fclust: An R Package for Fuzzy Clustering

by Maria Brigida Ferraro, Paolo Giordani and Alessio Serafini

Abstract Fuzzy clustering methods discover fuzzy partitions where observations can be softly assigned to more than one cluster. The package **fclust** is a toolbox for fuzzy clustering in the R programming language. It not only implements the widely used fuzzy *k*-means (F*k*M) algorithm, but also many F*k*M variants. Fuzzy cluster similarity measures, cluster validity indices and cluster visualization tools are also offered. In the current version, all the functions are rewritten in the C++ language allowing their application in large-size problems. Moreover, new fuzzy relational clustering algorithms for partitioning qualitative/mixed data are provided together with an improved version of the so-called Gustafson-Kessel algorithm to avoid singularity in the cluster covariance matrices. Finally, it is now possible to automatically select the number of clusters by means of the available fuzzy cluster validity indices.

Introduction

Standard clustering algorithms assign a set of observations to a limited number of clusters such that observations belonging to the same cluster are more similar to each other than to those in other groups. Generally speaking, such algorithms usually produce a *hard* partition of the observations, i. e. every observation is assigned to one and only one cluster. This may often be too strict leading to unfeasible partitions. The well-known Butterfly dataset (Ruspini, 1970) helps to clarify the problem. It is available in the matrix `butterfly` of the package **fclust**, provided that the first and the last rows of the matrix are removed.

```
> data("butterfly", package = "fclust")
> butterfly <- butterfly[-c(1,17),]
> rownames(butterfly) <- as.character(rep(1:nrow(butterfly)))
> plot(butterfly, type = "n", xlab = "Var. 1", ylab="Var. 2")
> text(butterfly[,1], butterfly[,2], labels = rownames(butterfly), cex = 0.7, lwd = 2)
```

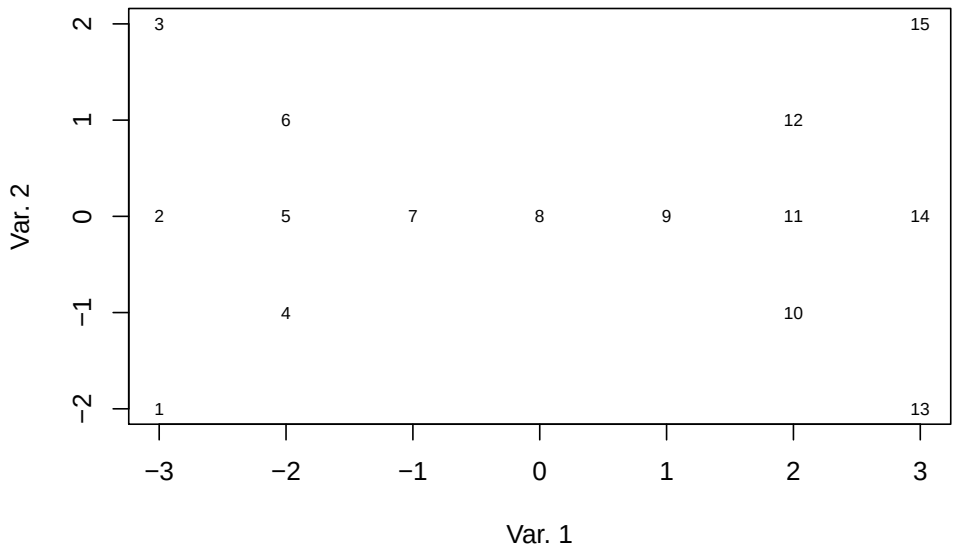

Figure 1: Scatterplot of the Butterfly data.

The Butterfly data refer to 15 observations and 2 variables. Two clusters corresponding to the left and right wings (observations n.1-n.7 and n.9-n.15, respectively) of the butterfly can be graphically depicted without any need of clustering tools. The assignment of observation n.8 (the body of the butterfly) is a much more complex issue because it is at the halfway between the two clusters. Standard algorithms fail to assign properly such an observation. For instance, let us consider the (hard) *k*-means (*k*M) algorithm (Hartigan and Wong, 1979), the most known clustering algorithm. We run the *k*M algorithm (using the function `kmeans`) a large number of times (`nt`).

```
> set.seed(12)
> nt <- 1000
```

```
> ca8 <- rep(NA,nt)
> lfv <- rep(NA,nt)
> for (n in 1: nt){
+   km.butterfly <- kmeans(butterfly, centers = 2, iter.max = 1000, nstart = 10)
+   lfv[n] <- km.butterfly[[5]]
+   if (km.butterfly$cluster[8] == km.butterfly$cluster[1]){
+     ca8[n] <- 1
+   }else{
+     ca8[n] <- 2
+   }
+ }

> summary(lfv)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  31.43   31.43   31.43   31.43   31.43   31.43

> table(ca8)

ca8
  1   2
560 440
```

We find (details not reported) that the same two clusters are always obtained (the one formed by observations n-1-n.7, labelled Cluster 1, and the other one by observations n-9-n.15, labelled Cluster 2), whilst observation n.8 is assigned to one of the two clusters (`ca8`) by chance without affecting the loss function value (`lfv`), i. e. the total within sum of squares.

The difficulty in the assignment of observation n.8 depends on the fact that it shares the features of both the groups. This situation frequently occurs in real life applications. In general, it may exist observations without clear assignments to the clusters. In such cases, it would be desirable to assign them to the clusters to a certain extent. For instance, in the butterfly problem, observation n.8 should be assigned to the two clusters with the same degree. This goal can be achieved by the *fuzzy* approach to clustering where observations belong to clusters with the so-called membership degrees in $[0, 1]$ and, for each observation, the sum of the membership degrees must be equal to 1. Therefore, in fuzzy clustering, the observations are not strictly assigned to one and only one cluster, but can share their memberships to more than one cluster. To differentiate the fuzzy approach from the standard hard one, it may also be referred to as *soft* clustering.

The most known fuzzy clustering algorithm is the fuzzy k-means (FkM), proposed by Bezdek (1981), which is the fuzzy counterpart of kM. It has been implemented in several functions in different R packages: we mention **cluster** (Maechler et al., 2017), **clue** (Hornik, 2005), **e1071** (Meyer et al., 2017), **skmeans** (Hornik et al., 2012), **vegclust** (De Caceres et al., 2010), **ppclust** (Cebeci et al., 2018) and **fclust** (Ferraro and Giordani, 2015). Among them, **fclust** offers a general toolbox for partitioning data using fuzzy clustering algorithms, computing cluster validity indices and visualizing clustering results. The current version (version 2.1.1) of the package has been deeply improved with respect to the previous ones. It contains several new routines and performance improvements. As a first improvement, all the functions have been rewritten in the C++ language using **Rcpp** (Eddelbuettel, 2013) and **RcppArmadillo** (Eddelbuettel and Sanderson, 2014) to enhance their computational efficiency. In addition, all the clustering algorithms can automatically select the optimal number of clusters using the cluster validity indexes available in the package. All the functions usually require data organized by quantitative variables and observations (object data). To increase the usability of the package, another relevant improvement is the implementation of fuzzy clustering algorithms for relational data (Davé and Sen, 2002). Relational data come from measures of dissimilarity between observations. Two clustering methods proposed by Davé and Sen (2002) have been considered. They do not require any assumption about the adopted measure of dissimilarity. Thus, such algorithms can be applied to a wide range of data. Specifically, whilst all the functions for object data require quantitative variables, those for relational data can handle all kinds of data (quantitative, qualitative or mixed) provided that a suitable measure of dissimilarity/distance is selected, e. g. the Gower distance (Gower, 1971) specifying the option `metric="gower"` of the function `daisy` in the package **cluster**. Finally, new functions to compare two partitions in a fuzzy environment have been implemented. Namely, the fuzzy versions of the Rand index (RI; Rand, 1971), the adjusted Rand index (ARI; Hubert and Arabie, 1985), and the Jaccard index (Jaccard, 1901; Halkidi et al., 2001) have been developed by Campello (2007). The standard indexes are implemented in different packages (see, for instance, Scrucca et al., 2016; Chung et al., 2018). The fuzzy vartiants are now available in **fclust**.

