Rankcluster: An R Package for Clustering Multivariate Partial Rankings
by Julien Jacques, Quentin Grimonprez and Christophe Biernacki

Abstract The Rankcluster package is the first R package proposing both modeling and clustering tools for ranking data, potentially multivariate and partial. Ranking data are modeled 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 multivariate rankings to be taken into account, and clustering is performed by means of mixtures of multivariate ISR models. The parameters of the cluster (central rankings and dispersion parameters) help the practitioners to interpret the clustering. Moreover, the Rankcluster package provides an estimate of the missing ranking positions when rankings are partial. After an overview of the mixture of multivariate ISR models, the Rankcluster package is described and its use is illustrated through the analysis of two real datasets.

Introduction

Ranking data occur when a number of observers 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, Marketing, etc. For instance, the voting system single transferable vote, used in Ireland, Australia and New Zealand, is based on preferential voting (Gormley and Murphy, 2008). In many applications, the study of ranking data discloses heterogeneity, due for instance to different political meanings, different human preferences, etc. Clustering analysis is then a useful tool for statistical exploration of such datasets, by characterizing the datasets in terms of a reduced number of clusters within which rankings are similar. In the present work, the clusters will be characterized by a median ranking (location parameter) and by a dispersion parameter. The interpretation of these two parameters for each cluster will produce a comprehensive and synthetic analysis of the whole dataset.

In the present paper we illustrate these ideas with two studies in sociology and in sport. In the sociology study, several students were asked to rank five words according to strength of association (least to most associated) with the target word “Idea”: Thought, Play, Theory, Dream and Attention. The cluster analysis of the resulting rankings from this study allows the population of students to be characterized in terms of three clusters that can be interpreted (thanks to the parameters specific to each cluster) as follows: scientific students, literary-minded students and a third group of students in an intermediate position between scientific and too literary-minded. The second study focuses on the performance of English football clubs during the last two decades, and more particularly those of the “Big Four” clubs: Manchester United, Liverpool, Arsenal and Chelsea. For this, the rankings of these four clubs at the English championship (Premier League) and according to the UEFA coefficients are analyzed. The cluster analysis exhibits two relatively similar clusters, reflecting a certain stability in the hierarchy between these clubs, except for the position of Chelsea within the hierarchy: it appears that Chelsea obtains better results in the second half of the period 1993-2013, which can be probably explained by the acquisition of the club by a Russian investor in 2003.

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 (when several rankings are simultaneously observed, as for instance in the sport application in which rankings at the Premier League and rankings according to the UEFA coefficient are observed simultaneously each year) with potential partial rankings (when an observer did not rank all the objects). To the best of our knowledge, this is the only clustering algorithm for ranking data with such a 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 assuming that the ranking generating process is 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 modeled by a mixture of ISR models and a 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, which allows them to be simulated during the estimation process.

This algorithm has been implemented in C++ and is available through the Rankcluster package for R, available on the CRAN website and presented in depth in the sequel of this paper.

The paper is organized as follows: the next section briefly presents the clustering algorithm
proposed in Jacques and Biernacki (2012). Then, the following sections describe the existing R packages dedicated to ranking data, and the functionality of the **Rankcluster** package. The **Rankcluster** package is then illustrated through the cluster analysis of two datasets: 1. The *words* dataset of Fligner and Verducci (1986) featuring univariate rankings without any missing ranking positions, and 2. the rankings of the Big Four English Football clubs (Manchester United, Liverpool, Arsenal and Chelsea) according to the Premier League results and to their UEFA coefficients between 1993 and 2013, featuring bivariate rankings with some missing ranking positions.

