Advances in Data Analysis and Classification
https://doi.org/10.1007/s11634-025-00626-9

REGULAR ARTICLE

Particle swarm optimization for preference rankings

Maurizio Romano[1] · **Claudio Conversano**[1] · **Roberta Siciliano**[2] · **Antonio D'Ambrosio**[3]

Received: 1 March 2024 / Revised: 2 January 2025 / Accepted: 27 January 2025

Abstract
Preference learning, or the analysis of preference rankings, is gaining more and more importance in various scientific disciplines. Preference learning methods allow predicting preferences on a set of alternatives. The ingredients are a pool of evaluators and a set of objects or items to be ranked in order of preference. The rank aggregation problem must be solved in order to aggregate preferences or rankings with the aim to find a consensus or collective decision. Branch-and-bound-like procedures are usable up to problems involving a relatively small number of objects, say less than 200. When the number of items becomes very large, the rank aggregation problem becomes increasingly difficult to approach so that it is universally recognized as an NP-hard problem. Several heuristic methods have been proposed to provide increasingly accurate solutions. These assume the Kemeny axiomatic approach that better deals with tied rankings. In this paper, we adopt a strategy based on Particle Swarm Optimization by adapting procedures born to solve optimization problems in continuous spaces to discrete combinatorial optimization problems. A simulation study shows the performance of the proposed algorithm in a controlled environment. A benchmarking

✉ Maurizio Romano
romano.maurizio@unica.it

Claudio Conversano
conversa@unica.it

Roberta Siciliano
roberta@unina.it

Antonio D'Ambrosio
antdambr@unina.it

[1] Department of Business and Economic, University of Cagliari, Cagliari, Italy

[2] Department of Electrical and Information Technology, University of Naples Federico II, Napoli, Italy

[3] Department of Economics and Statistics, University of Naples Federico II, Napoli, Italy

Published online: 17 February 2025

complex data set and two real world data sets with large number of items are considered. As a result, the proposed algorithm provides significant savings in computational time and comparable accuracy with respect to other recent algorithms.

Keywords Preference learning · Kemeny problem · Tied rankings · Heuristics · Particle swarm optimization

Mathematics Subject Classification 62-08 · 90-08 · 68W25

1 Introduction

Preference learning concerns learning from observed preference information (Fürnkranz and Hüllermeier 2011): it is a modern term that includes the historical topic of preference rankings analysis (Marden 1996). The ingredients of preference learning are always a pool of judges (or assessors) that must evaluate a set of items (or objects). The concept of judges and items has evolved over time. Depending on the scope of application and study, judges can be people, evaluation agencies, users of web platforms while items can be perceptions, services, universities, genes, and so on (see, for example, Birlutiu et al. 2010; Csató and Tóth 2020; D'Ambrosio et al. 2019; Vitelli et al. 2023).

In the context of the analysis of preference rankings, depending on whether a set of covariates is available or not, most statistical methods and models, supervised or unsupervised, parametric and non-parametric, can be used (Murphy and Martin 2003; Lee and Yu 2010; Grbovic et al. 2013; D'Ambrosio and Heiser 2016; Mollica and Tardella 2017; Müllensiefen et al. 2018; Vitelli et al. 2018; Plaia and Sciandra 2019; D'Ambrosio and Heiser 2019; Patil and Gupta 2023). Nevertheless, the most important step is to determine which ranking of a set of items best summarizes all rankings expressed by a pool of judges on the same items. Finding such ranking is a very old problem (de Borda 1781; de Condorcet 1785), in the literature, depending on the scientific field, it is known by several names, such as social choice problem, rank aggregation problem, Kemeny problem, median ranking problem, central ranking problem (see, among others, Cook and Kress 1990; Bossert and Storcken 1992; Zhang et al. 2012; Emond and Mason 2002; Crispino and Antoniano-Villalobos 2022).

The ways of aggregating preferences depend on the definition of formal axiomatic statements that define both the measures of agreement between rankings and the geometric space in which the problem is to be defined (Kemeny and Snell 1962; Cook and Saipe 1976; Cook et al. 1986, 2007; Biernacki and Jacques 2013). We frame ourselves in the axiomatic framework of Kemeny and Snell (1962), which is based on the definition of a rigorous and widely accepted set of axioms that any distance defined on preference rankings should respect, culminating in the definition of the Kemeny distance that has been proved to satisfy all the Kemeny and Snell's axioms.

Over the years it has been recognized as the rank aggregation method that better deals with both full and tied rankings: the Kemeny distance is unequivocally related with the well-known Kendall's distance in the case of the restricted space of full rankings (Heiser and D'Ambrosio 2013). It is completely defined in the permutation

polytope, which is the convex hull of the preference rankings with n items included in $\mathbb{R}^n$, which has become the geometric reference space of preference rankings (Thompson 1993; Marden 1996; Heiser and D'Ambrosio 2013; Alvo and Yu 2014). Moreover, it is a rank aggregation method stable and neutral (Young and Levenglick 1978; Young 1995).

There are other widely used distances for this problem (Spearman distance, Spearman's footrule, Kendall distance, Hamming distance, and so on). Nevertheless, most of these metrics are defined and used considering preference rankings only as full rankings and, in general, in the category of distance-based models (Diaconis 1988). When dealing with tied rankings, the results of the rank aggregation process can be inconsistent. For example, Emond and Mason (2002) showed that Spearman's distance is not independent of the so-called irrelevant alternatives, while Kendall's distance violates the triangular inequality. For an exhaustive discussion of these distances, we refer to Marden (1996). Especially when the number of items to be ranked is large, we always assume that the universe of rankings includes all possible tied rankings. In other words, we assume that tied rankings are not indifference declarations, but they are 'positive statements of agreement' (Emond and Mason 2002).

The rank aggregation problem is known to be an NP-hard problem. For this reason, when the number of items to be ranked is large, several heuristic solutions have been proposed over the years (Dwork et al. 2001; Amodio et al. 2016; Mandhani and Meila 2009; D'Ambrosio et al. 2017; Aledo et al. 2017, 2018; Badal and Das 2018; Aledo et al. 2021; Acampora et al. 2021; Yoo and Escobedo 2021; Akbari and Escobedo 2023a, b; D'Ambrosio et al. 2015). Most of these proposals either deal with full rankings or follow other axiomatic frameworks.

In this paper, we propose a heuristic algorithm called Particle Swarm Optimization for Preference Rankings (PSOPR) able to find extremely accurate solutions to the rank aggregation problem when the number of items is really large. As a matter of fact, Particle Swarm Optimization was born to optimize continuous functions which is not the case of preference rankings. We provide an adapted algorithm to deal with a very complex combinatorial optimization problem in the discrete domain. To the best of our knowledge, particle swarm optimization has never been proposed for solving such types of problems. Our results are compared in a simulation study with the proposals that were introduced to deal with rank aggregation by considering all kinds of rankings (full, tied, partial). We also show a couple of examples of preference rankings analysis using real world data sets. A final application outlines the usefulness of our proposal in a more general context of data analysis, such as clustering.

2 The Kemeny axiomatic framework

Rank data can be expressed through n dimensional rank vectors (or simply rankings) in which, once the positions of the items are fixed in any arbitrary order, the i-th element of the vector identifies the rank of the related item with respect to the others (Marden 1996). Rank data can also be expressed through orderings, when the objects are placed in order from the best to the worst. When an assessor cannot assign a clear position of preference for one item over another (or others), then we refer to a tied ranking. When

an assessor expresses a ranking by clearly assigning a position to each item, then we refer to full rankings. When instead in a rank vector some items are not ranked at all, then we refer to partial rankings.