In this paper, after reviewing the most relevant fuzzy clustering methods, we recall some of the original functions to present the new improvements and we introduce the new functions by examples. We assume that the latest version of **fclust** available on CRAN is already installed with

```
> install.packages("fclust")
```

and loaded into the R session using

```
> library(fclust)
```

Fuzzy clustering

In this section, we recall the fuzzy k-means algorithm (Bezdek, 1981) and its extension suggested by Gustafson and Kessel (1979). Whilst the former detects spherical clusters, the latter allows for clusters with ellipsoidal shape. Then, a fuzzy clustering algorithm for relational data is described (Davé and Sen, 2002)

Fuzzy k-means algorithm

The most known and used fuzzy clustering algorithm is the fuzzy k-means (FkM) (Bezdek, 1981). The FkM algorithm aims at discovering the best fuzzy partition of n observations into k clusters by solving the following minimization problem:

$$\begin{aligned} &\min_{\mathbf{U},\mathbf{H}} J_{\text{F}k\text{M}} = \sum_{i=1}^{n}\sum_{g=1}^{k} u_{ig}^{m} d^2\left(\mathbf{x}_i, \mathbf{h}_g\right), \\ &\text{s.t.} \quad u_{ig} \in [0,1]\,, \sum_{g=1}^{k} u_{ig} = 1, \end{aligned} \tag{1}$$

where $d\left(\mathbf{x}_i, \mathbf{h}_g\right)$ is the Euclidean distance. In (1), u_{ig} is the generic element of the membership degree matrix $\mathbf{U}$ of order $(n \times k)$, taking values in the interval [0,1] and representing the membership degree of observation i to cluster g. The row-wise sums of $\mathbf{U}$ are equal to 1. The propotypes (centroids) $\mathbf{h}_g = [h_{g1}, h_{g2}, \cdots, h_{gp}]$ $(g = 1, \cdots, k)$ are stored in the matrix $\mathbf{H}$ of order $(k \times p)$, being p the number of variables. Finally, the parameter m tunes the fuzziness of the obtained solution.

Gustafson-Kessel extensions of the FkM algorithm

The FkM algorithm, as the standard k-Means, can determine only spherical shaped clusters. This depends on the use of the Euclidean distance to measure the dissimilarity between observations. This limits its applicability when non-spherical clusters are expected. In order to overcome this drawback, Gustafson and Kessel (1979) extend the FkM algorithm by replacing the Euclidean distance with a cluster-specific Mahalanobis distance:

$$d_M^2\left(\mathbf{x}_i, \mathbf{h}_g\right) = \left(\mathbf{x}_i - \mathbf{h}_g\right)' \mathbf{M}_g \left(\mathbf{x}_i - \mathbf{h}_g\right), \tag{2}$$

where $\mathbf{M}_g$ is a symmetric and positive-definite matrix of order p. When $\mathbf{M}_g = \mathbf{I}$, (2) is equvalent to the Euclidean distance. The Gustafson-Kessel FkM (briefly, GK-FkM) consists in solving the following minimizatin problem

$$\begin{aligned} &\min_{\mathbf{U},\mathbf{H},\mathbf{M}_1,\ldots,\mathbf{M}_k} J_{\text{GK-F}k\text{M}} = \sum_{i=1}^{n}\sum_{g=1}^{k} u_{ig}^{m} d_M^2\left(\mathbf{x}_i, \mathbf{h}_g\right), \\ &\text{s.t.} \quad u_{ig} \in [0,1]\,, \sum_{g=1}^{k} u_{ig} = 1, |\mathbf{M}_g| = \rho_g. \end{aligned} \tag{3}$$

The constraints in (3) are similar to those in (1) except for the new ones on $\mathbf{M}_g$ ($|\mathbf{M}_g| = \rho_g > 0$, with ρ_g fixed for each g), added to avoid a trivial solution with $\mathbf{M}_g = \mathbf{0}$, that would be obtained since $J_{\text{GK-F}k\text{M}}$ is linear in $\mathbf{M}_g$.

For the generic g-th cluster, the iterative solution of $\mathbf{M}_g$ is $[\rho_g|\mathbf{V}_g|]^{\frac{1}{n}}\mathbf{V}_g^{-1}$, where $\mathbf{V}_g$ is the fuzzy covariance matrix for cluster g, defined as

$$\mathbf{V}_g = \frac{\sum_{i=1}^{n} u_{ig}^{m}(\mathbf{x}_i - \mathbf{h}_g)(\mathbf{x}_i - \mathbf{h}_g)'}{\sum_{i=1}^{n} u_{ig}^{m}}, \quad g = 1, \ldots, k. \tag{4}$$

The eigenvalues and eigenvectors of $\mathbf{V}_g$ describe the shape and orientation of the g-th cluster. When an eigenvalue is equal to 0 or when the condition number of $\mathbf{V}_g$ (i. e. the ratio between its maximum and minimum eigenvalue) is very large, the matrix is nearly singular, hence $\mathbf{V}_g^{-1}$ cannot be calculated. The condition $|\mathbf{V}_g| = \rho_g$ cannot overcome this drawback, as the determinant becomes 0. Babuska et al. (2002) propose to avoid these numerical problems by constraining the condition number of $\mathbf{V}_g$ to be smaller than a predefined threshold. Since this might lead to overfit the data, the update of $\mathbf{V}_g$ can be regularized by considering the covariance matrix of the whole dataset. See, for more details, Babuska et al. (2002).

Fuzzy clustering algorithms for relational data

In practical applications it may occur that only relational data are available. Relational data consist in the pairwise relations (similarities or dissimilarities) between observations, stored in a square matrix, say $\mathbf{D}$, not necessarily based on object data. In fact, $\mathbf{D}$ can be built either by computing the dissimilarity (or similarity) between all the pairs of observations on which a set of variables are collected (indirect relational data) or according to subjective expert knowledge, e. g. a teacher expresses her/his subjective degrees of dissimilarity for all the pair of pupils in her/his classroom (direct relational data). In the latter case, fuzzy clustering algorithms for object data can no longer be applied. In the former case, fuzzy clustering algorithms for object data should be preferred to those for relational data due to their computational efficiency. Nonetheless, fuzzy clustering algorithms usually assume to deal with quantitative variables preventing their applicability in case of qualitative/mixed variables. In such a case, fuzzy clustering methods for relational data can be fruitfully applied provided that a suitable dissimilarity measure for qualitative/mixed variables is used.

In the literature, there exist several proposals of fuzzy clustering algorithms for relational data. Among them, a valuable choice is represented by the class of algorithms proposed by Davé and Sen (2002), which are suitable for all kinds of dissimilarity. We assume that $\mathbf{D}$ is a dissimilarity matrix. If it contains similarities, these should be converted into dissimilarities. For this purpose, e. g. the function `sim2diss` of the package **smacof** (de Leeuw and Mair, 2009) can be used. The non-Euclidean fuzzy relational data clustering algorithm (NEFRC) consists in solving the following minimization problem:

$$\min_{\mathbf{U}} J_{\text{NEFRC}} = \sum_{g=1}^{k} \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} u_{ig}^m u_{jg}^m d(\mathbf{x}_i,\mathbf{x}_j)}{2\sum_{t=1}^{n} u_{tg}^m},$$
$$\text{s.t.} \quad u_{ig} \in [0,1], \sum_{g=1}^{k} u_{ig} = 1.$$

Notice that the NEFRC algorithm differs from the famous FANNY algorithm proposed by Kaufman and Rousseeuw (1990) since a general fuzzifier m is used and it is suitable for all kinds of dissimilarity.

The package also offers a robust variant of NEFRC involving the concept of noise cluster. It is an additional cluster such that the outliers are assigned to it with high membership degrees. It is not a cluster in a strict sense because the outliers are not necessarily similar to each other. Its role is that the membership degrees of the outliers to the standard clusters tend to be low without affecting the obtained partition. The robust version of NEFRC has been implemented in the current version of **fclust** and represents the relational counterpart of the F*k*M algorithm with noise cluster, already available in the package.