**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.

**The univariate ISR model**

Rank data arise when judges or observers are asked to rank several objects \(O_1, \ldots, O_m\) according to a given order of preference. The resulting ranking can be designated by its **ordering representation** \(x = (x^1, \ldots, x^m) \in P_m\) which signifies that Object \(O_x^h\) is the \(h\)th \((h = 1, \ldots, m)\), where \(P_m\) is the set of the permutations of the first \(m\) integers, or by its **ranking representation** \(x^{-1} = (x^{-1}_1, \ldots, x^{-1}_m)\), which contains the ranks assigned to the objects and means that \(O_x^i\) in the \(i\)th \((i = 1, \ldots, m)\). In the **Rankcluster** package, the ranking representation is used, but users working with the ordering representation can use a function implemented in the package in order to convert their data from one representation to the other.

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 insertion sort because of its optimal-ity properties (minimum number of paired comparisons), Biernacki and Jacques (2013) stated the following so-called ISR model:

\[
p(x; \mu, \pi) = \frac{1}{m!} \sum_{y \in P_m} p(x|y; \mu, \pi) = \frac{1}{m!} \sum_{y \in P_m} \pi^G(x,y,\mu)(1 - \pi)^{A(x,y) - G(x,y,\mu)},
\]

where

- \(\mu \in P_m\), the modal ranking, is a **location parameter**. Its **opposite ranking** \(\hat{\mu} = \mu \circ \hat{\varepsilon}\) with \(\hat{\varepsilon} = (m, \ldots, 1)\) is the rank of smallest probability,
- \(\pi \in [\frac{1}{2}, 1]\) is the probability of good paired comparison in the sorting algorithm (where good means in accordance with the modal ranking \(\mu\)). It is a **scale parameter**: the distribution is uniform when \(\pi = \frac{1}{2}\) and the mode \(\mu\) of the distribution is uniformly more pronounced as \(\pi\) grows, being a Dirac function in \(\mu\) when \(\pi = 1\),
- the sum over \(y \in P_m\) corresponds to all the possible initial presentation orders of the objects to be ranked (with identical prior probabilities equal to \(\frac{1}{m!}\) because they are unknown),
- \(G(x,y,\mu)\) is 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 precise definitions of \(G(x,y,\mu)\) and \(A(x,y)\) can be found in Biernacki and Jacques (2013).

**Mixture of multivariate ISR**

Now redefine \(x = (x^1, \ldots, x^l) \in \mathcal{P}_{m_1} \times \cdots \times \mathcal{P}_{m_l}\) as a **multivariate rank**, in which \(x^j = (x^{j1}, \ldots, x^{jm_j})\) is a rank of \(m_j\) objects \((1 \leq j \leq l)\).

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 \(J\) components \(x^j, \ldots, x^l\) of the multivariate rank datum \(x\) are assumed to be sampled from independent ISR distributions with corresponding modal rankings \(\mu^k_1, \ldots, \mu^k_J\) (each \(\mu^k_j \in P_{m_j}\) and good paired comparison probabilities \(\pi^k_1, \ldots, \pi^k_J \in [\frac{1}{2}, 1]\).

The unconditional probability of a rank \(x\) is then

\[
p(x; \theta) = \sum_{k=1}^{K} p_k \prod_{j=1}^{J} p(x^j; \mu^k_j, \pi^k_j),
\]
where $\theta = (\pi_{jk}^k, \mu_{jk}^k, p_k^k)_{k=1, \ldots, K, j=1, \ldots, J}$ and $p(x^j; \mu_{jk}^k, \pi_{jk}^k)$ is defined by (22.2.1).

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 this and is able to work with partial rankings whatever the positions of the missing data (see details in Jacques and Biernacki (2012)).

### Estimation algorithm

Parameter estimation is performed by maximum likelihood. However, maximum likelihood estimation is not straightforward since the model contains some missing data: the cluster memberships $z_i$ of the observations, the presentation orders $y_i$ (where $y_i = (y_{ij}^1, \ldots, y_{ij}^J)$ is the presentation order for the $i$th observation) and the unobserved ranking positions, denoted by $\hat{x}_i$ (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 by iterating an estimation (E) step, in which the conditional expectation of the completed-data log-likelihood is computed, and a maximization (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 $m$ ($m \leq 7$) 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 Stochastic EM (SEM) Gibbs approach is then proposed in Jacques and Biernacki (2012) to overcome these problems.