Let X and Y be two rankings of n objects. As in the τ_b correlation coefficient of Kendall (1938), the two rankings can be represented into the so-called *score matrices* $\mathbf{X}$ and $\mathbf{Y}$, which are skew-symmetric matrices of generic elements x_{ij} and y_{ij} respectively defined as:

$$x_{ij}\ (y_{ij}) = \begin{cases} 1 & \text{if object } i \text{ is ranked ahead of object } j \\ -1 & \text{if object } i \text{ is ranked behind object } j \\ 0 & \text{if the objects are tied, or if } i = j \end{cases} \tag{1}$$

Kemeny and Snell (1962) defined a distance between two rankings as:

$$d(X,\ Y) = \frac{1}{2} \sum_{i,\ j=1}^{n} \left| x_{ij} - y_{ij} \right|. \tag{2}$$

Such a distance, known as the Kenemy distance, satisfies the set of desirable axioms that a distance for rankings should have:

1. It must be a metric;
2. It must be invariant under addition or subtraction of equally ranked first and/or last objects;
3. It must be invariant under the same permutation of all objects;
4. Its minimum positive value is equal to one.

Let $\mathcal{H}^n$ be the universe of rankings with n objects. Good (1975) showed that

$$\left| \mathcal{H}^n \right| = \sum_{r=0}^{n} r! \left\{ {n \atop r} \right\}, \tag{3}$$

where $|\mathcal{H}^n|$ denotes the cardinal of $\mathcal{H}^n$, and $\{ {n \atop r} \}$ indicates the ways to partition a set of n objects into r non-empty subsets, sometimes called 'buckets' (Stirling number of the second kind).

Let $A_1, \ldots, A_m$ be a set of rankings of n items produced by m assessors. The median ranking $\hat{S}$ is defined as that ranking, or those rankings, that minimizes the sum of distances:

$$\hat{S} = \underset{S \in H^n}{arg\,min} \sum_{k=1}^{m} d(A_k,\ S). \tag{4}$$

It is clear that finding the median ranking becomes computationally intractable as early as a few dozen items ($|\mathcal{H}^{20}| = 2.677688e + 21$).

Emond and Mason (2002) proposed a branch-and-bound algorithm able to find the exact solution(s) in a reasonable amount of time till $n \leq 30$. They also defined a new rank correlation coefficient, τ_X, showing that it is related to the Kemeny distance:

$$\tau_X(X, Y) = 1 - 2\frac{d(X, Y)}{n(n-1)}. \tag{5}$$

Moreover, Emond and Mason (2002) slightly modified the score matrices in order to better deal with tied rankings. Let x_{ij} the generic elements of the score matrix $\mathbf{X}$ related to the ranking X: they assumed $x_{ij} = 1$ if either the i-th object is preferred to the j-th item or if they are in a tie. Thus, $\tau_X(X, Y)$ has been formally defined as:

$$\tau_X(X, Y) = \sum_{i,j=1}^{n} \frac{x_{ij}\, y_{ij}}{n(n-1)}. \tag{6}$$

$\tau_X(X, Y)$ provides a clear interpretation of the rank aggregation solution, since

$$\hat{S} = \underset{S \in H^n}{arg\ min} \sum_{k=1}^{m} d(A_k, S) = \underset{S \in H^n}{arg\ max} \frac{1}{mn(n-1)} \sum_{i,j=1}^{n} c_{ij} s_{ij}, \tag{7}$$

where c_{ij} are the elements of the so-called *combined input matrix* $\mathbf{C}$, which is the sum of the m score matrices of the sample of assessors, namely $\mathbf{C} = \sum_{j=1}^{m} \mathbf{A}_j$, and s_{ij} are the elements of the score matrix of the consensus ranking $\hat{S}$.

It means that, in comparing two solutions for the same problem, we say that it is better the one achieving the larger averaged τ_X because it corresponds to the one with the lower sum of the distances.

3 Particle swarm optimization (PSO)

Firstly introduced by Kennedy and Eberhart (1995), Particle Swarm Optimization (PSO) is a powerful heuristic optimization algorithm that tries to replicate bird flocks/fish schools behaviours to iteratively improving a candidate solution according to a loss function. As described by Meissner et al. (2006), the idea of PSO is to have a swarm of particles "flying" through a multidimensional search space, looking for the global optimum. By exchanging information the particles can influence each others' movements. Each particle retains an individual (or "cognitive") memory of the best position it has visited, as well as a global (or "social") memory of the best position visited by all particles in the swarm. A particle calculates its next position based on a combination of its last movement vector, the individual and global memories, and a random component.

PSO popularity immediately starts rising while considering that it performed remarkably well within optimization task no matter the huge dimensionality of the considered problem (Poli et al. 2007). Thus, PSO has been successfully applied to many fields and research topics such as, for instance: healthcare image analysis, financial

forecasting, environmental sciences (Banks et al. 2008), industrial engineering (Zhang et al. 2015), and some more general purpose aspects like data clustering (Alam et al. 2014).

Moreover, considering both PSO performance and versatility, researchers have shown interest to further exploring new variants and hybridizations for real world applications. Forby, many variants of PSO have been proposed in literature (see for example Imran et al. (2013) for an overview). For more details about the general usability see the systematic review proposed by Gad (2022).

In the next Section, we propose a new implementation of PSO suitable for Preference Rankings data.

4 The PSO-based algorithm for preference rankings

Recalling that the Particle Swarm Optimization works on a continuous space, the proposed PSOPR algorithm (see Algorithm 1) builds upon the classical implementation of PSO from the R package "pso" (Claus 2022) making it suitable for discrete data. Thus, it allows us to find the consensus ranking of some preference data if the parameters are properly configured. PSOPR utilizes the default values of the input parameters specified in the original implementation of pso. More clarification on this matters and details on the impact of each parameter in the algorithm behaviour will be discussed at the end of this section.

In the following, PSOPR is formally proposed as an heuristic for a $m \times n$ matrix $\mathbf{A}$ composed by m judges expressing preferences for n items. Moreover, whilst considering the eventuality that all the same preference rankings are represented only one time, a frequency weight vector W_k is defined. Considering $\lfloor \cdot \rceil = \text{round}(\cdot)$, likewise the standard implementation, we have $n_p = \lfloor 10 + 2\sqrt{n} \rceil$ particles representing a swarm (i.e. vectors of n positive integers each) $P \subset (\mathbb{Z}^+)^n \in [\mathbf{b_l}; \mathbf{b_u}]$ s.t.:

$$\mathbf{b_l} = (\underbrace{1, \quad 1, \quad \cdots, \quad 1}_{n \text{ elements}}) \qquad \mathbf{b_u} = (\underbrace{n, \quad n, \quad \cdots, \quad n}_{n \text{ elements}}),$$

where $\mathbb{Z}^+$ is the set of all positive integers, and $\mathbf{b_l}$ ($\mathbf{b_u}$) is the lower (upper) bound vector used to represent the range of values (i.e. rank) for each of the n items. Differently from the original implementation, the type of data that PSOPR has to deal with are rankings. Thus, each dimension (i.e. each value of the bounds) is always fixed because, by definition of ranking with ties allowed, each item rank can vary in the following range $[1; n]$. Hence, considering $\|\cdot\|$ as the Euclidean norm, the swarm' breadth (i.e. the search space diameter) is $d = \|\mathbf{b_u} - \mathbf{b_l}\|$.

Therefore, each particle $\mathbf{p}_i \in P$ with $i = 1, \ldots, n_p$ is composed by three components: position $\mathbf{p_i}.\mathbf{pos}$, velocity $\mathbf{p}_i.v$, and the personal (i.e. individual) best solution found by the particle $\mathbf{p_i}.\mathbf{pbest}$.