The package

In this section we present the main features of the package **fclust** with particular emphasis on the more recent updates. The list of algorithms with the corresponding functions is reported in Table 1. Apart from some peculiarities, all the available functions in the package require the same input arguments, involving the set-up of the clustering algorithms, i. e. number of starts, convergence criterion, data standardization. The user is not requested to specify such arguments because default options are specified. Obviously, the dataset to be clustered must be given.

Differently from the previous versions, the number of groups `k` is no longer required. Of course, the user can select the integer value of `k`, otherwise the optimal number of clusters is automatically chosen by maximizing or minimizing one of the available fuzzy cluster validity indices (see Table 2) to be specified in the option `index` (default `"SIL.F"`). By default the possible number of clusters is in the vector `k=2:6`, unless a different integer vector is selected by the user.

A fast way to apply one of the available algorithms is represented by the function `Fclust`:

```
> Fclust (X, k, type, ent, noise, stand, distance)
```

Function	Algorithm
`FKM`	standard F*k*M algorithm (Bezdek, 1981)
`FKM.ent`	F*k*M with entropy regularization (Li and Mukaidono, 1995, 1999)
`FKM.noise`	F*k*M with noise cluster (Davé, 1991)
`FKM.ent.noise`	F*k*M with entropy regularization and noise cluster (Li and Mukaidono, 1999; Davé, 1991)
`FKM.gk`	Gustafson and Kessel extension of F*k*M (Gustafson and Kessel, 1979)
`FKM.gk.ent`	Gustafson and Kessel extension of F*k*M with entropy regularization (Ferraro and Giordani, 2013)
`FKM.gk.noise`	Gustafson and Kessel extension of F*k*M with noise cluster (Gustafson and Kessel, 1979; Davé, 1991)
`FKM.gk.ent.noise`	Gustafson and Kessel extension of F*k*M with entropy regularization and noise cluster (Ferraro and Giordani, 2013; Davé, 1991)
`FKM.gkb`	Gustafson, Kessel and Babuska extension of F*k*M (Babuska et al., 2002; Gustafson and Kessel, 1979)
`FKM.gkb.ent`	Gustafson, Kessel and Babuska extension of F*k*M with entropy regularization (Babuska et al., 2002; Ferraro and Giordani, 2013)
`FKM.gkb.noise`	Gustafson, Kessel and Babuska extension of F*k*M with noise cluster (Babuska et al., 2002; Davé, 1991)
`FKM.gkb.ent.noise`	Gustafson, Kessel and Babuska extension of F*k*M with entropy regularization and noise cluster (Babuska et al., 2002; Ferraro and Giordani, 2013; Davé, 1991)
`FKM.pf`	F*k*M with polynomial fuzzifier (Winkler et al., 2009, 2011)
`FKM.pf.noise`	F*k*M with polynomial fuzzifier and noise cluster (Winkler et al., 2009, 2011; Davé, 1991)
`FKM.med`	fuzzy *k*-medoids algorithm (Krishnapuram et al., 2001)
`FKM.med.noise`	fuzzy *k*-medoids algorithm with noise cluster (Krishnapuram et al., 2001; Davé, 1991)
`NEFRC`	non-euclidean fuzzy relational algorithm (Davé and Sen, 2002)
`NEFRC.noise`	non-euclidean fuzzy relational algorithm with noise cluster (Davé and Sen, 2002; Davé, 1991)

Table 1: List of fuzzy clustering algorithms available in the package **fclust**.

In `Fclust` to choose a given algorithm, the options `type`, `ent`, `noise` and `distance` should be set. `type` is a character string specifying the type of algorithm to be used. The currently available options are `"standard"` (the default option for FKM-type algorithms, provided that `distance = FALSE`), `"polynomial"`, `"gk"`, `"gkb"`, `"medoids"`. `ent` (default `FALSE`) and `noise` (default `FALSE`) are logical values indicating, respectively, whether the entropy regularization and the noise cluster should be used. Morever, `distance` (default `FALSE`) is another logical value indicating whether the data in `X` are distances/dissimilarities. When `distance = TRUE`, `type` is constrained to be `"standard"` and NEFRC-type algorithms are run. Finally, `stand` is used for standardization (default: no standardization) and `k` indicates the desired number of clusters (only for this function, the default value is 2). For instance, the researcher interested in applying the F*k*M algorithm with noise cluster with $k = 3$ clusters to `X` can digit:

```
> Fclust (X = X, k = 3, type = "standard", noise = TRUE)
```

Function	Index
`PC`	partition coefficient
`MPC`	modified partition coefficient
`PE`	partition entropy
`XB`	partition entropy
`SIL`	(crisp) silhouette
`SIL.F`	fuzzy silhouette

Table 2: List of fuzzy cluster validity indices available in the package **fclust**.

In the following we are going to present the main features and improvements of the package by considering the standard F*k*M algorithm (function `FKM`), the Gustafson–Kessel extension of F*k*M according to the Babuska et al. (2002) variant (function `FKM.gkb`), and the clustering algorithm for relational data (function `NEFRC`).

Fuzzy *k*-means (`FKM`)

The `FKM` function is applied to the NBA dataset available in **fclust**. The dataset contains some statistics on 30 NBA teams for the regular season 2017-2018 (source: `https://stats.nba.com/teams/traditional/`): number of wins (`W`), field goals made (`FGM`), field goals attempted (`FGA`), field goal percentage (`FGP`), 3 point field goals made (`3PM`), 3 point field goals attempted (`3PA`), 3 point field goals percentage (`3PP`), free throws made (`FTM`), free throws attempted (`FTA`), free throw percentage (`FTP`), offensive rebounds (`OREB`), defensive rebounds (`DREB`), assists (`AST`), turnovers (`TOV`), steals (`STL`), blocks (`BLK`), blocked field goal attempts (`BLKA`), personal fouls (`PF`), personal fouls drawn (`PFD`) and points (`PTS`). In addition, two more variables are available indicating the conference (`Conference`) and the playoff appearance (`playoff`). Both the variables are objects of class `factor` with two levels.

The dataset can be loaded as following:

```
> data("NBA")
```

A subset of variables is considered for clustering purposes. The raw values of field goals, point field goals and free throws are removed (only the percentage values are considered), as well as the wins and the personal fouls.

```
> X.NBA <- NBA[,c(4,7,10,11,12,13,14,15,16,17,20)]
```

We apply the function `FKM` to the obtained dataset. The parameter of fuzziness `m` is set to `m = 1.2` (the default value `m = 2` was too high producing an extremely fuzzy partition with membership degrees not far from 0.5) and the number of starts is fixed to 50 (`RS = 50`) to avoid local optima. The number of clusters is automatically selected using the fuzzy silhouette index (`index = "SIL.F"`). Notice that the fuzzy silhouette index represents a fuzzy extension of the well-known silhouette (Kaufman and Rousseeuw, 1990) involving the use of the membership degree information (for further details, refer to Campello, 2007). Finally, we set `stand = 1` in order to standardize the data before running `FKM`:

```
> fkm.NBA <-  FKM(X = X.NBA, m = 1.2, RS = 50, stand = 1, index = "SIL.F")
```