The fundamental idea of the SEM-Gibbs 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 (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 one is 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.

### Existing R packages for ranking data

To the best of our knowledge, there are only two packages dedicated to the analysis of ranking data, available on the CRAN website, but their functionalities are significantly limited in comparison to our package Rankcluster, as we discuss now:

- The `pmr` package (Lee and Yu, 2013) provides some descriptive statistics and modeling tools using classical rank data models for full univariate ranking data: Luce models, distance-based models, and rank-ordered logit (see Marden (1995) for a description of these models). Visualization of ranking data using polytopes is also available for less than four objects to rank ($m \leq 4$).
- The `RMallow` package (Gregory, 2012) performs clustering of univariate ranking data using mixtures of distance-based models (Murphy and Martin, 2003).

Rankcluster proposes modeling and clustering tools on the basis of the mixture of multivariate ISR presented in the previous section. Comparing to the existing packages, Rankcluster is the only package taking into account multivariate and partial ranking data.

### Overview of the Rankcluster functions

This section presents briefly the functions of the Rankcluster package. For more details, refer to the help of the functions.

**The main function:** `rankclust()`

The `rankclust()` function performs modeling or cluster analysis of a dataset of rankings. For cluster analysis, the number of clusters can be specified by the user, or selected by the Bayesian Information Criterion (BIC) (Schwarz, 1978), or the Integrated Completed Likelihood (ICL) (Biernacki et al., 2000) in a list of possible numbers provided by the user. By default, the number of clusters is equal to one,
and consequently a single homogeneous multivariate ISR model is fitted to the data. The main inputs and outputs are described below.

This function has only one mandatory argument, data, which is a matrix composed of the \( n \) observed ranks in their ranking representation. For univariate rankings the number of columns of data is \( m \) (default value of argument \( m \)). For multivariate rankings, data has \( m_1 + \ldots + m_J \) 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 \( n \) must be filled with the vector of size \((m_1, \ldots, m_J)\).

Several arguments allow the different tuning parameters used in the SEM-Gibbs estimation to be specified. Refer to Jacques and Biernacki (2012) and to help(rankclust) for more details (section 22.5 also gives some examples). The option \( \text{run} \) allows the number of initializations of the SEM-Gibbs algorithm to be set (1 by default). In the case of multiple initializations, the best solution according to the approximated log-likelihood is retained.

The rankclust() function returns an instance of the "ResultTab" class. It contains 5 slots, including results which is a list containing all \( k = 1, \ldots, K \) classes, summarized in 18 slots, among which the main ones are:

- **proportion**: a \( K \)-vector of proportions \( p_1, \ldots, p_K \).
- **pi**: a \( K \times J \)-matrix composed of the scale parameters \( \pi_{k,j} \) (1 \( \leq k \leq K \) and 1 \( \leq j \leq J \)).
- **mu**: a matrix with \( K \) lines and \( m_1 + \ldots + m_J \) columns in which line \( k \) is composed of the location parameters \( \mu_{1,k}, \ldots, \mu_{J,k} \) of cluster \( k \).
- **bic, icl**: values of the log-likelihood, BIC and ICL, respectively.
- **tik**: an \( n \times K \) matrix containing the estimates of the conditional probabilities of belonging to each cluster for the observed ranks.
- **partition**: an \( n \)-vector containing the partition estimate resulting from the clustering.
- **partialRank**: a matrix containing the full rankings, estimated using the within cluster ISR parameters when the ranking is partial.
- **distanceProp, distancePi, distanceMu**: distances between the final estimate and the current value at each iteration of the SEM-Gibbs algorithm (except the burn-in phase) for respectively: proportions \( p_k \), scale parameters \( \pi_{k,j} \), location parameters \( \mu_{j,k} \).
- **distanceZ**: a vector of size \( \text{qsem-Bsem} \) containing the Rand index (Rand, 1971) between the final estimated partition and the current value of each iteration of the SEM-Gibbs algorithm (except the burning phase).