The position $\mathbf{p_i}.\mathbf{pos}$, represents a possible consensus ranking whilst each value of the n dimensions of the particle is the rank of the assigned item. It is worth to notice that there could be ex aequo. The velocity $\mathbf{p}_i.v$ is a parameter used to compute how far a certain particle should move from the previous $\mathbf{p_i}.\mathbf{pos}_t$ to the next one $\mathbf{p_i}.\mathbf{pos}_{t+1}$,

whilst the inertia $\omega = \dfrac{1 + 2\ln(2)}{8}$ defines the amount of velocity to keep at each update. Considering that each particle "remembers" the personal best solution found, the final solution estimated in the swarm is the global best $\mathbf{p_{best}} = \max(\mathbf{p_i}.\mathbf{pbest}) \quad \forall \mathbf{p}_i \in P$. Despite of the availability of different options to evaluate the goodness of the estimated solution, we consider the coefficient τ_X (Eq. 6)

The first phase of the algorithm (*Initialization*) consists of initializing all the particles. The position and velocity are randomly assigned by sampling with replacement from a uniform distribution of integers $U\{\min, \max\}$, s.t.:

$$\mathbf{p_i}.\mathbf{pos} \sim U\{1, n\} \quad \mathbf{p}_i.v \sim U\{-|n-1|, |n-1|\}. \tag{8}$$

Next, the algorithm starts the *Update* phase. Each particle starts moving to a new position by first updating the velocity:

$$\mathbf{p}_i.v_{t+1} = \omega \cdot \mathbf{p}_i.v_t + \phi_p \cdot \mathbf{r_p} \cdot (\mathbf{p_i}.\mathbf{pbest}_t - \mathbf{p_i}.\mathbf{pos}_t) + \phi_g \cdot \mathbf{r_g} \cdot (\mathbf{p_{best}}.\mathbf{pbest}_t - \mathbf{p_i}.\mathbf{pos}_t) \tag{9}$$

where $\mathbf{r_p} \sim U(0, 1)$ and $\mathbf{r_g} \sim U(0, 1)$ are two random numbers sampled with replacement from a uniform distribution of reals $U(\min, \max)$. Each $\mathbf{r_p}$ and $\mathbf{r_g}$ represents a stochastic component to compute the direction for the particle new position. They act as weights, making the particle randomly considering more the personal best ($\mathbf{p_i}.\mathbf{pbest}$) position and/or the swarm best ($\mathbf{p_{best}}.\mathbf{pbest}$) position respectively; ϕ_p and ϕ_g are two constants $\left(\phi_p = \phi_g = \dfrac{1 + 2\ln(2)}{2}\right)$, working as weights to a-priori force the particle to quantify how much it follows the personal best and/or the swarm best directions, respectively. Thus, the updated velocity provides the information to move the particle to a new place s.t. $\mathbf{p_i}.\mathbf{pos}_{t+1} = \lfloor \mathbf{p_i}.\mathbf{pos}_t + \mathbf{p_i}.\mathbf{v_{t+1}} \rceil$. Therefore, the updated particle position is evaluated, and both $\mathbf{p_i}.\mathbf{pbest}$ and $\mathbf{p_{best}}$ are also updated according the new values of $\tau_X(\mathbf{p_i}.\mathbf{pos}, \mathbf{A}, W_k)$ computed for each $\mathbf{p}_i \in P$. It is worth noticing that this update step could also be done asynchronously (see for example Zuwairie (2014)). The process is iterated whilst restarting (if the restart criteria is satisfied) until one of the following stopping criteria is met:

1. maximum number of iterations reached (max_j). This count does not reset if the algorithm restarts;
2. maximum number of "stagnate" (i.e. no improvement) iterations (max_s);
3. some particular cases (all-tie solution, $\tau_X(\cdot) = 1$).

To prevent situations where all particles in the swarm are stacked together in the same area (i.e. stagnate, probably a local maximum if not reached the global maxima), a lower bound threshold $\alpha_r = 0.0018$ is specified; α_r is the percentile of the swarm' breadth $d = \|\mathbf{b_u} - \mathbf{b_l}\|$, where $\mathbf{b_l}$ and $\mathbf{b_u}$ are updated at each iteration as follows:

$$\mathbf{b_l} = \min \mathbf{p_i}.\mathbf{pos} \qquad \mathbf{b_u} = \max \mathbf{p_i}.\mathbf{pos} \qquad \forall \mathbf{p}_i \in P \tag{10}$$

Once one of the stopping criteria is met the algorithm terminates, producing in output the estimated solution $\hat{\mathbf{S}} = \mathbf{p_{best}}.\mathbf{pos}$ that is the consensus ranking for the data matrix $\mathbf{A}$, and the associated measure of the adequacy of the solution $\hat{S}_{\tau_X} = \tau_X(\mathbf{p_{best}}.\mathbf{pos}, \mathbf{A}, W_k)$.

As previously mentioned, there are many input parameters that can be specified in Algorithm 1. Considering the fine tuning adopted in the original proposal, the default values available in Claus (2022) have been preserved. An exception to that is α_r, which is equal to zero in the original proposal, but that completely disables the possibility for the algorithm to restart if it stuck into a local optima. Thus, we consider as default a value close to zero ($\alpha_r << 0.01$) that allows the algorithm to restart if needed.

On the other hand, the most important differences with the original proposal are: the sampling is done from a uniform distribution of integers (instead of a uniform distribution of reals); the rounding operations are done during the iterative updates; the proper set of bounds. This permits PSOPR to deal with ranking data and guarantees that the output is, in fact, a ranking.

The other parameters (n_p, ω, ϕ_p, ϕ_g) are common inputs for the usual PSO. Forby, their impact on the algorithm has already been studied (see for example Kennedy and Eberhart (1995); Zuwairie (2014); Gad (2022)). Thus, in short:

- for $n_p \longrightarrow 1$ (i.e. the minimum number of particles), the computation time is drastically reduced, likewise the chances of finding a good solution;
- for $n_p \longrightarrow +\infty$, the computation time exponentially increases, but the search is done in deep, approaching an extensive search;
- $\omega \longrightarrow 0$ impacts relevantly in Eq. 9. In fact, as ω represents the inertia, each particle will not keep any amount of the previous velocity. Forby, after each iterative update, the search start to become more local than expected;
- $\omega \longrightarrow 1$ impacts relevantly in Eq. 9. Each particle will keep all the previous velocity and, instead of slowing down time by time, it accelerates. This changes the behaviour of the algorithm, as the search relies more on exploration rather than of local refinement;
- ϕ_p and ϕ_g play an important role into defining the algorithm behaviour as they largely impact on Eq. 9. If $\phi_p = \phi_g$ (default), there is a balance between exploratory and local search;
- if $\phi_p > \phi_g$ ($\phi_p < \phi_g$), the algorithm accelerates more slowly (quickly) at the global best, as the exploratory (local) search is preferred;
- for $\phi_p \longrightarrow 0$, each particle relies solely on the swarm best identified solution. Therefore, the swarm quickly converges to the first best position identified by one of the particles;
- for $\phi_g \longrightarrow 0$, each particle relies solely on its best identified solution ignoring the swarm. Therefore, the swarm prefers to explore in the area of its personal best.

Algorithm 1 Particle Swarm Optimization for Preference Rankings (PSOPR)

Input:

$\mathbf{A}$; n; m; $W_k = \text{NULL}$; $n_p = \lfloor 10 + 2\sqrt{n} \rfloor$;
$\omega = \frac{1 + 2\ln(2)}{8}$; $\phi_p = \frac{1 + 2\ln(2)}{2}$; $\phi_g = \frac{1 + 2\ln(2)}{2}$;
$max_j = 1000$; $max_s = \infty$; $\alpha_r = 0.0018$;

Phase 1: Initialization
$P = \{\mathbf{p}_1, \ldots, \mathbf{p}_{n_p}\}$
$\mathbf{b_l} = (\underbrace{1, \quad 1, \quad \cdots, \quad 1}_{n \text{ elements}})$
$\mathbf{b_u} = (\underbrace{n, \quad n, \quad \cdots, \quad n}_{n \text{ elements}})$
$d = \|\mathbf{b_u} - \mathbf{b_l}\|$
$\alpha_r = d \cdot \alpha_r$
$\mathbf{p_{best}} = \mathbf{p}_1$
for $i = 1$ **to** n_p **do**
  $\mathbf{p_i}.\mathbf{pos} \sim U\{1, n\}$ **Eq. 8**
  $\mathbf{p_i}.v \sim U\{-|n-1|, |n-1|\}$ **Eq. 8**
  $\mathbf{p_i}.\mathbf{pbest} = \mathbf{p_i}.\mathbf{pos}$
  if $\tau_X(\mathbf{p_i}.\mathbf{pos}, \mathbf{A}, W_k) > \tau_X(\mathbf{p_{best}}.\mathbf{pos}, \mathbf{A}, W_k)$ **then**
    $\mathbf{p_{best}} = \mathbf{p}_i$
  end if
end for