The `summary` method returns the most relevant information:

```
> summary(fkm.NBA)


 Fuzzy clustering object of class 'fclust'

 Number of objects:
 30

 Number of clusters:
 2

 Cluster sizes:
Clus 1 Clus 2
    18     12


 Clustering index values:
SIL.F k=2 SIL.F k=3 SIL.F k=4 SIL.F k=5 SIL.F k=6
0.2994904 0.2508281 0.2558217 0.2586680 0.2700120


 Closest hard clustering partition:
       Houston Rockets        Toronto Raptors  Golden State Warriors
                     2                      2                      2
        Boston Celtics     Philadelphia 76ers    Cleveland Cavaliers
                     1                      2                      2
Portland Trail Blazers         Indiana Pacers   New Orleans Pelicans
                     1                      2                      2
 Oklahoma City Thunder              Utah Jazz Minnesota Timberwolves
                     1                      2                      2
     San Antonio Spurs         Denver Nuggets              Miami Heat
                     1                      2                      1
       Milwaukee Bucks     Washington Wizards            LA Clippers
                     2                      2                      1
       Detroit Pistons      Charlotte Hornets     Los Angeles Lakers
                     1                      1                      1
       New York Knicks          Brooklyn Nets          Chicago Bulls
                     1                      1                      1
      Sacramento Kings          Orlando Magic          Atlanta Hawks
                     1                      1                      1
      Dallas Mavericks      Memphis Grizzlies           Phoenix Suns
                     1                      1                      1

 Cluster memberships:
  Clus 1
  [1] "Boston Celtics"         "Portland Trail Blazers"
 [3] "Oklahoma City Thunder"  "San Antonio Spurs"
 [5] "Miami Heat"             "LA Clippers"
 [7] "Detroit Pistons"        "Charlotte Hornets"
 [9] "Los Angeles Lakers"     "New York Knicks"
[11] "Brooklyn Nets"          "Chicago Bulls"
[13] "Sacramento Kings"       "Orlando Magic"
[15] "Atlanta Hawks"          "Dallas Mavericks"
[17] "Memphis Grizzlies"      "Phoenix Suns"
 Clus 2
  [1] "Houston Rockets"        "Toronto Raptors"
 [3] "Golden State Warriors"  "Philadelphia 76ers"
 [5] "Cleveland Cavaliers"    "Indiana Pacers"
 [7] "New Orleans Pelicans"   "Utah Jazz"
 [9] "Minnesota Timberwolves" "Denver Nuggets"
[11] "Milwaukee Bucks"        "Washington Wizards"

 Number of objects with unclear assignment (maximal membership degree <0.5):
 0
```

```
 Membership degree matrix (rounded):
                       Clus 1 Clus 2
Houston Rockets          0.02   0.98
Toronto Raptors          0.01   0.99
Golden State Warriors    0.02   0.98
Boston Celtics           0.92   0.08
Philadelphia 76ers       0.11   0.89
Cleveland Cavaliers      0.05   0.95
Portland Trail Blazers   0.95   0.05
Indiana Pacers           0.34   0.66
New Orleans Pelicans     0.00   1.00
Oklahoma City Thunder    0.78   0.22
Utah Jazz                0.14   0.86
Minnesota Timberwolves   0.12   0.88
San Antonio Spurs        0.77   0.23
Denver Nuggets           0.03   0.97
Miami Heat               1.00   0.00
Milwaukee Bucks          0.03   0.97
Washington Wizards       0.03   0.97
LA Clippers              0.96   0.04
Detroit Pistons          1.00   0.00
Charlotte Hornets        0.98   0.02
Los Angeles Lakers       0.93   0.07
New York Knicks          0.96   0.04
Brooklyn Nets            0.99   0.01
Chicago Bulls            1.00   0.00
Sacramento Kings         0.98   0.02
Orlando Magic            1.00   0.00
Atlanta Hawks            0.98   0.02
Dallas Mavericks         0.97   0.03
Memphis Grizzlies        0.99   0.01
Phoenix Suns             0.99   0.01

 Cluster summary:
       Cl.size Min.memb.deg. Max.memb.deg. Av.memb.deg. N.uncl.assignm.
Clus 1      18          0.77             1         0.95               0
Clus 2      12          0.66             1         0.92               0

 Euclidean distance matrix for the prototypes (rounded):
       Clus 1
Clus 2   2.91

 Available components:
 [1] "U"         "H"         "F"         "clus"      "medoid"
 [6] "value"     "criterion" "iter"      "k"         "m"
[11] "ent"       "b"         "vp"        "delta"     "stand"
[16] "Xca"       "X"         "D"         "call"
```

According to `SIL.F`, we select the solution with $k = 2$ clusters. The obtained clusters can be plotted on the plane spanned by the first two principal components. This can be done by using the method `plot` associated to an **fclust** object specifying the option `pca = TRUE`.

We can see that the first component is able to distinguish the two clusters. Teams with high first component scores belong to Cluster 2 and those with low scores to Cluster 1. The first component loadings are (the script is omitted):

```
   FGP    3PP    FTP   OREB   DREB    AST    TOV    STL    BLK   BLKA
 0.455  0.305  0.278 -0.157  0.158  0.395  0.071  0.160  0.370 -0.338
   PTS
 0.369
```

Hence, it appears that the clusters are mainly distinguish in terms of `FGP`, `AST`, `BLK`, `PTS`, `BLKA`, `3PP` and `FTP`, i. e. the variables with the highest first component loadings in absolute value.

```
> plot(fkm.NBA, pca = TRUE)
```

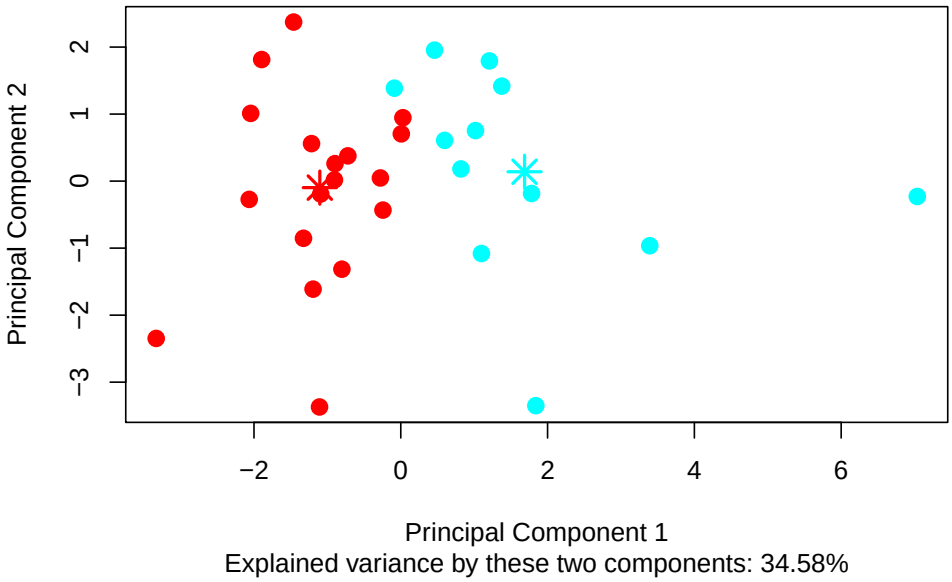

Figure 2: Scatterplot of the NBA teams on the plane spanned by the first two principal components. Points are marked according to the obtained partition (Cluster 1: red, Cluster 2: cyan).

In order to interpret the clusters, we inspect the prototypes. To this purpose, we apply the function `Hraw` to visualize the prototypes by using the original units of measurement.

```
> round(Hraw(X = X.NBA, H = fkm.NBA$H), 3)

         FGP   3PP   FTP  OREB   DREB    AST    TOV   STL   BLK  BLKA
Clus 1 0.451 0.358 0.759 9.830 33.757 22.483 14.252 7.446 4.581 4.996
Clus 2 0.474 0.367 0.780 9.513 33.891 24.425 14.305 8.096 5.145 4.548
          PTS
Clus 1 104.362
Clus 2 109.357
```

We can see that Cluster 2 recognizes the best teams. In fact, the values of the prototype of Cluster 2 are better than the corresponding ones of Cluster 1, except for a few variables such as `OREB` and `BLKA`. To further characterize the obtained clusters, we consider the variables `Conference` and `Playoff`. In particular, we aim at discovering whether the two clusters can be interpreted in terms of the geographical location and/or the playoff appearance. From a statistical point of view, this consists in comparing the fuzzy partition resulting from `FKM` with the hard ones corresponding to the classification variables `Conference` or `Playoff`. For this purpose, the fuzzy cluster similarity measures available in the package are considered. Such measures, proposed by Campello (2007), are summarized in Table 3.