If \( \text{res} \) is the name assigned to the result of rankclust(), each slot can be obtained by \( \text{res[k]}[@\text{slotname}] \), where \( k \) is the cluster index and \( \text{slotname} \) is the name of the selected slot (proportion, pi ...). For the slots bic and icl, \( \text{res["slotname"]} \) returns a vector of size \( K \) containing the values of the slot for each cluster index.

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 \( x \circ x^{-1} = x^{-1} \circ x \).
- **distCayley(), distHamming(), distKendall(), distSpearman()**: compute usual distances between rankings (Marden, 1995) for either ranking or ordering representation.
- **frequency()**: transforms a raw dataset composed of a matrix of ranks (one rank per line, with possibly equal lines if the same rank is observed several times) into a matrix rank/frequency containing in line each different observed ranks and one additional last column with the frequency of observation of these ranks. Conversely, **unfrequency()** transforms a rank/frequency dataset in a raw dataset, as requested in input argument of rankclust().
- **kh2()**: performs a chi-squared goodness-of-fit tests and returns the \( p \)-value of the test (See Biernacki and Jacques (2013) for details).
- **kullback()**: estimates the Kullback-Leibler divergence between two ISR models.
- **simulISR()**: simulates a univariate and unimodal dataset of full rankings according to the ISR model.
- **probability()**: computes the probability of a ranking (multivariate or not) according to the ISR model.
Rankcluster through examples

This section illustrates the use of the rankclust() function on two real datasets. The first one, words, is a well-known dataset in ranking studies, 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.

The words dataset

This dataset 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.

![BIC value on the words dataset](image)

**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 and then loading the words dataset:

```r
install.packages("Rankcluster")
library(Rankcluster)
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 burn-in phase of 100 iterations (Bsem). For likelihood approximation the numbers of iterations (Ql and Bl) have been divided by two. Option maxTry=20 allows the estimation algorithm to be restarted up to 20 times if one cluster becomes empty (which frequently occurs 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 per number of clusters). The reader who wants to test more quickly the package can reduced the number of runs, using for instance:

```r
res=rankclust(words$data, m=words$m, K=1:5, Qsem=200, Bsem=20, Ql=200, Bl=20,
+ maxTry=20)
```

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

The parameter estimates for K = 3 are given below for proportions $p_k$, scales $\pi_k$ and modes $\mu_k$:

```r
res[3]@proportion
[1] 0.3061224 0.4918367 0.2020408
res[3]@pi
dim 1
```
The word Thought is most associated with Idea for all clusters. Looking at 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 in this cluster are probably literary-minded students who rank the word Dream just after Thought. Students in 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 intermediate 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 (\( \pi_3 \simeq 0.86 \)).

The big4 dataset

In the two last decades, English football has been dominated by four clubs, forming the “Big Four”: Manchester United, Liverpool, Arsenal and Chelsea. In this example, we analyse the rankings of these four teams in the English championship (Premier League) and their rankings according to the UEFA coefficient, a 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, available in the package, is composed of Premier League rankings and UEFA rankings from 1993 to 2013, in ranking notation (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 ties 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)
```

Then, the number of clusters is estimated with the BIC criterion. For this, clustering for 1 to 3 clusters is performed with the rankclust() function. It should be noted that, for 3 clusters, the algorithm has to be launched several times since it often converges to a solution with one empty cluster. The values of the BIC criterion are plotted on Figure 2, and tend to select two groups (which confirms that with 3 clusters the estimation algorithm often converges to one empty cluster).

