Rankcluster: An **R** package for clustering multivariate partial rankings

Julien Jacques[1,2*], Quentin Grimonprez[2], Christophe Biernacki[1,2]

[1] *Laboratoire Paul Painlevé, UMR CNRS 8524, University Lille I, Lille, France*

[2] *MODAL team, Inria Lille-Nord Europe*

Abstract

Rankcluster is the first **R** package proposing both modelling and clustering tools for ranking data, potentially multivariate and partial. Ranking data are modelled by the Insertion Sorting Rank (ISR) model, which is a meaningful model parametrized by a central ranking and a dispersion parameter. A conditional independence assumption allows to take into account multivariate rankings, and clustering is performed by the mean of mixtures of multivariate ISR model. The clusters' parameters (central rankings and dispersion parameters) help the practitioners in the interpretation of the clustering. Moreover, the *Rankcluster* package provides an estimation of the missing ranking positions when rankings are partial. After an overview of the mixture of multivariate ISR model, the *Rankcluster* package is described and its use is illustrated through two real datasets analysis.

Keywords: model-based clustering, multivariate rankings, partial rankings, **R**, Rankcluster.

All results from sections 4 and 5 were obtained with Rankcluster 0.91.6. Since Rankcluster 0.92, the data format has changed: ranks must be provided in their ranking representation (and not ordering representation). This change was made to manage tied objects.

1. Introduction

Ranking data occur when a number of subjects are asked to rank a list of objects according to their personal preference order. Such data are of great interest in human activities involving preferences, attitudes or choices like Psychology, Sociology, Politics,

*Corresponding author. Tel.: +33 320 436 760, Fax: +33 320 434 302

Email addresses: julien.jacques@polytech-lille.fr (Julien Jacques[1,2]), quentin.grimonprez@inria.fr (Quentin Grimonprez[2]), christophe.biernacki@math.univ-lille1.fr (Christophe Biernacki[1,2])

Marketing, *etc.* For instance, the voting system *single transferable vote* occurring in Ireland, Australia and New Zeeland, is based on preferential voting (Gormley and Murphy, 2008). In a lot of applications, the study of ranking data discloses heterogeneity, due for instance to different political meanings, different human preferences, *etc.*

Recently, Jacques and Biernacki (2012) proposed a model-based clustering algorithm in order to analyse and explore such ranking data. This algorithm is able to take into account multivariate rankings with potential partial rankings (when a subject did not rank all the objects). To the best of our knowledge, this is the only clustering algorithm for ranking data with a so wide application scope. This algorithm is based on an extension of the Insertion Sorting Rank (ISR) model (Biernacki and Jacques, 2013) for ranking data, which is a meaningful and effective model obtained by modelling the ranking generating process assumed to be a sorting algorithm. The ISR model is parametrized by a location parameter (the modal ranking) and a dispersion parameter. The heterogeneity of the rank population is modelled by a mixture of ISR whereas conditional independence assumption allows the extension to multivariate rankings. Maximum likelihood estimation is performed through a SEM-Gibbs algorithm, in which partial rankings are considered as missing data, what allows to simulate them during the estimation process.

This algorithm has been implemented in C++ and is available through the *Rankcluster* package for R, available on R-forge (and soon on the CRAN website) and presented at long in the sequel of this paper.

The paper is organised as follows: Section 2 briefly presents the clustering algorithm proposed in Jacques and Biernacki (2012). Section 3 describes the existing R packages dedicated to ranking data, whereas Section 4 discusses the functionalities of the *Rankcluster* package. Section 5 illustrates the use of *Rankcluster* through the cluster analysis of two datasets: 1. Fligner and Verducci's *words* univariate dataset without any missing ranking positions (Fligner and Verducci, 1986), and 2. the rankings of the Big Four of English Football (Manchester United, Liverpool, Arsenal and Chelsea) according to the Premier League results and to their UEFA coefficients between 1993 and 2013 (bivariate dataset with missing ranking positions).

2. Overview of the model-based clustering algorithm

This section gives an overview of the model-based clustering algorithm for multivariate partial rankings proposed in Jacques and Biernacki (2012). It relies on the univariate ISR model that we introduce first.

2.1. The univariate ISR model

Rank data arise when judges or subjects are asked to rank several objects $\mathcal{O}_1, \ldots, \mathcal{O}_m$ according to a given order of preference. The resulting ranking can be designed by its *ordering* representation $x = (x^1, \ldots, x^m) \in \mathcal{P}_m$ which signifies that Object $\mathcal{O}_{x^h}$ is the hth ($h = 1, \ldots, m$), where $\mathcal{P}_m$ is the set of the permutations of the first m integers.