Function	Index
`RI.F`	Fuzzy version of Rand index
`ARI.F`	Fuzzy version of adjusted Rand index
`JACCARD.F`	Fuzzy version of Jaccard index

Table 3: List of fuzzy cluster similarity measures available in the package **fclust**.

To report the values of the three measures, the function `Fclust.compare` can be used. The input required by `Fclust.compare` (and similarly for `RI.F`, `ARI.F` and `JACCARD.F`) is a fuzzy membership degree matrix (`U`) and a vector of class labels (`VC`).

```
> Fclust.compare(VC = NBA$Playoff, U = fkm.NBA$U)

    ARI.F      RI.F JACCARD.F
0.3077549 0.6537339 0.4825140
```

```
> Fclust.compare(VC = NBA$Conference, U = fkm.NBA$U)

      ARI.F        RI.F   JACCARD.F
-0.02547957  0.48701485  0.31724090
```

It is clear that the clusters cannot be interpreted from a geographical point of view, whilst, to some extent, they are related to the playoff appearance. Such a comment holds especially for Cluster 2. In fact, 11 out of 12 teams belonging to Cluster 2 reached the playoff stage. The only exception is `Denver Nuggets`, which was one of the best teams in terms of number of wins (`W`) but not qualified to the playoff stage, because the number of wins was not sufficient to reach the playoff stage in the Western conference.

Gustafson-Kessel extensions of the F*K*M algorithm (`FKM.gk` and `FKM.gkb`)

The Gustafson-Kessel extension of the F*k*M algorithm is implemented in the functions `FKM.gk` and `FKM.gkb`. The former implements the GK-F*k*M algorithm in the original proposal, whilst the latter, recently added to the package, considers the computational improvement suggested by Babuska et al. (2002). A simulated dataset similar to the one in Babuska et al. (2002) is used to show the differences between the two functions. Three different clusters with different size (100, 80, and 60) in two-dimensional space are generated as follows:

$$y = \begin{cases} 6 - 2.0x & \text{with } x \sim U(1,3) \text{ for Cluster 1,} \\ -5 + 1.5x & \text{with } x \sim U(3.2,6) \text{ for Cluster 2,} \\ 3x & \text{with } x \sim U(-1,1) \text{ for Cluster 3.} \end{cases} \tag{5}$$

Data can be found in **fclust** and loaded with the following command:

```
> data(synt.data2)
```

Figure 3 shows the scatterplot of the simulated data. In this case the cluster covariance matrices are singular because the two variables are perfectly collinear in all the clusters.

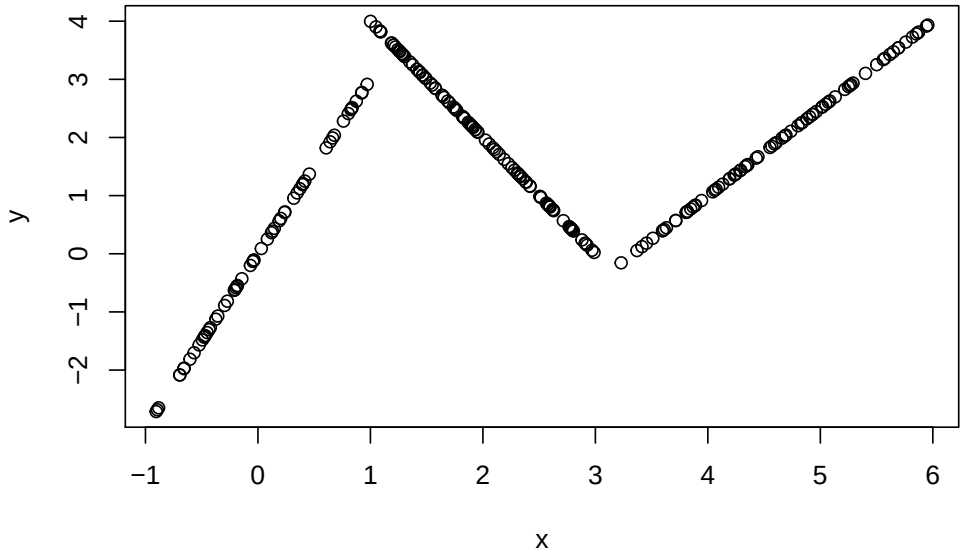

Figure 3: Scatterplot of the `synt.data2` dataset.

By employing the standard function `FKM.gk` numerical problems occur. By setting `m = 2`, `k = 3` and `RS = 1`, we have:

```
> fkm.gk.synt <- FKM.gk(X = synt.data2, k = 3, RS = 1)
```

The following `warning` message appears:

```
Warning message:
In FKM.gk(X = synt.data2, k = 3, RS = 1) :
  When k=3, at least one cluster covariance matrix seems to be singular.
  Increase the number of starts RS or use FKM.gkb
```

Thus, we can see that the algorithm stops because at least one cluster covariance matrix is singular. In this case, the function returns the standard object of class `fclust` containing the sub-optimal solution at the previous iteration, i. e. the one with no singular cluster covariance matrices. By studying the number of iterations and the loss function value of such a local optimum solution, we get:

```
> fkm.gk.synt$iter

Start 1
     13


> fkm.gk.synt$value

   Start 1
0.06044555
```

For comparative purpose, we run the recommended function `FKM.gkb` using the same start:

```
> fkm.gkb.synt <- FKM.gkb(X = synt.data2, k = 3, RS = 1, seed = 123)


> fkm.gkb.synt$iter

Start 1
     16


> fkm.gkb.synt$value

     Start 1
1.482029e-05
```

The method required two more iterations for convergence. The obtained solution is characterized by a lower loss function value and is not affected by singularity problems.

Fuzzy clustering for indirect relational data (dichotomous variables)

The NEFRC algorithm can be applied using the function `NEFRC`. Differently from the other functions for clustering object data, it requires distances/dissimilarities as input argument. Consistently with the other functions, the available clustering indices (except for the Xie and Beni one) can be used to select the optimal number of clusters `k`. In particular, the silhouette index (`SIL`) and its fuzzy extension (`SIL.F`) have been rearranged for relational data. Specifically, the input of `NEFRC` is employed to compute the silhouette and the fuzzy silhouette indices. This is the default option when `SIL.F` is called by `NEFRC`. In order to use the distance/dissimilarity matrix for computing the fuzzy silhouette index, the option `distance = TRUE` in `SIL.F` should be set. Generally speaking, the fuzzy silhouette index can be applied for any kind of data (quantitative or qualitative or mixed) provided that a suitable distance/dissimilarity matrix is used as input.

The function `NEFRC` is presented by considering the congressional voting records data (Schlimmer, 1987) available in **fclust**. The data collect 1984 United Stated voting records for 475 U.S. House of Representative congressmen on 16 key votes identified by the Congressional Quartely Almanac (CQA). The congressmen are split into Democrats and Republicans (variable `class`). The 16 key votes are objects of class `factor` with three levels according to the CQA scheme: `y` refers to the types of votes "voted for", "paired for" and "announced for"; `n` to "voted against", "paired against" and "announced against"; `yn` to "voted present", "voted present to avoid conflict of interest" and "did not vote or otherwise make a position known".