![BIC value on the big4 dataset](image)

**Figure 2:** Values of the BIC criterion with mixture of ISR for the big4 dataset.
The clustering with $K = 2$ is obtained in about 25 seconds on a laptop 2.80GHz CPU:

```r
R> res = rankclust(big4$data, m=big4$m, K=2, Qsem=1000, Bsem=100, QL=500, BL=50,
+      maxTry=20, run=5)
```

The printed outputs for $K = 2$ are given below: value of the log-likelihood ($ll$), values of BIC and ICL criteria, estimate of the proportions $p_k$'s, the modes $\mu_k$'s, the scales $\pi_k$'s, the estimated partition and finally the conditional probability that the observations belong to each cluster ($t_{ik}$).

```r
R> res[2]
```

```
# The R Journal Vol. 6/1, June 2014 ISSN 2073-4859
```

The estimated clustering exhibits two groups, the second one being larger than the first one ($p_1 \approx 0.39$ and $p_2 \approx 0.61$). The values of the modes for each cluster and dimension lead 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 Chelsea (club 4). Indeed, in the first cluster Chelsea is the last in both rankings, but it is in second position in the second cluster. Moreover, in 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 can be explained by the acquisition of the club by Russian investor Roman Abramovich in 2003, who brought great players to the club.
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
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. Note that the full list of the cluster members with their probabilities and entropies 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 representative rankings of the cluster correspond to rankings after 2003. Similarly, the four observations whose membership in cluster 2 is the most questionable correspond to observations before 2003. This confirms the previous analysis indicating that cluster 2 is due to the rise of Chelsea.

<table>
<thead>
<tr>
<th>Year</th>
<th>UEFA</th>
<th>Prem. League</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>2011</td>
<td>(1,3,4,2)</td>
<td>(1,4,3,2)</td>
<td>2.367e-04</td>
</tr>
<tr>
<td>2008</td>
<td>(4,2,3,1)</td>
<td>(1,4,3,2)</td>
<td>6.862e-05</td>
</tr>
<tr>
<td>2013</td>
<td>(2,4,3,1)</td>
<td>(1,4,3,2)</td>
<td>3.529e-05</td>
</tr>
<tr>
<td>2009</td>
<td>(3,2,4,1)</td>
<td>(1,2,4,3)</td>
<td>3.097e-05</td>
</tr>
<tr>
<td>2005</td>
<td>(1,2,3,4)</td>
<td>(3,4,2,1)</td>
<td>2.151e-05</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Year</th>
<th>UEFA</th>
<th>Prem. League</th>
<th>Entropy</th>
</tr>
</thead>
<tbody>
<tr>
<td>2002</td>
<td>(1,4,2,3)</td>
<td>(3,2,1,4)</td>
<td>0.6755106</td>
</tr>
<tr>
<td>1993</td>
<td>(1,2,3,4)</td>
<td>(1,2,3,4)</td>
<td>0.6599892</td>
</tr>
<tr>
<td>2000</td>
<td>(1,4,3,2)</td>
<td>(1,3,2,4)</td>
<td>0.6474507</td>
</tr>
<tr>
<td>1997</td>
<td>(1,2,3,4)</td>
<td>(1,3,2,4)</td>
<td>0.6403972</td>
</tr>
<tr>
<td>2010</td>
<td>(1,0,0,4)</td>
<td>(2,4,3,1)</td>
<td>0.4613709</td>
</tr>
</tbody>
</table>

Table 1: Rankings with the highest entropies and probabilities in the second cluster.
of the mode 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 scale estimate is for cluster 2 about 0.002 for both the UEFA coefficients and the Premier League rankings. Note that these variabilities are relatively small, due to the low overlapping of the two clusters (scale coefficients are quite high). In a similar way, the slot distance illustrates the convergence of the SEM-Gibbs algorithm by given the Rand index (Rand, 1971) 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.

Conclusion

Rankcluster is the first R package dedicated to ranking data that allows modeling and cluster analysis for multivariate partial ranking data. Available on the CRAN website, this package is simple to use with its main function, rankclust(), having only one mandatory argument, the ranking dataset. By default a single homogeneous multivariate ISR model is fitted to the data, and specifying 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 illustrates the possibilities of the package, and also constitutes a user guide for practitioners.

Bibliography