Based on the assumption that a rank datum is the result of a sorting algorithm based on paired comparisons, and that the judge who ranks the objects uses the insertion sort because of its optimality properties (minimum number of paired comparisons), Biernacki and Jacques (2013) state the following so-called ISR model:

$$\mathrm{p}(x;\mu,\pi) = \frac{1}{m!}\sum_{y\in\mathcal{P}_m}\mathrm{p}(x|y;\mu,\pi) = \frac{1}{m!}\sum_{y\in\mathcal{P}_m}\pi^{G(x,y,\mu)}(1-\pi)^{A(x,y)-G(x,y,\mu)}, \tag{1}$$

where

- $\mu \in \mathcal{P}_m$, the modal ranking, is a *location parameter*. Its *opposite* ranking $\bar{\mu}$ ($\bar{\mu} = \mu \circ \bar{e}$ with $\bar{e} = (m,\ldots,1)$) is the rank of smallest probability,
- $\pi \in [\frac{1}{2}, 1]$, which is the probability of good paired comparison according to μ in the sort algorithm, is a *scale parameter*: the distribution is uniform when $\pi = \frac{1}{2}$ and the mode μ of the distribution is uniformly more pronounced when π grows, being a Dirac in μ when $\pi = 1$,
- the sum over $y \in \mathcal{P}_m$ corresponds to all the possible initial presentation orders of the objects to rank (with identical prior probabilities equal to $1/m!$),
- $G(x, y, \mu)$ is equal to the number of good paired comparisons during the sorting process leading to return x when the presentation order is y,
- $A(x, y)$ corresponds to the total number of paired comparisons (good or wrong).

The accurate definitions of $G(x, y, \mu)$ and $A(x, y)$ can be found in Biernacki and Jacques (2013).

2.2. Mixture of multivariate ISR

Let now redefine $x = (x^1,\ldots,x^p) \in \mathcal{P}_{m_1}\times\ldots\times\mathcal{P}_{m_p}$ as a *multivariate* rank, in which $x^j = (x^{j1},\ldots,x^{jm_j})$ is a rank of m_j objects ($1 \le j \le p$).

The population of multivariate ranks is assumed to be composed of K groups in proportions p_k ($p_k \in [0,1]$ and $\sum_{k=1}^K p_k = 1$). Given a group k, the p components $x^1,\ldots,x^p$ of the multivariate rank datum x are assumed to be sampled from independent ISR distributions with corresponding modal rankings $\mu_k^1,\ldots,\mu_k^p$ (each $\mu_k^j \in \mathcal{P}_{m_j}$) and good paired comparison probabilities $\pi_k^1,\ldots,\pi_k^p \in [\frac{1}{2}, 1]$.

The unconditional probability of a rank x is then

$$\mathrm{p}(x;\boldsymbol{\theta}) = \sum_{k=1}^K p_k \prod_{j=1}^p \frac{1}{m_j!}\sum_{y\in\mathcal{P}_{m_j}}\mathrm{p}(x^j|y;\mu_k^j,\pi_k^j), \tag{2}$$

where $\boldsymbol{\theta} = (\pi_k^j,\mu_k^j,p_k)_{k=1,\ldots,K\,,j=1,\ldots,p}$ and $\mathrm{p}(x^j|y;\mu_k^j,\pi_k^j)$ is defined by (1).

Each component x^j of x can be full or partial. Frequently, the objects in the top positions will be ranked and the missing ones will be at the end of the ranking, but our model does not impose such situation and is able to work with partial ranking whatever are the positions of the missing data (see details in Jacques and Biernacki (2012)).

2.3. Estimation algorithm

Let $\mathbf{x} = \{x_1, \ldots, x_n\}$ be a sample of n multivariate rankings, and $\mathbf{z} = \{z_1, \ldots, z_n\}$ the corresponding latent cluster memberships. Let $\check{I}_i^j$ and $\hat{I}_i^j$ be respectively the sets of indices of observed and unobserved positions in the jth component x_i^j of the ith observation x_i. Similarly, let $\check{x}_i^j$ and $\hat{x}_i^j$ correspond to the previous notations for the jth component of the ith observation, $\check{x}_i = \{\check{x}_i^1, \ldots, \check{x}_i^p\}$ and $\hat{x}_i = \{\hat{x}_i^1, \ldots, \hat{x}_i^p\}$. Let also define $\check{\mathbf{x}} = \{\check{x}_i; i = 1, \ldots, n\}$ and $\hat{\mathbf{x}} = \{\hat{x}_i; i = 1, \ldots, n\}$. Finally $y_i = (y_i^1, \ldots, y_i^p) \in \mathcal{P}_{m_1} \times \ldots \times \mathcal{P}_{m_p}$ denotes the presentation orders of the objects for the ith observation and $\mathbf{y} = \{y_1, \ldots, y_n\}$.
Assuming that triplets (x_i, y_i, z_i) arise independently $(i = 1, \ldots, n)$, the observed-data log-likelihood of model (2) is:

$$l(\boldsymbol{\theta}; \check{\mathbf{x}}) = \sum_{i=1}^{n} \ln \left(\sum_{k=1}^{K} p_k \prod_{j=1}^{p} \frac{1}{m_j!} \sum_{y \in \mathcal{P}_{m_j}} \sum_{x \in \mathcal{X}_i^j} \mathrm{p}(x|y; \mu_k^j, \pi_k^j) \right),$$

where $\mathcal{X}_i^j = \{x \in \mathcal{P}_{m_j} : x^h = \check{x}_i^{jh}, \forall h \in \check{I}_i^j\}$ is the set of all the rankings compatible with the observed part $\check{x}_i^j$ of x_i^j.