Phase 2: Iterative update
for $j = 1$ **to** max_j **do**
  if A stopping criterion is met then
    STOP
  end if
  for $i = 1$ **to** n_p **do**
    $\mathbf{r_p}, \mathbf{r_g} \sim U(0, 1)$
    $\mathbf{p_i}.v = \omega \cdot \mathbf{p_i}.v + \phi_p \cdot \mathbf{r_p} \cdot (\mathbf{p_i}.\mathbf{pbest} - \mathbf{p_i}.\mathbf{pos}) + \phi_g \cdot \mathbf{r_g} \cdot (\mathbf{p_{best}}.\mathbf{pbest} - \mathbf{p_i}.\mathbf{pos})$
    $\mathbf{p_i}.\mathbf{pos} = \lfloor \mathbf{p_i}.\mathbf{pos} + \mathbf{p_i}.v \rceil$
    if $\tau_X(\mathbf{p_i}.\mathbf{pos}, \mathbf{A}, W_k) > \tau_X(\mathbf{p_i}.\mathbf{pbest}, \mathbf{A}, W_k)$ **then**
      $\mathbf{p_i}.\mathbf{pbest} = \mathbf{p_i}.\mathbf{pos}$
    end if
    if $\tau_X(\mathbf{p_i}.\mathbf{pos}, \mathbf{A}, W_k) > \tau_X(\mathbf{p_{best}}.\mathbf{pos}, \mathbf{A}, W_k)$ **then**
      $\mathbf{p_{best}} = \mathbf{p}_i$
    end if
  end for
  $\mathbf{b_l} = \min \mathbf{p_i}.\mathbf{pos} \quad \forall \mathbf{p}_i \in P$
  $\mathbf{b_u} = \max \mathbf{p_i}.\mathbf{pos} \quad \forall \mathbf{p}_i \in P$
  $d = \|\mathbf{b_u} - \mathbf{b_l}\|$
  if $d < \alpha_r$ **then**
    Restart
  end if
end for

Output: $\{\hat{\mathbf{S}} = \mathbf{p_{best}}.\mathbf{pos}\}$; $\{\hat{S}_{\tau_X} = \tau_X(\mathbf{p_{best}}.\mathbf{pos}, \mathbf{A}, W_k)\}$

5 Benchmarking

To assert the usefulness of PSOPR, it is fundamental to compare it with the most popular algorithms in the field in terms of two main aspects: goodness of the estimated solution (in terms of τ_X), and computational efficiency (in terms of running time expressed in seconds).

Considering the state of the art, our comparisons rely on the `ConsRank R` package (D'Ambrosio 2023), which implements many algorithms for finding the consensus ranking like DECOR, QUICK, and FAST (D'Ambrosio et al. 2015, 2017; Albano and Plaia 2021). These are, to the best of our knowledge, the only proposals based on the Kemeny's axiomatic framework, not limiting the searching space to the universe of only full rankings (namely, $n!$). Nevertheless, the `RankAggSIgFUR R` package (Parker et al. 2023) that implements the SIGFUR algorithm (Badal and Das 2018) is also considered but for the simulation study only, since it limits the searching space to the universe of full rankings only.

In this section, a simulation study is first presented to assess the performance of the proposed PSOPR algorithm in a controlled environment (Sect. 5.1). Secondly, a popular complex and simulated data set used for benchmarking algorithms aimed at finding the consensus ranking is considered (Sect. 5.2). Finally, our empirical investigation is completed in the analysis of two real world and public data sets presenting a large number of items (Sect. 5.3, 5.4).

5.1 Simulation study

We used the `rgmm` function implemented in the `PerMallows R` package (Irurozki et al. 2016) to generate a sample of m permutations from a Generalized Mallows Model $GMM(n, \sigma, \theta)$, where n is the number of items, σ is the central permutation, and θ represents the dispersion parameter. Note that although called a dispersion parameter, in fact θ is a concentration parameter in that the greater its value, the greater the concentration of the rankings in the sample of judges around the consensus ranking. If θ is zero, then each ranking has an equal probability of being the consensus ranking.

Considering the two main aspects needed to compare the algorithms (goodness of the estimated solution and computational efficiency), different sets of data are sampled from GMM with all the combinations of the following parameters:

- $m = 20$,
- $n \in \{100, 250, 500, 1000\}$,
- $\sigma = \{1, \ldots, n\}$,
- $\theta \in \{0.0125, 0.1, 0.8\}$.

As it is possible to notice in Figs. 1 and 2, PSOPR obtains τ_X values at the same levels of the other state of the art algorithms. Up to $n = 100$ items there is not a huge difference in terms of computation time but, as soon as n increases such required time drastically increases. This is especially noticeable on algorithms that instead of computing for a couple of seconds, starts to need more that 24 h thus not being feasible anymore (i.e. DECOR/QUICK with $n \geq 1000$ and FAST/SIGFUR with $n \geq 500$). The performance of the algorithms in terms of averaged τ_x is quite similar. Note how

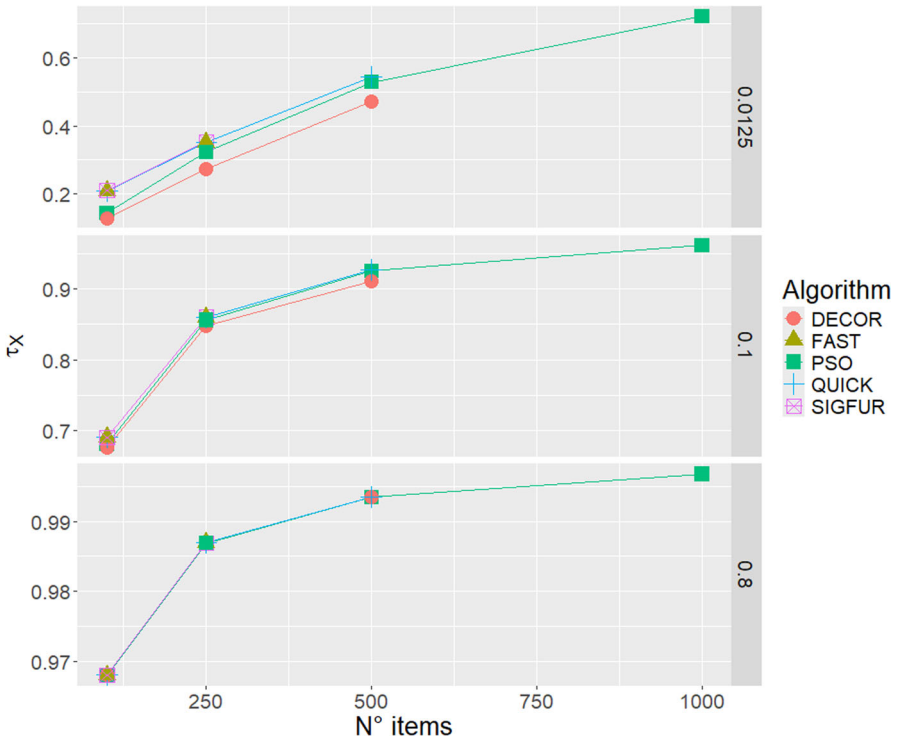

Fig. 1 Algorithms goodness of the best solution found (the higher τ_X, the better) while varying the dispersion parameter ($\theta \in \{0.0125, 0.1, 0.8\}$) and the number of items (n)

the computing time for branch-and-bound-like algorithms strongly depends also on the value of θ: when $\theta = 0.8$ the noise level in sampling rankings is quite low, hence the search space is reduced after a few tens of evaluated branches. The only algorithm whose computing time depends neither on the number of objects nor on θ is PSOPR.

This is explainable by the fact that PSOPR computational time is almost invariant to n whilst depending only on the number of particles n_p. Thus, despite of still being feasible for $n \leq 100$, those results suggest to use PSOPR for data set with $n \geq 250$, especially if $n >> 250$.

