0.0896.

The function gives the belonging probability matrix in output; that can also be visualized using the function `plot(fpdc_class)`, which generates the silhouette plot (Fig. 14) and parallel coordinates plots, not showed for the sake of space.

The silhouette plot in Fig. 14 shows that, while most points are well clustered, some points have a belonging probability close to 0.5. These points are between the clusters and may require further attention. Therefore, instead of using the crisp partition for the entire data set, an alternative option is to assign a crisp partition to the points with belonging probability higher than a certain threshold and let an expert in the field analyze the points between clusters.

To investigate which points have a lower belonging probability, we can look at the matrix of probabilities:

```
> mp=apply(fpdc_class$probability,1,max)
> lp=which(mp<0.6)
> length(fpdc_class$label[-lp])
[1] 336
> table(fpdc_class$label[-lp],wdbc[-lp,2])

     B   M
 1   1 109
 2 223   3
```

In this example, we used a threshold of 0.6; the threshold is subjective and depends on the number of clusters. If we do not assign observations with a belonging probability lower than 0.6 to a cluster, the error rate decreases to 0.0119, with only one malignant tumor classified as benign.

6 Concluding remarks

Data clustering, grouping units into homogeneous groups, is probably one of the most used numerical methods to improve human knowledge from data. The optimal clustering algorithm does not exist and depends on the domain and the specific cluster definition. However, PDC and the PDQ may represent a good alternative to model-based clustering when the number of variables is large, and the distributional assumptions cannot be verified. Gaussian PDC and Student's t PDC ensure more flexibility than the more known geometric approaches. FPDC is computationally feasible even with many variables and provides good stability and efficient estimation of the groups' centers. Mixed-data clustering remains a challenging task, and the PDQ for mixed-type data offers a good answer and an interesting perspective for future work. This article illustrates the `FPDclustering` package that implements the Probability Distance Clustering algorithm and its variants, which together allow the user to face a vast range of clustering problems, including a large number of numerical variables or mixed-type variables. The article briefly summarizes the methodological aspects of the algorithm and refers those more interested in it to the literature; it also offers several examples that illustrate the package's functionalities and capabilities.

Acknowledgements This material is based upon work supported by the National Science Foundation under Grant No. 2209974 (Tortora).

Funding Open access funding provided by Università degli Studi di Napoli Federico II within the CRUI-CARE Agreement.

References

Aggarwal CC (2014) Data classification. Algorithms and applications. CRC Press Taylor and Francis Group, Boca Raton

Ahmad A, Khan SS (2019) Survey of state-of-the-art mixed data clustering algorithms. IEEE Access 7:31883–31902

Alivernini F, Lucidi F (2008) The Academic Motivation Scale (AMS): factorial structure, invariance and validity in the Italian context. Test Psychometr Methodol Appl Psychol 15(4):211–220

Ben-Israel A, Iyigun C (2008) Probabilistic d-clustering. J Classif 25(1):5–26

Bezdek JC (2013) Pattern recognition with fuzzy objective function algorithms. Springer, Berlin

Bonanomi A, Olivari MG, Mascheroni E, Gatti E, Confalonieri E (2018) Using a multidimensional Rash analysis to evaluate the psychometric properties of the motivated strategies for learning questionnaire (MSLQ) among high school students. Test Psychometr Methodol Appl Psychol 25(1):83–100

Chaturvedi A, Green PE, Caroll JD (2001) K-modes clustering. J Classif 18(1):35–55

Chiesi F, Primi C (2009) Assessing statistics attitudes among college students: psychometric properties of the Italian version of the Survey of Attitudes toward Statistics (SATS). Learn Individ Differ 19(2):309–313

Chiesi F, Primi C, Carmona J (2011) Measuring statistics anxiety: cross-country validity of the Statistical Anxiety Scale (SAS). J Psychoeduc Assess 29(6):559–569

De Soete G, Carroll JD (1994) K-means clustering in a low-dimensional Euclidean space. In: Diday E, Lechevallier Y, Schader M et al (eds) New approaches in classification and data analysis. Springer, Berlin, pp 212–219

Fabbricatore R (2023) Latent class analysis for proficiency assessment in higher education: integrating multidimensional latent traits and learning topics. PhD thesis, University of Naples Federico II

Ferraro M, Giordani P, Serafini A (2019) fclust: an r package for fuzzy clustering. The R Journal, 11. https://journal.r-project.org/archive/2019/RJ-2019-017/RJ-2019-017.pdf

Foss AH, Markatou M (2018) KAMILA: clustering mixed-type data in R and Hadoop. J Stat Softw 83:1–44

Foss AH, Markatou M, Ray B (2019) Distance metrics and clustering methods for mixed-type data. Int Stat Rev 87(1):80–109