Maximum likelihood estimation is not straightforward since several missing data occur: the cluster memberships z_i of the observations, the presentation orders y_i and the unobserved ranking positions $\hat{x}_i$ (for partial rankings). In such a situation, a convenient way to maximize the likelihood is to consider an EM algorithm (Dempster et al., 1977). This algorithm relies on the completed-data log-likelihood, and proceeds in iterating an E step, in which the conditional expectation of the completed-data log-likelihood is computed, and a M step, in which the model parameters are estimated by maximizing the conditional expectation computed in the E step. Unfortunately, the EM algorithm is tractable only for univariate full rankings with moderate m ($m \leq 7$), respectively for mathematical and numerical reasons. In particular, when partial rankings occur, the E step is intractable since the completed-data log-likelihood is not linear for all three types of missing data (refer to Jacques and Biernacki (2012) for its expression). A SEM-Gibbs approach is then proposed in Jacques and Biernacki (2012) to overcome these problems.

The fundamental idea of this algorithm is to reduce the computational complexity that is present in both E and M steps of EM by removing all explicit and extensive use of the conditional expectations of any product of missing data. First, it relies on the SEM algorithm (Geman and Geman, 1984; Celeux and Diebolt, 1985) which generates the latent variables at a so-called stochastic step (S step) from the conditional probabilities computed at the E step. Then these latent variables are directly used in the M step. Second, the advantage of the SEM-Gibbs algorithm in comparison with the basic SEM ones relies on the fact that the latent variables are generated without calculating conditional probabilities at the E step, thanks to a Gibbs algorithm. Refer to Jacques and Biernacki (2012) for more details.
Let noticed that label switching can occur with the SEM algorithm when clusters are not

well separated. To avoid this situation, we recommend to use model selection criteria as BIC (Schwarz, 1978) or ICL (Biernacki et al., 2000) to select the number K of clusters.

3. Existing R packages for ranking data

To the best of our knowledge, there exists only two packages dedicated to the analysis of ranking data, available on the CRAN website, but none of them seems to work to date. More important, their announced functionalities are significantly limited in comparison to our package *Rankcluster*, as we discuss now:

- *pmr* package for R: provide some descriptive statistics and modelling tools using classical rank data models for full univariate ranking data: Luce models, distance-based models, and rank-ordered logit (refer to Marden (1995) for a description of these models). Visualization of ranking data using polytopes is also available for less than four objects to rank ($m \leq 4$).
  Let note that even after correcting their `NAMESPACE` file (by adding the missing command line `exportPattern("^[[:alpha:]]+")`) and recompiling their package, the main function seems to not work even with the proposed examples.

- *RMallow* package for R: suppose to perform clustering of univariate ranking data using mixture of Mallows model (Murphy and Martin (2003)).

Rankcluster proposes modelling and clustering tools on the basis of the mixture of multivariate ISR presented model in Section 2. Comparing to the existing packages, *Rankcluster* is the only package taking into account multivariate and partial ranking data.

4. Overview of the *Rankcluster* functions

This section presents functions from Rankcluster 0.91.6. Since Rankcluster 0.92, the data format has changed: ranks (`data` parameter) must be provided in their ranking representation[1] (and not ordering representation). This change was made to manage tied objects.

This section presents, firstly, the main function `rankclust()` which performs cluster analysis, and, secondly, several companion functions which can be helpful for additional ranking data analysis.

4.1. The main function: `rankclust()`

Cluster analysis can be performed with the `rankclust()` function. Illustration of its use is given in Section 5.

[1] The *ranking* representation $x^{-1} = (x_1^{-1}, \ldots, x_m^{-1})$ contains the ranks assigned to the objects, and means that Object $\mathcal{O}_i$ is in the x_i^{-1}th position ($i = 1, \ldots, m$). Notice that x is associated to the ordering representation.

4.1.1. Input arguments

This function has only one mandatory argument, `data`, which is a matrix composed of the n observed ranks in their ordering representation. For **univariate rankings** the number of columns of `data` is m (default value of argument `m`). For **multivariate rankings**, `data` has $m_1 + \ldots + m_p$ columns: the first m_1 columns contain x^1 (first dimension), the columns $m_1 + 1$ to $m_1 + m_2$ contain x^2 (second dimension), and so on. In this case, the argument `m` must be filled with the vector of sizes $(m_1, \ldots, m_j)$. If the user works with a **ranking**[2] **representation** of the ranks, the `convertRank()` function can be used to transform ranking representation into ordering one.

The number of clusters (1 by default) can be set up with option `K`. Vector of numbers of clusters are possible (for instance `K=1:10`). In order to select the number of clusters, two criteria are available: BIC, by default, and ICL, selected by `criterion = "icl"`.

Additionally, several parameters allow to set up the different tuning parameters (iterations numbers) used in the SEM-Gibbs estimation. Refer to Jacques and Biernacki (2012) and to `rankclust()` help for more details. Section 5 gives also some examples of iterations numbers choices. The option `run` allows to set the number of initializations of the SEM-Gibbs algorithm (1 by default). In the case of multiple initializations, the best solution according to the approximated log-likelihood is retained.