Usually, the branch-and-bound algorithm struggle while increasing n. Forby, by construction, this simulation study was focused on assessing the performances while increasing the number of items and varying the dispersion parameter (i.e. lower to stronger consensus).

5.2 EMD data set

The `ConsRank R` package (D'Ambrosio 2023) includes the *EMD* data set (Emond and Mason 2000), described in Table 1. This peculiar data set, despite of not having a huge number of items ($n = 15$), contains many missing (i.e., `NA`) data and many local

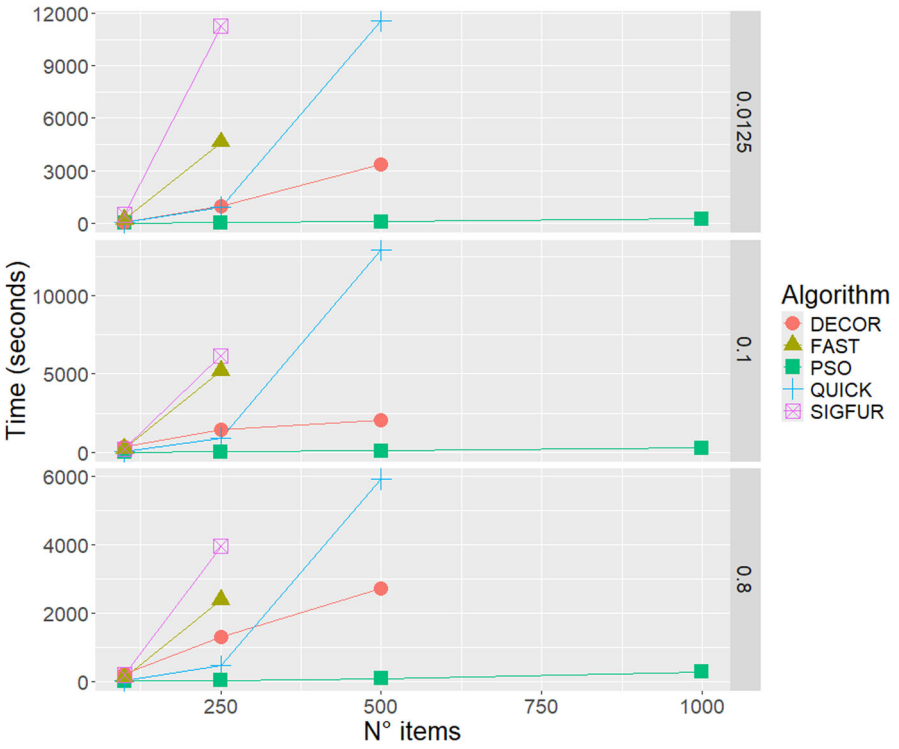

Fig. 2 Algorithms computational time (in seconds) while varying the dispersion parameter ($\theta \in \{0.0125, 0.1, 0.8\}$) and the number of items (n)

maxima that are usually a deadlock for heuristics. Moreover, it is also well known that there are only three global maxima with the same exact $\tau_X = 0.16556$ s.t.:

$$S_1. < \text{D J (E-K) (A-B) I N (C-L) H F G (M-O)} >$$
$$S_2. < \text{D J (E-K) (A-B-N) (C-L) I H F G (M-O)} >$$
$$S_3. < \text{D J (E-K) (B-N) A (C-L) I H F G (M-O)} >$$

where (S_1, S_2, S_3) are the orderings representing the three consensus rankings with the highest value of τ_X. Hence, all the tree well known solutions of the consensus ranking problem for the *EMD* data set are considering the item *D* at first place, item *J* at second place, items *E* and *K* ex-aequo at third place, and so on.

Thus, all those characteristics define the popularity of this data set for benchmarking within the context of algorithms that are not limiting the searching space to the universe of full rankings only (Plaia et al. 2021; Amodio et al. 2016; D'Ambrosio et al. 2017).

As it is possible to notice in Table 2, while running all the algorithms with default parameters, PSOPR and all the state-of-the-art heuristics were able to find at least one global maxima. It is crucial to consider that the computational time for algorithms like PSOPR and DECOR strictly depends on the iteration number, but for others like

Table 1 *EMD* data set as contained in *ConsRank R* package

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	W_k
2	8	4	8	7	1	2	5	2	3	6	7	8	–	–	4
7	8	4	5	7	1	6	5	3	2	7	9	10	11	12	4
11	12	3	11	7	1	4	5	12	2	6	10	11	8	9	4
11	10	4	8	9	1	7	12	2	3	2	6	13	5	14	4
4	7	8	6	13	2	3	12	9	1	5	10	5	11	11	4
6	10	14	5	7	1	8	3	2	3	4	11	13	12	9	4
1	6	4	5	–	1	2	7	3	1	5	2	6	5	5	4
2	9	5	1	4	3	2	7	3	1	8	6	3	4	8	5
3	9	7	1	2	8	13	6	1	10	5	11	9	4	14	5
7	2	–	1	2	10	5	3	9	8	6	7	7	6	4	5
6	1	3	3	6	2	6	5	4	5	1	1	2	1	1	5
7	4	6	1	5	14	10	12	15	3	13	9	8	2	11	5
4	2	9	1	3	12	6	10	13	14	11	9	7	8	5	5
2	4	3	3	11	8	10	9	6	10	5	1	5	7	5	5
8	4	7	2	1	11	4	6	3	12	6	10	13	5	9	7
–	–	5	6	12	9	10	8	2	11	1	4	7	2	3	7
4	3	5	11	12	10	13	7	6	8	2	1	9	9	11	7
–	–	3	1	1	5	5	4	5	2	4	2	6	7	8	7
–	–	4	7	2	10	11	5	8	8	9	1	2	3	6	7
6	6	8	1	1	3	5	1	10	7	2	10	9	4	6	7
9	8	7	6	3	4	–	2	5	1	3	7	6	4	6	7

Table 2 State of the art heuristics performance with the *EMD* dataset

Heuristic	Solution	τ_X	Time (s)
PSOPR	$\{S_1, S_2\}$	0.16556	3.789
DECOR	$\{S_2, S_3\}$	0.16556	26.87
QUICK	$\{S_2\}$	0.16556	0.28
FAST	$\{S_3\}$	0.16556	1.29

QUICK and FAST mostly depends on the number of items (n). In this specific context, where there is a small number of items ($n < 45$, D'Ambrosio et al. (2017)), it is not surprising that QUICK (and FAST as a consequence) runs in an instant. Nevertheless, PSOPR has proved to be 7x times faster than DECOR whilst finding an equivalent number of solutions (one more than QUICK/FAST).

5.3 Global university rankings

A bigger ($n = 1,173$) public data set is now considered. Available at https://preflib.org (Mattei and Walsh 2013), Boehmer and Schaar (2022) provided the *Global University Ranking* data set (Table 3), that contains rankings of universities according to different criteria for year 2015 (i.e. file *00046-00000004.soi*). In this case, differently from the EMD data set, solutions are not known nor published by the authors. Therefore, considering that the Branch and Bound (BB) algorithm can't produce solutions in acceptable time with a huge n, with few obvious exceptions (i.e. all judges with the same rankings), there is no way to know the "real" solution.

As it is possible to observe in Table 4, the only algorithm (in the set of considered ones) able to estimate a solution in an acceptable computational time is the PSOPR. In fact, dealing with a huge number of items is a hard task: both DECOR and QUICK aren't even able to start due to a stack overflow error, whilst FAST (whose elaboration time depends not only on the number of iterations but mostly on the number of items) runs for many hours before interrupting due to a stack overflow error (in facts it relies on executing QUICK 10 times from different starting points). An excerpt of the solution estimated by PSOPR is shown in Table 5 i.e. top 50).

Table 3 First 5 columns of the *Global University Ranking* dataset (year 2015)

Bielefeld Univ	Hebrew Univ. of Jerusalem	Federal Univ. of Santa Maria	Univ. at Buffalo	National Univ. of Còrdoba	.
663	163	393	395	945	.
363	135	962	231	889	.
528	359	971	285	779	.
366	90	968	289	783	.
762	33	978	229	962	.
471	114	942	196	706	.
536	15	964	418	924	.
471	14	961	310	921	.
408	491	491	297	491	.
405	491	491	264	491	.
262	491	491	226	491	.
448	491	491	232	491	.
346	489	489	315	489	.
319	287	395	279	395	.
244	212	395	180	395	.
336	154	395	128	395	.
298	123	395	134	395	.
381	74	384	260	384	.
252	332	376	230	376	.