Galli S, Chiesi F, Primi C (2008) The construction of a scale to measure mathematical ability in psychology students: an application of the Rasch Model. Test Psychometr Methodol Appl Psychol 15(1):1–16

Gordon AD (1999) Classification. CRC Press, Cambridge

Huang Z (1998) Extensions to the K-means algorithm for clustering large data sets with categorical values. Data Min Knowl Disc 2(3):283–304

Ikotun AM, Ezugwu AE, Abualigah L, Abuhaija B, Heming J (2023) K-means clustering algorithms: a comprehensive review, variants analysis, and advances in the era of big data. Inf Sci 622:178–210

Iyigun C, Ben-Israel A (2008) Probabilistic distance clustering adjusted for cluster size. Probab Eng Inf Sci 22(04):603–621

Jain AK (2009) Data clustering: 50 years beyond K-means. Pattern Recogn Lett 31(8):651–666

Kokkula R (2022) Unsupervised learning on country data. kaggle. https://www.kaggle.com/datasets/rohan0301/unsupervised-learning-on-country-data/metadata?resource=download

Kroonenberg PM (2008) Applied multiway data analysis. Ebooks Corporation, Hoboken

Mangasarian OL, Street WN, Wolberg WH (1995) Breast cancer diagnosis and prognosis via linear programming. Oper Res 43(4):570–577

Markos A, Iodice D'Enza A, van de Velden M (2019) Beyond tandem analysis: joint dimension reduction and clustering in R. J Stat Softw 91(10):1–24. https://doi.org/10.18637/jss.v091.i10

Mbuga F, Tortora C (2021) Spectral clustering of mixed-type data. Stats 5(1):1–11

Megahed FM, Jones-Farmer LA (2015) Statistical perspectives on "big data". Springer, Cham, pp 29–47. https://doi.org/10.1007/978-3-319-12355-4_3

Menardi G (2011) Density-based Silhouette diagnostics for clustering methods. Stat Comput 21:295–308. https://doi.org/10.1007/s11222-010-9169-0

Oyewole GJ, Thopil GA (2023) Data clustering: application and trends. Artif Intell Rev 56(7):6439–6475

R Core Team (2016) R: a language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. https://www.R-project.org/. ISBN 3-900051-07-0

Rousseeuw P (1987) Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. J Comput Appl Math 20:53–65

Scrucca L, Fop M, Murphy TB, Raftery AE (2016) `mclust` 5: clustering, classification and density estimation using Gaussian finite mixture models. R J 8(1):289–317. https://doi.org/10.32614/RJ-2016-021

South A (2011) `rworldmap`: a new R package for mapping global data. R J 3(1)

Sundqvist M, Chiquet J, Rigaill G (2023) Adjusting the adjusted Rand index. Comput Stat 38(1):327–347. https://doi.org/10.1007/s00180-022-01230-7

Szepannek G (2018) clustmixtype: user-friendly clustering of mixed-type data in r. R J. https://doi.org/10.32614/RJ-2018-048

Tjur T (2011) Statistics in the computer age: personal reflections. Comput Stat 26(3):371–379

Tortora C (2011) Non-hierarchical clustering methods on factorial subspaces. PhD thesis, Università di Napoli Federico II

Tortora C, Palumbo F (2022) Clustering mixed-type data using a probabilistic distance algorithm. Appl Soft Comput 130:109704

Tortora C, Gettler Summa M, Palumbo F (2013) Factor PD-clustering. In: Berthold UL, Dirk V (eds). Algorithms from and for nature and life, Springer International Publishing, pp 115–123

Tortora C, Gettler Summa M, Marino M, Palumbo F (2016) Factor probabilistic distance clustering (FPDC): a new clustering method for high dimensional data sets. Adv Data Anal Classif 10(4):441–464

Tortora C, El-Sherbiny A, Browne RP, Franczak BC, McNicholas PD (2019) MixGHD: model based clustering and classification using the mixture of generalized hyperbolic distributions. R package version 2.3.2

Tortora C, McNicholas PD, Palumbo F (2020) A probabilistic distance clustering algorithm using Gaussian and Student-t multivariate density distributions. SN Comput Sci 1(2):1–22

Tortora C, Vidales N, Palumbo F, Kalra T, McNicholas PD (2024) FPDclustering. R package version 2.3.1

Van de Velden M, Iodice D'Enza A, Markos A (2019) Distance-based clustering of mixed data. Wiley Interdiscip Rev Comput Stat 11(3):e1456

Van der Laan M, Pollard K, Bryan J (2003) A new partitioning around medoids algorithm. J Stat Comput Simul 73(8):575–584

Vichi M, Kiers HAL (2001) Factorial K-means analysis for two way data. Comput Stat Data Anal 37:29–64

Wickham H (2016) ggplot2: elegant graphics for data analysis. Springer, New York