The dataset can be loaded as follows:

```
> data("houseVotes")
```

It contains the following variables:

```
> colnames(houseVotes)

 [1] "class"                                 "handicapped-infants"
 [3] "water-project-cost-sharing"            "adoption-of-the-budget-resolution"
 [5] "physician-fee-freeze"                  "el-salvador-aid"
 [7] "religious-groups-in-schools"           "anti-satellite-test-ban"
 [9] "aid-to-nicaraguan-contras"             "mx-missile"
```

```
[11] "immigration"                              "synfuels-corporation-cutback"
[13] "education-spending"                       "superfund-right-to-sue"
[15] "crime"                                    "duty-free-exports"
[17] "export-administration-act-south-africa"
```

Since the level yn might indicate unknown preferences for some key votes, these values are considered as missing and, therefore, the rows with at least one yn value are removed:

```
> level.drop <- droplevels(houseVotes, exclude = "yn")
> houseVotesComplete <- level.drop[complete.cases(level.drop),]
```

The research interest relies in discovering whether a two-cluster structure exists and a relationship between the political position and the system of voting emerges. Even if the dataset is not relational, NEFRC is the only one R routine for getting a fuzzy partition based on qualitative variables. For this purpose, the Gower distance, implemented in the function daisy of the package **cluster**, is used to generate the dissimilarity matrix:

```
> X.houseVotesComplete <- houseVotesComplete[,-1]
> library(cluster)
> D.houseVotes <- daisy(x = X.houseVotesComplete, metric = "gower")
```

The standard algorithm for relational data is employed by running the function NEFRC setting m = 1.5 and k = 2 in order to assess whether the clusters are related to the parties (class).

```
> nefrc.houseVotes <- NEFRC(D = D.houseVotes, k  = 2, m = 1.5, index = "SIL.F")
```

The summary method is similar to that of FKM and hence not reported. The two clusters can be interpreted in terms of the parties. In fact, we get the following cluster similarity measures:

```
> Fclust.compare(VC = houseVotesComplete$class, U = nefrc.houseVotes$U)

    ARI.F      RI.F JACCARD.F
0.4871095 0.7435544 0.5914710
```

Morevover, we have:

```
> table(nefrc.houseVotes$clus[,1], houseVotesComplete$class)


    democrat republican
  1       19        101
  2      105          7
```

```
> plot(nefrc.houseVotes)
```

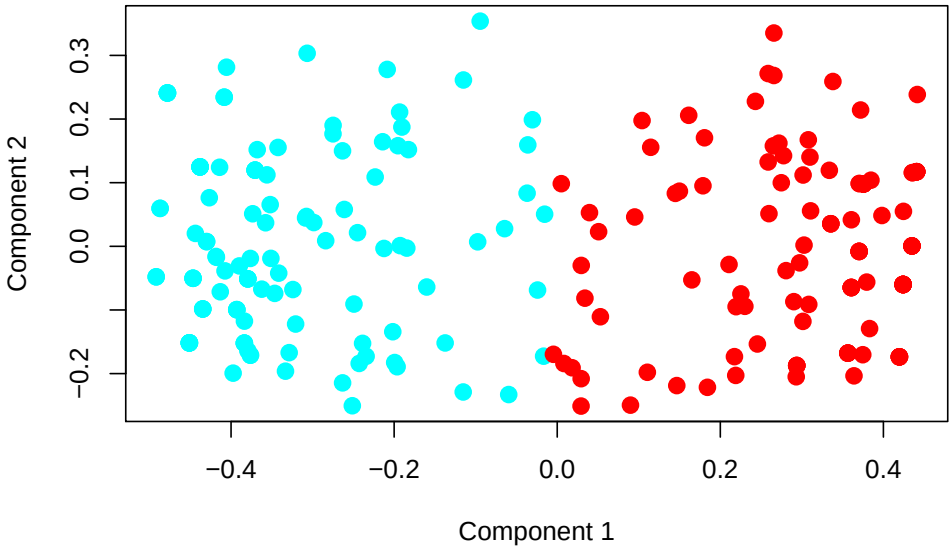

Figure 4: Scatterplot of relational data with plot method. Points are marked according to the obtained classification (Cluster 1: red, Cluster 2: cyan).

Therefore, Cluster 1 and Cluster 2 refer to the Republicans and Democrats, respectively. In Figure 4 the clusters are plotted in the low dimensional space spanned by the first two components. Note that the `plot` method for relational data is based on non-metric multidimensional scaling (Kruskal, 1964) by calling the function `isoMDS` of the package **MASS** (Venables and Ripley, 2002).

To further interpret the clusters, Figure 5 displays the barplots of the 16 key votes for the two clusters (by considering the closest hard clustering partition). We can observe that the votes are highly connected with the Congressmen political positions. This holds for almost all the 16 key votes with particular reference to, e. g. `"adoption-of-the-budget"`, `"education-spending"` and `"anti-satellite-test-ban"`.

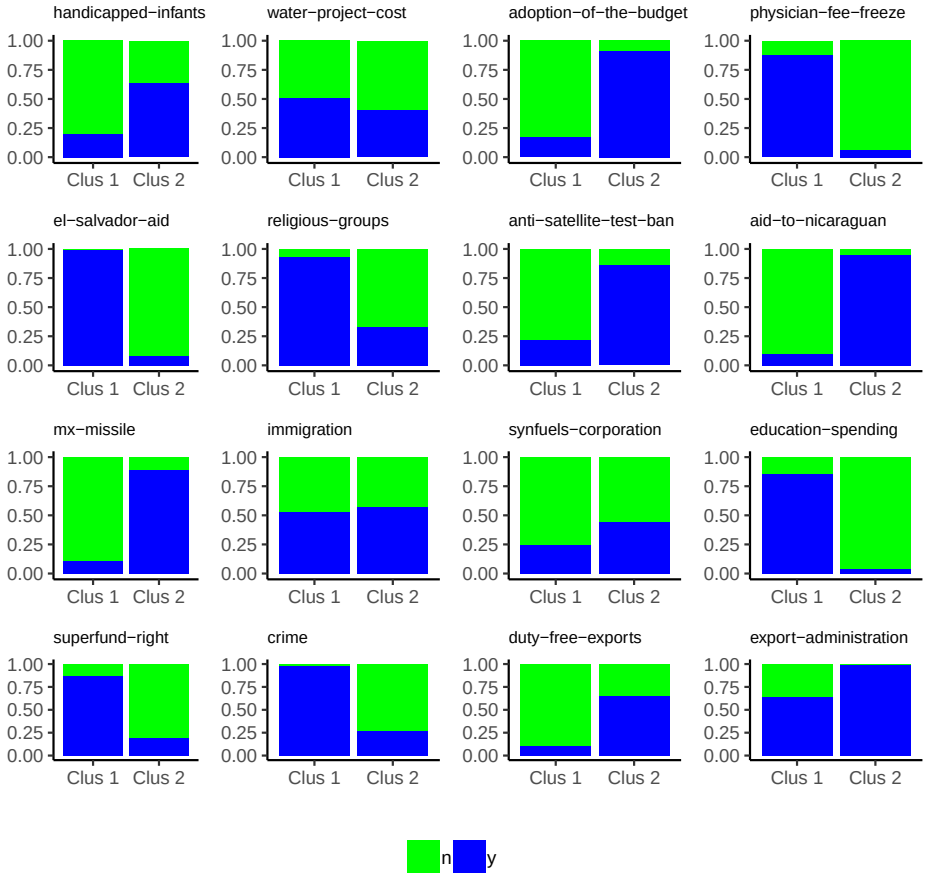

Figure 5: Barplot of the 16 key votes for the two clusters (n: green, y: blue).

Fuzzy clustering for indirect relational data (ordinal variables)

In this section, a dataset with ordinal data is analyzed by using `NEFRC`. The data refer to the Math Anxiety Scale Survey administered to 20 students in a statistics course (Bai et al., 2009). In the survey, each student answers 14 questions by using a Likert scale with five levels (`"Strongly Disagree"`, `"Disagree"`, `"Neutral"`, `"Agree"`, `"Strongly Agree"`). First, we load the dataset:

```
> library(likert)
> data("mass")
```

Then, we compute the dissimilarity matrix by using the Gower distance. When applied to ordinal variables, such a distance is based on ranks. Note that the first variable of `mass` is `Gender`, not useful for clustering purposes and, thus, omitted in the computation of the dissimilarity matrix. We have:

```
> library(cluster)
> D.mass <- daisy(x = mass[,-1], metric = "gower")
```

Finally, we run the function `NEFRC` automatically selecting the number of clusters by means of `SIL.F`:

```
> nefrc.mass <- NEFRC(D = D.mass, index = "SIL.F")
```

The fuzzy silhouette values, employed to select the number of clusters, are:

```
> nefrc.mass$criterion

SIL.F k=2 SIL.F k=3 SIL.F k=4 SIL.F k=5 SIL.F k=6
0.5330319 0.4623684 0.4039311 0.4428360 0.4685703
```