Table 4 State of the art heuristics performance with the *Global University Ranking* dataset (year 2015)

Heuristic	τ_X	Time (s)
PSOPR	0.4375569	6.579
DECOR	–	–
QUICK	–	–
FAST	–	–

Table 5 Top 50 universities as estimated by PSOPR using the Global University Ranking data set (year 2015)

Rank	University	Rank	University
1	Harvard University	26	University of Minnesota }
2	Stanford University	27	University of British Columbia
3	{Massachusetts Institute of Technology	28	University of Copenhagen
4	University of California}	29	Osaka University
5	University of Cambridge	30	University of Maryland
6	University of Oxford	31	{Boston University
7	{California Institute of Technology	32	Carnegie Mellon University
8	Columbia University}	33	Karolinska Institute}
9	{Princeton University	34	{University of Texas at Austin
10	University of Chicago	35	Utrecht University}
11	Yale University}	36	Vanderbilt University
12	Johns Hopkins University	37	University of Southern California
13	Cornell University	38	{University of Arizona
14	University of Pennsylvania	39	University of Zurich}
15	{University College London	40	Washington University in St. Louis
16	University of Toronto}	41	Brown University
17	Duke University	42	University of Edinburgh
18	University of Michigan	43	{Georgia Institute of Technology
19	University of Tokyo	44	National University of Singapore
20	{Kyoto University	45	Ohio State University}
21	Northwestern University}	46	{University of Geneva
22	{Imperial College London	47	University of Rochester}
23	New York University}	48	Emory University
24	University of North Carolina at Chapel Hill	49	{Leiden University
25	{McGill University	50	University of Florida}

Table 6 First 25 rows of the *Italian Football* dataset

Player	Role	Team	Match played	Goal	Yellow card	.
Danilo	Defender	JUV	37	3	5	.
G. Di Lorenzo	Defender	NAP	37	3	2	.
Kim Min-Jae	Defender	NAP	35	2	5	.
M. Lautaro	Striker	INT	38	21	3	.
K. Kvaratskhelia	Striker	NAP	34	12	1	.
V. Osimhen	Striker	NAP	32	26	4	.
S. Milinkovic-Savic	Midfielder	LAZ	36	9	10	.
F. Anguissa	Midfielder	NAP	36	3	3	.
M. Zaccagni	Striker	LAZ	35	10	9	.
Felipe Anderson	Striker	LAZ	38	9	3	.
R. Leao	Striker	MIL	35	15	5	.
Carlos Augusto	Defender	MON	35	6	4	.
S. Lobotka	Midfielder	NAP	38	1	2	.
A. Romagnoli	Defender	LAZ	34	2	6	.
A. Rabiot	Midfielder	JUV	32	8	9	.
M. Pessina	Midfielder	MON	35	5	6	.
T. Koopmeiners	Midfielder	ATA	33	10	7	.
N. Barella	Midfielder	INT	35	6	6	.
R. Ibanez	Defender	ROM	33	3	9	.
F. Tomori	Defender	MIL	33	1	5	.
Luis Alberto	Midfielder	LAZ	35	6	4	.
Bremer	Defender	JUV	30	4	6	.
Amir Rrahmani	Defender	NAP	29	2	2	.
C. Smalling	Defender	ROM	32	3	7	.
S. Posch	Defender	BOL	30	6	6	.
.	.	.	.	.	.	.

5.4 *Serie A* footballers

Another case study analyzing a public dataset is presented. Therefore, we considered the performances of *Serie A* footballers that played in Italy in years 2022/2023[1]; an excerpt of this data is shown in Table 6. Specifically, we considered defenders (193), midfielders (157) and strikers (121) jointly with the following characteristics: PTS, CR, Goal Difference, Appearances, Starting player, Minutes played, Goals, Shots, Target Shots, penalty goal, Successful dribbles, Assists, Successful Passes, Key Passes, Fouls Committed, Fouls Suffered, Yellow Cards, Red Cards, Stolen Balls, Tackles, and Clean Sheets. Each of those has been considered as a judge who ranks players according to

[1] Available at https://www.kickest.it/it/serie-a/statistiche/giocatori/tabellone/2022-2023.

Table 7 State of the art heuristics performance with the *Italian Football* dataset

Subset (n)	Heuristic	τ_X	Time (s)
Full (471)	PSOPR	0.4426848	250.54
	DECOR	0.4413856	368.65
	QUICK	0.4500606	1616.13
	FAST	0.4506104	8974.75
Defenders only (193)	PSOPR	0.4596055	109.37
	DECOR	0.4561487	98.46
	QUICK	0.4677811	113.94
	FAST	0.4682360	615.7
Midfielders only (157)	PSOPR	0.4817355	97.56
	DECOR	0.4786908	75.99
	QUICK	0.4855035	80.67
	FAST	0.4858924	454.89
Strikers only (121)	PSOPR	0.5039879	56.81
	DECOR	0.5000722	55.05
	QUICK	0.5062574	48.44
	FAST	0.5060934	270.3

such an aspect. Thus, for the Italian Football data set (X), we have 471 players (the items, n) and 21 characteristics (the judges, m).

As it is possible to observe in Table 7, PSOPR has proved to be faster likewise DECOR but consistently provides a better solution. Moreover, the provided solution is very close to both those of QUICK and FAST, with the relevant consideration of being definitely much faster when n is big: $\sim 3\times$ than QUICK and $\sim 17\times$ than FAST (Table 8).

6 Clustering example

As an example of the usefulness of our proposal within a general data analysis framework, we show how rank aggregation can be used in a clustering example. We used the subset of the 157 midfielders of the Italian *Serie A* footballers. We computed the matrix of Kemeny distances among the 21 "judges" and did clustering using the method of k-medoids (Kaufman and Rousseeuw 2008). As it is possible to observe in Table 9, the "optimal" number of clusters has been identified with the GAP statistic (Tibshirani et al. 2001) using the `R` package `cluster` (Maechler et al. 2023). The default method *firstSEmax* has been considered. It looks for the smallest number of clusters (k) such that its value is not more than 1 standard error away from the first local maximum. Results reported in Fig. 3 highlight the suggested choice of using $k = 2$.

The unconditional rank aggregation solution using PSOPR returned a solution with averaged $\tau_X = 0.4817355$. The first cluster is composed of PTS (which is the cluster medoid), CR, Goal Difference, Appearances, Starting player, Minutes played, Goals,

Table 8 Top 25 players comparison as estimated by the state of the art algorithms using the Italian Football dataset (Full)