Finally, the computing times (total, for the SE and M steps and for the likelihood approximation) can be printed by setting the option `detail` to `TRUE` (`FALSE` by default).

4.1.2. Output arguments

The `rankclust()` function returns an instance of the `ResultTab` class. Its attributes will contain 4 slots:

- `K`: a vector of the number of clusters,
- `results`: a list of `ResultList` class, containing the results for each number of clusters (one element of the list is associated to one number of clusters),
- `data`: the data used for clustering,
- `criterion`: the model selection criterion used,
- `convergence`: a boolean indicating if none problem of empty class has been encountered (for any number of clusters).

Each element of the list `results` contains all the results for a given number K of classes, which are summarized in the following 18 slots:

[2]The *ranking* representation $x^{-1} = (x_1^{-1}, \ldots, x_m^{-1})$ contains the ranks assigned to the objects, and means that Object $\mathcal{O}_i$ is in the x_i^{-1}th position $(i = 1, \ldots, m)$. Notice that x is associated to the ordering representation.

- `proportion`: a K-vector of proportions $p_1, \ldots, p_K$,
- `pi`: a $K \times p$-matrix composed of the scale parameters π_k^j ($1 \leq k \leq K$ and $1 \leq j \leq p$),
- `mu`: a matrix with K lines and $m_1 + \ldots + m_p$ columns in which line k is composed of the location parameters $(\mu_k^1, \ldots, \mu_k^p)$ of cluster k,
- `ll, bic, icl`: values of the log-likelihood, BIC criterion and ICL criterion,
- `tik`: a $n \times K$-matrix containing the estimation of the conditional probabilities for the observed ranks to belong to each cluster,
- `partition`: a n-vector containing the partition estimation resulting from the clustering,
- `entropy`: a $n \times 2$-matrix containing for each observation its estimated cluster (column 2, similar to `partition`) and its entropy (column 1), defined as $-\sum_{k=1}^{K} t_{ik} \log(t_{ik})$ where t_{ik} is the conditional probabilities for the ith observation to belong to cluster k, given by `tik`. The `entropy` output illustrates the confidence in the clustering of each observation (a high entropy means a low confidence in the clustering),
- `probability`: a $n \times 2$-matrix similar to the `entropy` output, containing for each observation its estimated cluster (column 2, similar to `partition`) and its probability $p(x_i; \mu_k, \pi_k)$ given its cluster. This probability is estimated using the last simulation of the presentation orders used for the likelihood approximation. The `probability` output exhibits the best representative of each cluster.
- `convergence`: a boolean indicating if none problem of empty class has been encountered,
- `partial`: a boolean indicating the presence of partial rankings,
- `partialRank`: a matrix containing the full rankings, estimated using the within cluster ISR parameters when the ranking is partial. When ranking is full, `partialRank` simply contains the observed ranking. Available only in presence of at least one partial ranking.
- `distanceProp, distancePi, distanceMu`: distances between the final estimation and the current value at each iteration of the SEM-Gibbs algorithm (except the burning phase) for respectively: proportions p_k, scale parameters π_k^j, location parameters μ_k^j. For μ_k^j, the Kendall distance for ranking has been considered (Marden, 1995). For p_k and π_k^j, the distance is the mean squared difference. `distanceProp`, `distancePi` and `distanceMu` are lists of `Qsem-Bsem` elements, each element being a $K \times p$-matrix.
  These elements are reachable by `res[k]@slotname[[iteration]]`.

- `distanceZ`: a vector of size `Qsem-Bsem` containing the rand index (Rand, 1971) between the final estimated partition and the current value at each iteration of the SEM-Gibbs algorithm (except the burning phase). Let precise that the rand index is not affected by label switching.

- `distancePartialRank`: Kendall distance between the final estimation of the partial rankings (missing positions in such rankings are estimated) and the current value at each iteration of the SEM-Gibbs algorithm (except the burning phase). `distancePartialRank` is a list of `Qsem-Bsem` elements, each element being a matrix of size $n \times p$. Available only in presence of at least one partial ranking.

- `proportionInitial, piInitial, muInitial, partialRankInitial`: initializations of the parameters in the SEM-Gibbs algorithm (for expert use only).

If `res` is the result of `rankclust()`, each slot of `results` can be reached by `res[k]@slotname`, where `k` is the number of clusters and `slotname` is the name of the slot we want to reach (`proportion`, `pi` ...). For the slots `ll, bic, icl`, `res["slotname"]` returns a vector of size K containing the values of the slot for each number of clusters.

4.2. Companion functions

In addition to the main function, `rankclust()`, several companion functions are available in *Rankcluster*:

- `convertRank()`: converts ranking representation x^{-1} of a rank to its ordering representation x, and *vice-versa* since $x \circ x^{-1} = x^{-1} \circ x$.

- `criteria()`: estimate the log-likelihood, BIC and ICL criteria from a dataset and a corresponding estimation of the ISR model parameters (see details with `help(criteria)`).