Hence, $k = 2$ clusters are suggested. Since the default options are used, the solution could also be obtained by considering the function `Fclust`:

```
> nefrc.mass <- Fclust(X = D.mass,  k = 2, noise = FALSE, distance = TRUE)
```

The clusters can be interpreted according to the observed qualitative variables. For this purpose, we calculate the p-values resulting from the χ^2 tests by which we study the independence between the closest hard clustering partition and every observed variables. The p-values are stored in the vector `PV`:

```
> PV <- rep(NA,ncol(mass))
> for (j in 1:ncol(mass)) PV[j] <- chisq.test(nefrc.mass$clus[,1], mass[,j])$p.value
```

At the significance level $\alpha = 0.05$, we are interested in those variables such that the corresponding p-value is lower than α:

```
> alpha <- 0.05
> names(mass)[PV < alpha]

 [1] "I find math interesting."
 [2] "I get uptight during math tests."
 [3] "Mind goes blank and I am unable to think clearly when doing my math test."
 [4] "I worry about my ability to solve math problems."
 [5] "I get a sinking feeling when I try to do math problems."
 [6] "I find math challenging."
 [7] "Mathematics makes me feel nervous."
 [8] "Mathematics makes me feel uneasy."
 [9] "Math is one of my favorite subjects."
[10] "I enjoy learning with mathematics."
[11] "Mathematics makes me feel confused."
```

We inspect the contingency tables (not reported here) between such a subset of observed variables and the closest hard clustering partition and we find that Cluster 1 is characterized by large frequencies for the modalities `"Strongly Disagree"` and `"Disagree"` with respect to the variables `"I find math interesting."`, `"Math is one of my favorite subjects."` and `"I enjoy learning with mathematics."` and large frequencies for the modalities `"Agree"` and `"Strongly Agree"` with respect to the variables `"I get uptight during math tests."`, `"Mind goes blank and I am unable to think clearly when doing my math test."`, `"I worry about my ability to solve math problems."`, `"I get a sinking feeling when I try to do math problems."`, `"I find math challenging."`, `"Mathematics makes me feel nervous."`, `"Mathematics makes me feel uneasy."` and `"Mathematics makes me feel confused."`. Of course, the opposite comment holds for Cluster 2. Therefore, the partition distinguishes the students liking math (assigned to Cluster 2) from those who experience feelings of stress when faced with math (assigned to Cluster 1).

Fuzzy clustering for direct relational data

In the previous two subsections, `NEFRC` is applied in order to discover homogeneous clusters of observations on which qualitative variables are collected. In these cases, suitable dissimilarity matrices are built before running `NEFRC`. In the current subsection, we consider the case where variables are not available and the only information about the observations is expressed in terms of their dissimilarities or distances. The data are stored in the following object of class `dist`:

```
> library(smacof)
> data("FaceExp")
```

`FaceExp` contains the dissimilarities between pairs of 13 facial expressions related to particular stimuli:

```
> labels(FaceExp)
```

```
 [1] "Grief at death of mother"         "Savoring a Coke"
 [3] "Very pleasant surprise"           "Maternal love-baby in arms"
 [5] "Physical exhaustion"              "Something wrong with plane"
 [7] "Anger at seeing dog beaten"       "Pulling hard on seat of chair"
 [9] "Unexpectedly meets old boyfriend" "Revulsion"
[11] "Extreme pain"                     "Knows plane will crash"
[13] "Light sleep"
```

The dissimilarities have been calculated in a psychological experiment where a set of subjects were invited to judge how much two pictures of emotional expressions differ. Thus, all the possible pairs of emotional expressions were compared by the subjects and the dissimilarities were derived. See, for further details, Abelson and Sermat (1962) .

By means of NEFRC the aim is to discover whether similar facial expressions are perceived by the subjects in connection with similar emotions intended by the stimuli. In this case, we do not know the number of clusters and, therefore, we determine it according to SIL.F.

```
> nefrc.FaceExp <- NEFRC(D = FaceExp, index = "SIL.F")
```

We find that $k = 3$ should be set:

```
> nefrc.FaceExp$criterion

SIL.F k=2 SIL.F k=3 SIL.F k=4 SIL.F k=5 SIL.F k=6
0.5298465 0.5929045 0.5470887 0.5436513 0.4003177
```

The interpretation of the clusters can be done by seeking a common feature for the facial expressions, i. e. the stimuli, assigned to the same cluster. We have:

```
> round(nefrc.FaceExp$clus[(nefrc.FaceExp$clus[,1] == 1), 2], 2)

Savoring a Coke   Very pleasant surprise  Maternal love-baby in arms
       0.64                0.85                       0.75
Pulling hard on seat    Unexpectedly meets old boyfriend
        0.59                            0.94

> round(nefrc.FaceExp$clus[(nefrc.FaceExp$clus[,1] == 2), 2], 2)

Grief at death of mother      Physical exhaustion    Revulsion
    0.79                              0.81               0.69
Extreme pain    Light sleep
    0.56            0.64


> round(nefrc.FaceExp$clus[(nefrc.FaceExp$clus[,1] == 3), 2], 2)

Something wrong with plane      Anger at seeing dog beaten     Knows plane will crash
       0.52                                0.93                          0.78
```

Cluster 1 groups pleasant stimuli with the only exception of "Pulling hard on seat of chair" for which the membership degree is however the lowest one (0.59). The facial expressions showing pain belong to Cluster 2. "Light sleep" is also assigned to the cluster. It follows that the subjects tend to associate such an expression with suffering. Finally, anxiety characterizes Cluster 3.

Conclusion

In this paper we have described the main features of the package **fclust**. **fclust** represents a toolbox for fuzzy cluster analysis. The functions in the package offer a wide range of fuzzy clustering algorithms, fuzzy cluster validity indices, measures of similarity for comparing hard and fuzzy partitions and visualization tools for fuzzy clustering results. Particular attention has been paid to the new improvements and implementations available in the current version of the package (version 2.1.1). First of all, the functions have been updated by using the C++ language, with a remarkable reduction in computation time. Furthermore, the package now includes some fuzzy clustering algorithms for relational data, allowing the user to perform a fuzzy clustering analysis when the variables are qualitative or mixed. In such cases, a dissimilarity matrix can be built by using the existing R functions (e. g. dist or daisy in the package **cluster**) and the available functions for relational data (NEFRC and NEFRC.noise) can then be applied. As far as we know, NEFRC and NEFRC.noise represent the first

available R functions for fuzzy clustering of qualitative or mixed variables. All the functions have been revised in such a way that the number of clusters can be automatically selected. This might increase the computation time, but it is crucial in order to spread the use of fuzzy clustering methods especially for non-expert users. In this connection, the function `Fclust` for running the available algorithms using the default options and specifying the desired number of clusters is also offered.

Bibliography

R. P. Abelson and V. Sermat. Multidimensional scaling of facial expressions. *Journal of Experimental Psychology*, 63(6):546–554, 1962. URL http://dx.doi.org/10.1037/h0042280. [p15]

R. Babuska, P. J. Van der Veen, and U. Kaymak. Improved covariance estimation for gustafson-kessel clustering. In *Proceedings of the 2002 IEEE International Conference on Fuzzy Systems*, page 1081–1085, 2002. URL https://doi.org/10.1109/FUZZ.2002.1006654. [p4, 5, 6, 10]

H. Bai, L. Wang, W. Pan, and M. Frey. Measuring mathematics anxiety: Psychometric analysis of a bidimensional affective scale. *Journal of Instructional Psychology*, 36(3):185–193, 2009. URL https://eric.ed.gov/?id=EJ952267. [p13]

J. C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Plenum Press, New York, 1981. [p2, 3, 5]

R. J. Campello. A fuzzy extension of the Rand index and other related indexes for clustering and classification assessment. *Pattern Recognition Letters*, 28(7):833–841, 2007. URL https://dx.doi.org/10.1016/j.patrec.2006.11.010. [p2, 6, 9]