Rank	PSO	DECOR	QUICK	FAST
1	{G.Di.Lorenzo	{G.Di.Lorenzo	{G.Di.Lorenzo	{G.Di.Lorenzo
2	K.Kvaratskhelia}	K.Kvaratskhelia}	K.Kvaratskhelia}	K.Kvaratskhelia}
3	Felipe.Anderson	Felipe.Anderson	{Danilo	M.Lautaro
4	{F.Anguissa	F.Anguissa	Felipe.Anderson}	{Danilo
5	M.Lautaro	{M.Lautaro	S.Milinkovic.Savic	F.Anguissa
6	S.Milinkovic.Savic}	S.Milinkovic.Savic}	{F.Anguissa	Felipe.Anderson
7	{Danilo	{Danilo	M.Lautaro	S.Milinkovic.Savic}
8	S.Lobotka}	M.Zaccagni	M.Zaccagni}	{M.Pessina
9	M.Zaccagni	S.Lobotka}	{Carlos.Augusto	M.Zaccagni
10	Carlos.Augusto	Carlos.Augusto	M.Pessina	S.Lobotka}
11	M.Pessina	M.Pessina	R.Leao	{A.Rabiot
12	D.Frattesi	D.Frattesi	S.Lobotka}	Carlos.Augusto
13	{L.Pellegrini.ROM	{L.Pellegrini.ROM	{A.Rabiot	D.Frattesi
14	R.Leao}	N.Barella	D.Frattesi	D.Udogie
15	{A.Rabiot	R.Leao}	T.Koopmeiners	H.Theo
16	Kim.Min.Jae	{A.Rabiot	V.Osimhen}	Kim.Min.Jae
17	N.Barella	Kim.Min.Jae	Luis.Alberto	L.Pellegrini.ROM
18	V.Osimhen}	V.Osimhen}	{H.Theo	Luis.Alberto
19	Luis.Alberto	Luis.Alberto	L.Pellegrini.ROM}	N.Barella
20	{P.Ciurria	{D.Udogie	{A.Candreva	P.Ciurria
21	T.Koopmeiners}	P.Ciurria	B.Dia	P.Zielinski
22	{N.Vlasic	T.Koopmeiners}	D.Udogie	R.Leao
23	P.Dybala}	P.Dybala	F.Kostic	T.Koopmeiners
24	{D.Udogie	{H.Calhanoglu	G.Strefezza	V.Osimhen}
25	H.Theo}	H.Theo}	H.Calhanoglu}	F.Kostic

Shots, Target Shots, Successful dribbles, Assists, Successful Passes, Key Passes, Fouls Committed, Fouls Suffered, Yellow Cards, Red Cards, Stolen Balls, Tackles, and Clean Sheets, with within cluster $\tau_X = 0.6160379$. The second cluster is composed of Fouls Committed (which is the cluster medoid), Penalty goals and Yellow Cards, with within cluster $\tau_X = 0.7011269$. Note that we detected three solutions of the rank aggregation problem of the second cluster center, with a few differences at the positions 127, 128 and 129 (first versus second solution, 'M. Brozovic' and 'R.Mandragora' in a tie with rank 127, then 'P. Ciurria' instead of 'R.Mandragora' that precedes 'M. Brozovic' and 'P. Ciurria' both in a tie) and at the positions 99, 100 and 101 (second versus third solution, 'L. Samardzic' and 'N. Matic' in a tie with rank 99 and then 'S. Lobotka' instead of 'N. Matic' that precedes 'S. Lobotka' and 'L. Samardzic' both in a tie). Table 10 shows in the second column the first solution. Note that in any case the top

25 and the worst 26 midfielders would not have changed. The weighed τ_X associated with the cluster solution is equal to 0.6281935, which compared with the unconditional τ_X shows a good gain in terms of overall cluster homogeneity.

7 Concluding remarks

In this paper, a Particle Swarm Optimization-based algorithm for the rank aggregation problem completely defined within the Kemeny's axiomatic approach has been introduced. A comparison with an already proposed differential evolution algorithm

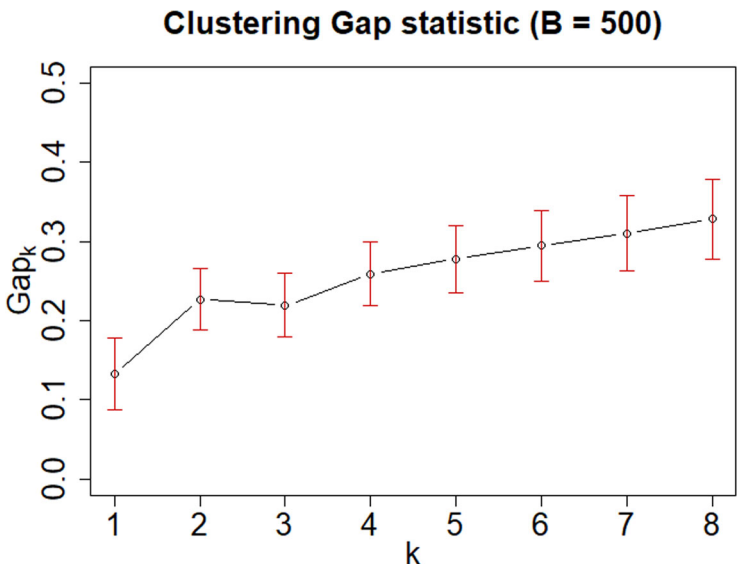

Fig. 3 Gap statistic and standard errors in 500 Monte Carlo bootstrap replications

Table 9 Details for the Gap statistic with 500 Monte Carlo bootstrap samples

k	$\log(W(k))$	$E^*[\log(W(k))]$	$\text{Gap} = E^*[\log(W(k))] - \log(W(k))$	$SE(Gap)$
1	9.128103	9.260826	0.1327236	0.04537412
2	8.834945	9.061786	0.2268410	0.03895869
3	8.720574	8.939883	0.2193091	0.04009425
4	8.575440	8.834534	0.2590947	0.04065714
5	8.462667	8.740984	0.2783164	0.04240186
6	8.356765	8.651727	0.2949624	0.04457111
7	8.250696	8.561134	0.3104377	0.04766944
8	8.139061	8.467409	0.3283475	0.05002222

Table 10 Top 25 and worst 26 midfielders players. The first column refers to the median ranking considering all 21 assessors

Rank	Median ranking	Cluster center 1	Cluster center 2
1	S. Milinkovic-Savic	S. Milinkovic-Savic	S. Vignato
2	F. Anguissa	F. Anguissa	N. Pyyhtia
3	S. Lobotka	S. Lobotka	V. Carboni
4	M. Pessina	A. Rabiot	K. Zedadka
5	D. Frattesi	M. Pessina	E. Darboe
6	{Luis Alberto	{D. Frattesi	D. Samek
7	T. Koopmeiners}	T. Koopmeiners}	A. Barberis
8	{N. Barella	{Luis Alberto	T. Cipot
9	R. Pereyra}	N. Barella}	B. Bayeye
10	A. Rabiot	L. Coulibaly	G. Gaetano
11	L. Pellegrini ROM	R. Pereyra	D. Demme
12	N. Vlasic	{L. Pellegrini ROM	TH. Helgason
13	{H. Calhanoglu	P. Ciurria}	P. Pogba
14	P. Zielinski}	Walace	G. Wijnaldum
15	{A. Candreva	{N. Vlasic	{A. Hrustic
16	F. Kostic	P. Zielinski}	Y. Adli}
17	P. Ciurria}	{H. Calhanoglu	B. Tahirovic
18	Walace	M. de Roon	E. Ilkhan
19	S. Tonali	S. Tonali}	F. Paoletti
20	Brahim Diaz	{A. Candreva	M. Adopo
21	L. Coulibaly	F. Kostic	A. Vranckx
22	{L. Ferguson	L. Ferguson}	D. Crnigoj
23	M. Bourabia}	S. Lovric	{C. Volpato
24	S. Lovric	Brahim Diaz	M. D'Alessandro}
25	N. Dominguez	N. Dominguez	F. Ranocchia
—	—	—	—
132	C. Volpato	S. Iling-Junior	{J. Cuadrado
133	S. Iling-Junior	D. Crnigoj	M. Lopez}
134	D. Crnigoj	B. Tahirovic	I. Bennacer
135	A. Barberis	C. Volpato	N. Barella
136	B. Tahirovic	A. Barberis	L. Henderson
137	T. Cipot	A. Hrustic	Joan Gonzalez
138	G. Gaetano	T. Cipot	Walace
139	{A. Hrustic	{E. Ilkhan	S. Tonali
140	M. D'Alessandro}	M. D'Alessandro}	F. Djuricic
141	TH. Helgason	M. Adopo	D. Frattesi
142	{E. Ilkhan	Gerard Yepes	F. Anguissa

Table 10 (continued)