- `distCayley(), distHamming(), distKendall(), distSpearman()`: compute usual distances between rankings (refer to Marden (1995)) for either ranking or ordering representation (refer to the help of the functions for details),

- `frequence()`: transform a raw dataset composed of a list of ranks into a matrix rank/frequency, in which the last column is the frequency of observation of the rank. Conversely, `unfrequence()` transform a rank/frequency dataset in a raw dataset, as requested in input argument of `rankclust()`,

- `khi2()`: perform a chi-squared goodness-of-fit tests and return the p-value of the test (refer to Biernacki and Jacques (2013) for details).

- `kullback()`: estimate the Kullback-Leibler divergence between two ISR models,

- `simulISR()`: simulate a univariate and unimodal dataset of rankings according to the ISR model,

5. *Rankcluster* through examples

All results from this section were obtained with Rankcluster 0.91.6. Since Rankcluster 0.92, the data format has changed: ranks must be provided in their ranking representation (and not ordering representation). This change was made to manage tied objects.

This section illustrates the use of the `rankclust()` function on two real datasets. The first one, `words`, is a well-known dataset in ranking study, due to Fligner and Verducci (1986), which consists of words associations by students. The second one, `big4` consists of the rankings of the Big Four of English Football (Manchester United, Liverpool, Arsenal and Chelsea) according to the Premier League results and to their UEFA coefficients between 1993 and 2013 (see Appendix Appendix A).

5.1. The `words` dataset

This data was collected under the auspices of the Graduate Record Examination Board (Fligner and Verducci, 1986). A sample of 98 college students were asked to rank five words according to strength of association (least to most associated) with the target word "Idea": 1 = Thought, 2 = Play, 3 = Theory, 4 = Dream and 5 = Attention.

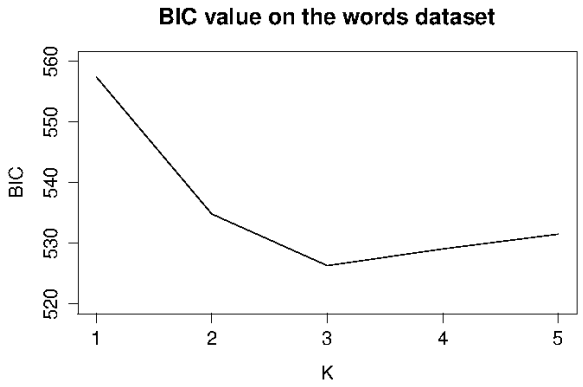

Figure 1: Value of the BIC criterion with mixture of ISR for the `words` dataset.

First we start by installing and loading the *Rankcluster* package:

```
R> install.packages("Rankcluster")
R> library(Rankcluster)
```

and then loading the `words` dataset:

```
R> data(words)
```

Using the `rankclust()` function, a clustering with respectively 1 to 5 clusters is estimated:

```
R> res=rankclust(words$data,m=words$m,K=1:5,Qsem=1000,Bsem=100,Ql=500,Bl=50,
```

maxTry=20,run=10)

The number of SEM-Gibbs iterations (Qsem) has been set to 1000, with a burning phase of 100 iterations (Bsem). For likelihood approximation the numbers of iterations (Ql and Bl) have been divided by two. Option maxTry=20 allows to restart the estimation algorithm in the limit of 20 times if one cluster becomes empty (frequent for $K = 5$). Finally, the SEM-Gibbs algorithm is initialized 5 times (run=5), and the best solution (according to the approximated likelihood) is retained. Computing time on a laptop with 2.80GHz CPU is about 3 minutes (7 seconds per run et per number of clusters). The reader who wants to test more quickly the package can reduced the number of runs.

The values of the BIC criterion, reached by res["bic"] and plotted on Figure 1, tend to select three clusters.

The parameter estimation for $K = 3$ are given below for proportions p_k, probabilities π_k and modes μ_k:

```
> res[3]@proportion
[1] 0.3061224 0.4918367 0.2020408
> res[3]@pi
dim 1
cl 1 0.9060649
cl 2 0.9416822
cl 3 0.8642753
> res[3]@mu
dim 1
cl 1 2 5 3 4 1
cl 2 2 5 4 3 1
cl 3 5 2 4 3 1
```

The words Thought is the most associated with Idea for all clusters. Regarding the rankings of the four other words can suggest an interesting interpretation of the clusters. Indeed, the first cluster, composed of about 30% of the students, is characterized by the following modal ranking: Play, Attention, Theory, Dream, Thought. Students of this cluster are probably literary-minded students, rankings the word Dream just after Thought. Students of the second cluster (about half of total students) are probably more scientific since they rank the word Theory just after Thought, and so before the word Dream: Play, Attention, Dream, Theory, Thought. This cluster is also the most homogeneous, with a high scale parameter value (low dispersion): $\pi_2 \simeq 0.94$. Finally, the last cluster is characterized by the following mode: Attention, Play, Dream, Theory, Thought. The only difference in the modal ranking with the scientific students is the preference of Play rather than Attention. This cluster, which is the smallest (20% of the students), can be qualified as intermediary cluster, probably composed of a set of students not too scientific or too literary-minded, as evidenced by the smallest of the three scale parameter values ($\pi_3 \simeq 0.86$).