Z. Cebeci, F. Yildiz, A. T. Kavlak, C. Cebeci, and H. Onder. *ppclust: Probabilistic and Possibilistic Cluster Analysis*, 2018. URL https://CRAN.R-project.org/package=ppclust. R package version 0.1.1. [p2]

N. C. Chung, B. Miasojedow, M. Startek, and A. Gambin. *Jaccard: Test Similarity Between Binary Data Using Jaccard/Tanimoto Coefficients*, 2018. URL https://CRAN.R-project.org/package=jaccard. R package version 0.1.0. [p2]

R. N. Davé. Characterization and detection of noise in clustering. *Pattern Recognition Letters*, 12(11): 657–664, 1991. URL https://doi.org/10.1016/0167-8655(91)90002-4. [p5]

R. N. Davé and S. Sen. Robust fuzzy clustering of relational data. *IEEE Transactions on Fuzzy Systems*, 10(6):713–727, 2002. URL https://dx.doi.org/10.1109/TFUZZ.2002.805899. [p2, 3, 4, 5]

M. De Caceres, X. Font, and F. Oliva. The management of vegetation classifications with fuzzy clustering. *Journal of Vegetation Science*, 21:1138–1151, 2010. URL https://doi.org/10.1111/j.1654-1103.2010.01211.x. [p2]

J. de Leeuw and P. Mair. Multidimensional scaling using majorization: SMACOF in R. *Journal of Statistical Software*, 31(3):1–30, 2009. URL http://www.jstatsoft.org/v31/i03/. [p4]

D. Eddelbuettel. *Seamless R and C++ Integration with Rcpp*. Springer-Verlag, New York, 2013. URL https://doi.org/10.1007/978-1-4614-6868-4. [p2]

D. Eddelbuettel and C. Sanderson. Rcpparmadillo: Accelerating r with high-performance c++ linear algebra. *Computational Statistics & Data Analysis*, 71:1054–1063, 2014. URL https://dx.doi.org/10.1016/j.csda.2013.02.005. [p2]

M. B. Ferraro and P. Giordani. A new fuzzy clustering algorithm with entropy regularization. In *Proceedings of the 9th Scientific Meeting of the Classification and Data Analysis Group (CLADAG 2013)*, 2013. ISBN 9788867871179. [p5]

M. B. Ferraro and P. Giordani. A toolbox for fuzzy clustering using the r programming language. *Fuzzy Sets and Systems*, 279:1–16, 2015. URL https://dx.doi.org/10.1016/j.fss.2015.05.001. [p2]

J. C. Gower. A general coefficient of similarity and some of its properties. *Biometrics*, 27(4):857–871, 1971. URL https://doi.org/10.2307/2528823. [p2]

D. E. Gustafson and W. C. Kessel. Fuzzy clustering with a fuzzy covariance matrix. In *Proceedings of the 1978 IEEE Conference on Decision and Control Including the 17th Symposium on Adaptive Processes*, page 761–766, 1979. URL https://doi.org/10.1109/CDC.1978.268028. [p3, 5]

M. Halkidi, Y. Batistakis, and M. Vazirgiannis. On clustering validation techniques. *Journal of Intelligent Information Systems*, 17(2-3):107–145, 2001. URL https://doi.org/10.1023/A:1012801612483. [p2]

J. A. Hartigan and M. A. Wong. Algorithm as 136: A K-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1):100–108, 1979. URL https://doi.org/10.2307/2346830. [p1]

K. Hornik. A CLUE for CLUster Ensembles. *Journal of Statistical Software*, 14(12):1–25, 2005. URL https://doi.org/10.18637/jss.v014.i12. [p2]

K. Hornik, I. Feinerer, M. Kober, and C. Buchta. Spherical *k*-means clustering. *Journal of Statistical Software*, 50(10):1–22, 2012. URL https://doi.org/10.18637/jss.v050.i10. [p2]

L. Hubert and P. Arabie. Comparing partitions. *Journal of Classification*, 2(1):193–218, 1985. URL https://dx.doi.org/10.1007/BF01908075. [p2]

P. Jaccard. Étude comparative de la distribution florale dans une portion des alpes et des jura. *Bulletin del la Société Vaudoise des Sciences Naturelles*, 37:547–579, 1901. [p2]

L. Kaufman and P. J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley & Sons, New York, 1990. URL https://doi.org/10.2307/2532178. [p4, 6]

R. Krishnapuram, A. Joshi, O. Nasraoui, and L. Yi. Low-complexity fuzzy relational clustering algorithms for web mining. *IEEE Transactions on Fuzzy Systems*, 9(4):595–607, 2001. URL https://doi.org/10.1109/91.940971. [p5]

J. B. Kruskal. Nonmetric multidimensional scaling: a numerical method. *Psychometrika*, 29(2):115–129, 1964. URL https://doi.org/10.1007/BF02289694. [p13]

R.-P. Li and M. Mukaidono. A maximum-entropy approach to fuzzy clustering. In *Proceedings of 1995 IEEE International Conference on Fuzzy Systems*, page 2227–2232, 1995. URL https://doi.org/10.1109/FUZZY.1995.409989. [p5]

R.-P. Li and M. Mukaidono. Gaussian clustering method based on maximum-fuzzy-entropy interpretation. *Fuzzy Sets and Systems*, 102(2):253–258, 1999. URL https://doi.org/10.1016/S0165-0114(97)00126-7. [p5]

M. Maechler, P. Rousseeuw, A. Struyf, M. Hubert, and K. Hornik. *Cluster: Cluster Analysis Basics and Extensions*, 2017. URL https://cran.r-project.org/web/packages/cluster. R package version 2.0.6. [p2]

D. Meyer, E. Dimitriadou, K. Hornik, A. Weingessel, and F. Leisch. *E1071: Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071), TU Wien*, 2017. URL https://CRAN.R-project.org/package=e1071. R package version 1.6-8. [p2]

W. M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(33):846–850, 1971. URL https://dx.doi.org/10.2307/2284239. [p2]

E. H. Ruspini. Numerical methods for fuzzy clustering. *Information Sciences*, 2(3):319–350, 1970. URL https://doi.org/10.1016/S0020-0255(70)80056-1. [p1]

J. C. Schlimmer. *Concept Acquisition through Representational Adjustment*. Department of Information and Computer Science, University of California, Irvine, 1987. [p11]

L. Scrucca, M. Fop, T. B. Murphy, and A. E. Raftery. mclust 5: Clustering, Classification and Density Estimation Using Gaussian Finite Mixture Models. *The R Journal*, 8(1):289–317, 2016. URL https://journal.r-project.org/archive/2016/RJ-2016-021/index.html. [p2]

W. N. Venables and B. D. Ripley. *Modern Applied Statistics with S*. Springer-Verlag, New York, 2002. URL https://doi.org/10.1007/978-0-387-21706-2. [p13]

R. Winkler, F. Klawonn, F. Höppner, and R. Kruse. Fuzzy cluster analysis of larger data sets. In *Scalable Fuzzy Algorithms for Data Management and Analysis: Methods and Design*, page 302–331. IGI Global, Hershey, 2009. URL https://doi.org/10.4018/978-1-60566-858-1.ch012. [p5]

R. Winkler, F. Klawonn, and R. Kruse. Fuzzy clustering with polynomial fuzzifier function in connection with *m*-estimators. *Applied and Computational Mathematics*, 10(1):146–163, 2011. [p5]

Maria Brigida Ferraro
Department of Statistical Sciences, Sapienza University of Rome
P.le Aldo Moro 5, 00185 Rome, Italy
ORCiD: 0000-0002-7686-5938
mariabrigida.ferraro@uniroma1.it

Paolo Giordani
Department of Statistical Sciences, Sapienza University of Rome
P.le Aldo Moro 5, 00185 Rome, Italy
ORCiD: 0000-0003-4091-3165
paolo.giordani@uniroma1.it

Alessio Serafini
Department of Statistical Sciences, Sapienza University of Rome
P.le Aldo Moro 5, 00185 Rome, Italy
ORCiD: 0000-0002-8579-5695
alessio.serafini@uniroma1.it