5.2. The `big4` *dataset*

In the two last decades, the English football has been dominated by four clubs, forming the "Big Four": Manchester United, Liverpool, Arsenal and Chelsea. In this application, we analyse the rankings of these four teams at the English championship (Premier League) and their rankings according to the UEFA coefficients. This coefficient is an European statistic on football teams based on the results of European football competitions and used for ranking and seeding teams in international competitions. The `big4` dataset is composed of Premier League rankings and UEFA rankings from 1993 to 2013, in ordering notation (see Table A.2 in Appendix Appendix A, in which club "1" is Manchester United, "2" is Liverpool, "3" is Arsenal and "4" is Chelsea). In 2001 Arsenal and Chelsea had the same UEFA coefficient and then are tied for the first ranking dimension. With *Rankcluster*, one way to take into account such tied in ranking data is to consider the corresponding ranking positions as missing: the UEFA ranking becomes then $(1, 0, 0, 2)$ for 2001, what means that Manchester United is the first, Liverpool is the last, and the two intermediate positions are for Arsenal and Chelsea in an unknown order.

First, the `big4` dataset is loaded:

```
R> data(big4)
```

Clustering for 1 to 3 clusters is estimated with `rankclust()` function in about 25 seconds on a laptop 2.80GHz CPU (less than 2 seconds per run and per number of clusters):

```
R> res=rankclust(big4$data,m=big4$m,K=1:3,Qsem=1000,Bsem=100,Ql=500,Bl=50,
maxTry=20,run=5)
```

The values of the BIC criterion are plotted on Figure 2, and tend to select two groups.

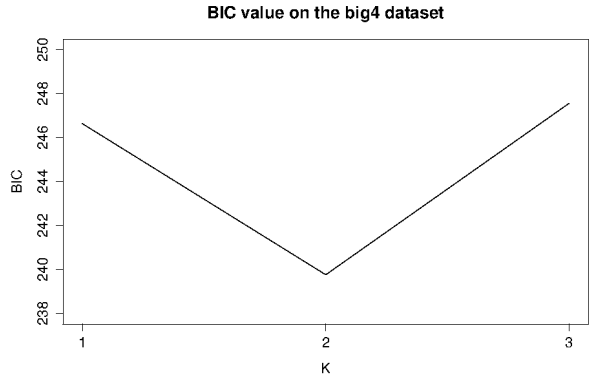

Figure 2: Values of the BIC criterion with mixture of ISR for the `big4` dataset.

The printed outputs for $K = 2$ are given below: value of the log-likelihood (`ll`), values of BIC and ICL criteria, estimation of the proportions p_k's, the dispersion parameters

π_k^j's, the location parameters μ_k^j's, the estimated partition and finally the conditional probability of the observations to belong to each cluster (t_{ik}).

```
R> res[2]
**********************************************************************
Number of clusters:  2
**********************************************************************
ll= -108.5782
bic = 244.5571
icl = 253.5601
proportion:  0.3854034 0.6145966
mu:
dim1 dim2
cl1 1 3 2 4 1 3 2 4
cl2 1 4 2 3 1 4 3 2
pi:
dim1 dim2
cl1 0.9698771 0.7759781
cl2 0.6945405 0.7707101
partition:
[1] 2 1 1 1 2 1 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2
tik:
              [,1]          [,2]
[1,] 6.280426e-01 0.371957427
[2,] 9.933149e-01 0.006685094
[3,] 9.565052e-01 0.043494814
[4,] 9.923445e-01 0.007655450
[5,] 6.609592e-01 0.339040841
[6,] 9.916482e-01 0.008351815
[7,] 1.215925e-03 0.998784075
[8,] 3.500066e-01 0.649993363
[9,] 3.441535e-02 0.965584647
[10,] 4.063712e-01 0.593628767
[11,] 9.589745e-01 0.041025518
[12,] 9.751644e-01 0.024835551
[13,] 6.806917e-02 0.931930833
[14,] 3.175113e-03 0.996824887
[15,] 1.263591e-03 0.998736409
[16,] 7.713598e-06 0.999992286
[17,] 9.115424e-04 0.999088458
[18,] 1.734860e-01 0.826513968
[19,] 1.605939e-04 0.999839406
[20,] 7.425989e-02 0.925740107
[21,] 2.171738e-02 0.978282624
```

****

The estimated clustering exhibits two groups, the second one being larger than the first one ($p_1 \simeq 0.39$ and $p_2 \simeq 0.61$). The values of the location parameters for each cluster and dimension leads to two interesting remarks. First, the ranking in each dimension is very similar in both clusters: exactly the same for cluster 1 and just one transposition in the last two positions for cluster 2. This means that the performance of the clubs at the Premier League is highly correlated with their UEFA rankings, which is related to the results of the clubs in the European competitions over the previous five seasons. This first comment shows a certain inertia in the performance of the clubs. Secondly, the distinction between the two clusters is essentially due to the position of the club "4", Chelsea: indeed, in the first cluster Chelsea is the last in both rankings, but it is in second position in the second cluster. Moreover, on the partition, we find that cluster 2 is mainly present in the second half of the period 1993-2013 (see for instance the conditional probabilities of cluster membership on Figure 3). This rise of Chelsea's results can be explained by the acquisition of the club by a Russian investor (Abramovich) in 2003, who brought great players in the club.

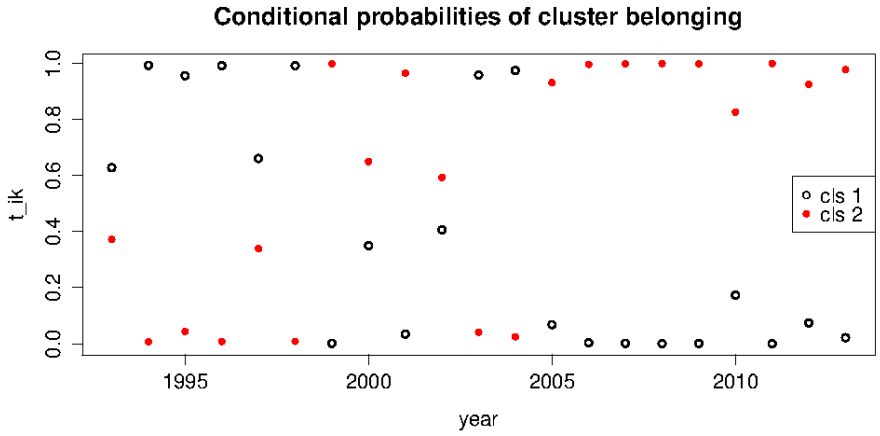

Figure 3: Conditional probabilities for each observation (year) to belong to cluster 1 (black circle) and two (red filled circle).

In addition to this information, the `summary()` function gives an overview of the partition by printing the five ranks of highest probability and the five ranks of highest entropy for each cluster:

```
R> summary(res)
```

The ranks of highest probability are the best representatives of the cluster, whereas the ranks of highest entropy are those for which their membership to the cluster are the less obvious. Notice that the full list of the cluster member with their probability and

entropy are available through the slots `probability` and `entropy`. Table 1 gives an example of these outputs for cluster 2. The rankings of 2011 are the most representative of the cluster, and the five most representatives of the cluster correspond to rankings after 2003. Similarly, the four observations whose membership to cluster 2 is the most questionable correspond to observations before 2003. This information confirms the previous analysis indicating that cluster 2 is due to the advent of Chelsea in the first positions.

year	UEFA	Prem. League	entropy
2002	(1,3,4,2)	(3,2,1,4)	0.6755106
1993	(1,2,3,4)	(1,2,3,4)	0.6599892
2000	(1,4,3,2)	(1,3,2,4)	0.6474507
1997	(1,2,3,4)	(1,3,2,4)	0.6403972
2010	(1,0,0,4)	(4,1,3,2)	0.4613709
year	UEFA	Prem. League	probability
2011	(1,4,2,3)	(1,4,3,2)	2.367e-04
2008	(4,2,3,1)	(1,4,3,2)	6.862e-05
2013	(4,1,3,2)	(1,4,3,2)	3.529e-05
2009	(4,2,1,3)	(1,2,4,3)	3.097e-05
2005	(1,2,3,4)	(4,3,1,2)	2.151e-05

Table 1: Rankings with the highest entropies and probabilities in the second cluster.

The `summary()` function prints also the estimated full ranking for each partial ranking. For instance, in 2001 Arsenal and Chelsea had the same UEFA coefficient, and when asking to our model to differentiate these two teams, Arsenal is ranked before Chelsea, what is not surprising as we already remarked that the results of Chelsea were generally among the worst of the Big Four before 2003.

Finally, the variability of estimation of the model parameters can be achieved by the mean of the distances between the final estimation and the current value at each step of the SEM-Gibbs algorithm (refer to Jacques and Biernacki (2012) for accurate definitions of these distances). These distances are available in the slots `distanceProp, distancePi, distanceMu` of the output `res[2]`. The standard deviation of these distances can be used for instance as an indicator of estimation variability. For instance, the standard deviation of the Kendall distance between the final estimation of the location parameters and its current value at each step of the SEM-Gibbs algorithm is for cluster 2: 0.52 for the UEFA coefficients rankings and 0.43 for Premier League rankings. Similarly, the standard deviation of the dispersion parameters estimations is for cluster 2 about 0.002 for both the UEFA coefficients and the Premier League rankings. Let note that these variabilities are relatively small, due to the low overlapping of the two clusters (dispersion coefficients are quite high). In a similar way, the slot `distancePartition`

illustrates the convergence of the SEM-Gibbs algorithm by given the rand index between the final partition and the current partition at each SEM-Gibbs iteration (Figure 4).

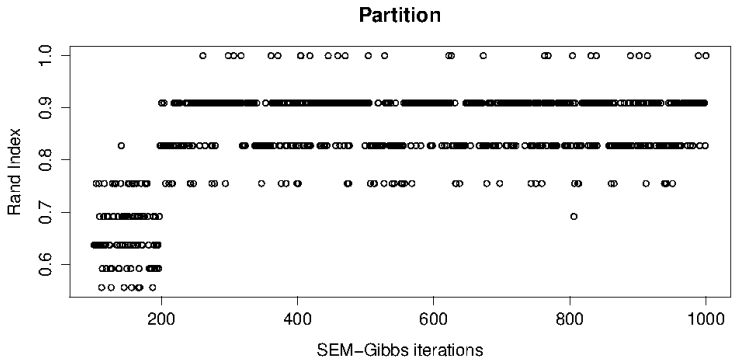

Figure 4: Evolution of the partition along with the SEM-Gibbs iterations: values of the Rand index between current and final partitions.

6. Conclusion

Rankcluster is the first R package dedicated to ranking data, allowing modelling and cluster analysis for multivariate partial ranking data. Available on R-forge, this package is simple of use with its main function, `rankclust()`, having only one mandatory argument, the ranking dataset. By default a modelling is performed, and mentioning the number of desired clusters leads to perform a cluster analysis, with selection of the number of clusters if several numbers are given. The analysis of two real datasets presented in this paper allows to illustrate the possibilities of the package, and also constitutes a user guide for the practitioners.

Appendix A. The big4 dataset

The `big4` dataset[3] (Table A.2) is composed of the rankings (in ordering notation) of the "Big Four" of English football: Manchester United (quoted by 1), Liverpool (2), Arsenal (3) and Chelsea (4), at the English football championship (Premier League) and according to their UEFA coefficients (statistics based on European competitions), from 1993 to 2013.

[3]Sources: Wikipedia website `http://en.wikipedia.org/wiki/Big_Four_(English_football)` and UEFA website `http://fr.uefa.com/memberassociations/uefarankings/club/index.html`

year	Premier League	UEFA coefficient
1993	(1,2,3,4)	(1,2,3,4)
1994	(1,3,2,4)	(1,3,2,4)
1995	(1,3,2,4)	(1,2,4,3)
1996	(1,3,2,4)	(1,2,3,4)
1997	(1,2,3,4)	(1,3,2,4)
1998	(1,3,2,4)	(3,1,2,4)
1999	(1,4,2,3)	(1,3,4,2)
2000	(1,4,3,2)	(1,3,2,4)
2001	(1,0,0,2)	(1,4,2,3)
2002	(1,3,4,2)	(3,2,1,4)
2003	(1,3,2,4)	(1,3,4,2)
2004	(1,3,2,4)	(3,4,1,2)
2005	(1,2,3,4)	(4,3,1,2)
2006	(2,3,1,4)	(4,1,2,3)
2007	(2,3,4,1)	(1,4,2,3)
2008	(4,2,3,1)	(1,4,3,2)
2009	(4,2,1,3)	(1,2,4,3)
2010	(1,0,0,4)	(4,1,3,2)
2011	(1,4,2,3)	(1,4,3,2)
2012	(1,4,3,2)	(1,3,4,2)
2013	(4,1,3,2)	(1,4,3,2)

Table A.2: Rankings of the "Big Four" of English football: Manchester United (1), Liverpool (2), Arsenal (3) and Chelsea (4), according to the Premier League results and to their UEFA coefficients from 1993 to 2013.

References

Biernacki, C., Celeux, G., Govaert, G., 2000. Assessing a mixture model for clustering with the integrated completed likelihood. IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (4), 719–725.

Biernacki, C., Jacques, J., 2013. A generative model for rank data based on sorting algorithm. Computational Statistics & Data Analysis 58, 162–176.

Celeux, G., Diebolt, J., 1985. The SEM algorithm: A probabilistic teacher algorithm derived from the EM algorithm for the mixture problem. Computational Statistics Quarterly 2 (1), 73–82.

Dempster, A., Laird, N., Rubin, D., 1977. Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society. Series B. Methodological 39 (1), 1–38, with discussion.

Fligner, M., Verducci, J., 1986. Distance based ranking models. Journal of the Royal Statistical Society. Series B. Methodological 48 (3), 359–369.

Geman, A., Geman, D., 1984. Stochastic relaxation, gibbs distributions and the bayesian restoration of images. IEEE Transactions on Pattern Analysis and Matching Intelligence 6, 721–741.

Gormley, I., Murphy, T., 2008. A mixture of experts model for rank data with applications in election studies. Annals of Applied Statistics 2 (4), 1452–1477.

Jacques, J., Biernacki, C., 2012. Model-based clustering for multivariate partial ranking data. Tech. Rep. 8113, Inria Research Report.

Marden, J., 1995. Analyzing and modeling rank data. Vol. 64 of Monographs on Statistics and Applied Probability. Chapman & Hall, London.

Murphy, T., Martin, D., 2003. Mixtures of distance-based models for ranking data. Comput. Statist. Data Anal. 41 (3-4), 645–655.

Rand, W., 1971. Objective criteria for the evaluation of clustering methods. Journal of the American Statistical Association 55, 846–850.

Schwarz, G., 1978. Estimating the dimension of a model. The Annals of Statistics 6 (2), 461–464.