Rank	Median ranking	Cluster center 1	Cluster center 2
143	P. Pogba}	A. Bianco	{L. Ferguson
144	Y. Adli	{G. Gaetano	M. Locatelli}
145	A. Bianco	TH. Helgason	{A. Blin
146	M. Adopo	Y. Adli}	K. Linetty}
147	A. Vranckx	{F. Paoletti	N. Dominguez
148	Gerard Yepes	P. Pogba}	M. de Roon
149	{D. Demme	A. Vranckx	T. Rincon
150	F. Paoletti	{B. Bayeye	B. Cristante
151	N. Pyyhtia}	D. Demme}	A. Rabiot
152	B. Bayeye	N. Pyyhtia	{F. Bandinelli
153	{S. Vignato	{D. Samek	S. Milinkovic-Savic}
154	V. Carboni}	S. Vignato	S. Amrabat
155	{D. Samek	V. Carboni}	M. Leris
156	K. Zedadka}	E. Darboe	M. Hjulmand
157	E. Darboe	K. Zedadka	L. Coulibaly

shows that the results are extremely accurate, although compared with branch-and-bound-like proposals it does not match its performance in terms of accuracy. Of course, branch-and-bound-like proposals are usable up to problems involving a relatively small number of objects (say, when $n \leq 200$). A couple of examples on a practical application of the rank aggregation problem in data analysis context, specifically also the use in clustering problems, were shown. Compared with solutions proposed through differential evolution algorithms, the performance are averagely better what's more, with significant savings in computational time. The latest have been shown to be consistent not only on real data but also in a simulation study. Thus, the comparisons have also been extended to algorithms that limit the searching space to the universe of only full rankings (i.e. RankAggSIgFUR R package).

Funding Open access funding provided by Università degli Studi di Cagliari within the CRUI-CARE Agreement. Maurizio Romano acknowledge financial support under the National Recovery and Resilience Plan (NRRP), Mission 4 Component 2 Investment 1.5 - Call for tender No.3277 published on December 30, 2021 by the Italian Ministry of University and Research (MUR) funded by the European Union – Next Generation EU. Project Code ECS0000038 – Project Title eINS Ecosystem of Innovation for Next Generation Sardinia – CUP F53C22000430001- Grant Assignment Decree No. 1056 adopted on June 23, 2022 by the MUR. Claudio Conversano acknowledge financial support under the National Recovery and Resilience Plan (NRRP) funded by the European Union – Next Generation EU for the projects eINS Ecosystem of Innovation for Next Generation Sardinia (ECS0000038) and GRINS -Growing Resilient, INclusive and Sustainable (GRINS PE00000018). For this work Roberta Siciliano was supported by the Italian Ministry of Research, under the complementary actions to the NRRP "Fit4MedRob - Fit for Medical Robotics" Grant (PNC0000007). The research work of Antonio D'Ambrosio is supported by the European Union - NextGenerationEU, in the framework of the ECSC - Centro Nazionale HPC, Big Data e Quantum Computing (ECSC E63C22000980007 – CUP CN_00000013).

Declarations

Code availability R Script codes for reproducibility purpose of the PSOPR algorithm is now available on the Github repository https://urlr.me/XrajPG

References

Acampora G, Iorio C, Pandolfo G, Siciliano R, Vitiello A (2021) A memetic algorithm for solving the rank aggregation problem. Algorithms as a Basis of Modern Applied Mathematics, 447–460

Akbari S, Escobedo AR (2023) Approximate condorcet partitioning: Solving large-scale rank aggregation problems. Comput Oper Res 153:106164

Akbari S, Escobedo AR (2023) Beyond kemeny rank aggregation: A parameterizable-penalty framework for robust ranking aggregation with ties. Omega 119:102893

Alam S, Dobbie G, Koh YS, Riddle P, Ur Rehman S (2014) Research on particle swarm optimization based clustering: A systematic review of literature and techniques. Swarm Evol Comput 17:1–13. https://doi.org/10.1016/j.swevo.2014.02.001

Albano A, Plaia A (2021) Element weighted Kemeny distance for ranking data. Electron J Appl Stat Anal 14(1):117–145. https://doi.org/10.1285/i20705948v14n1p117

Aledo JA, Gámez JA, Rosete A (2017) Utopia in the solution of the bucket order problem. Decis Support Syst 97:69–80

Aledo JA, Gámez JA, Rosete A (2018) Approaching rank aggregation problems by using evolution strategies: the case of the optimal bucket order problem. Eur J Oper Res 270(3):982–998

Aledo JA, Gámez JA, Rosete A (2021) A highly scalable algorithm for weak rankings aggregation. Inf Sci 570:144–171

Alvo M, Yu PL (2014) Statistical methods for ranking data, vol (1341). Springer, NP

Amodio S, D'Ambrosio A, Siciliano R (2016) Accurate algorithms for identifying the median ranking when dealing with weak and partial rankings under the kemeny axiomatic approach. Eur J Oper Res 249(2):667–676

Badal PS, Das A (2018) Efficient algorithms using subiterative convergence for kemeny ranking problem. Comput Oper Res 98:198–210

Banks A, Vincent J, Anyakoha C (2008) A review of particle swarm optimization. part ii: hybridisation, combinatorial, multicriteria and constrained optimization, and indicative applications. Nat Comput 7(1):109–124

Biernacki C, Jacques J (2013) A generative model for rank data based on insertion sort algorithm. Comput Stat Data Anal 58:162–176

Birlutiu A, Groot P, Heskes T (2010) Multi-task preference learning with an application to hearing aid personalization. Neurocomputing 73(7):1177–1185. https://doi.org/10.1016/j.neucom.2009.11.025

Boehmer N, Schaar N (2022) Collecting, classifying, analyzing, and using realworld elections. arXiv preprint arXiv:2204.03589

Bossert W, Storcken T (1992) Strategy-proofness of social welfare functions: the use of the kemeny distance between preference orderings. Soc Choice Welfare 9(4):345–360

Claus B (2022) pso: Particle swarm optimization [Computer software manual]. Retrieved from https://CRAN.R-project.org/package=pso (R package version 1.0.4)

Cook WD, Golany B, Penn M, Raviv T (2007) Creating a consensus ranking of proposals from reviewers' partial ordinal rankings. Comput Oper Res 34(4):954–965

Cook WD, Kress M (1990) A data envelopment model for aggregating preference rankings. Manag Sci 36(11):1302–1310
Cook WD, Kress M, Seiford LM (1986) An axiomatic approach to distance on partial orderings. Revue française d'automatique, d'informatique et de recherche opérationnelle. Recherche opérationnelle 20(2):115–122
Cook WD, Saipe A (1976) Committee approach to priority planning: the median ranking method. Cahiers Centre études Rech Opér 18(3):337–351
Crispino M, Antoniano-Villalobos I (2022) Informative priors for the consensus ranking in the bayesian mallows model. Bayesian Anal 1(1):1–24
Csató L, Tóth C (2020) University rankings from the revealed preferences of the applicants. Eur J Oper Res 286(1):309–320
D'Ambrosio A (2023) ConsRank: Compute the median ranking(s) according to the Kemeny's axiomatic approach [Computer software manual]. Retrieved from https://CRAN.R-project.org/package=ConsRank (R package version 2.1.4)
D'Ambrosio A, Amodio S, Iorio C (2015) Two algorithms for finding optimal solutions of the Kemeny rank aggregation problem for full rankings. Electr J Appl Stat Anal 8(2):198–213. https://doi.org/10.1016/j.ejor.2015.08.048
D'Ambrosio A, Heiser WJ (2016) A recursive partitioning method for the prediction of preference rankings based upon kemeny distances. Psychometrika 81(3):774–794
D'Ambrosio A, Heiser WJ (2019) A distribution-free soft-clustering method for preference rankings. Behaviormetrika 46(2):333–351
D'Ambrosio A, Iorio C, Staiano M, Siciliano R (2019) Median constrained bucket order rank aggregation. Comput Stat 34:787–802
de Borda JC (1781) Mémoire sur les élections au scrutin. Histoire de l'Academie Royale des Sciences
de Condorcet M (1785) Essai sur l'application de l'analyse a la probabilite des decisions rendues a la pluralite des voix (essay on the application of analysis to the probability of majority decisions)
Diaconis P (1988) Group representations in probability and statistics. Lecture Notes-Monograph Series, 11 , i–192, Retrieved 2024-09-10, from http://www.jstor.org/stable/4355560
Dwork C, Kumar R, Naor M, Sivakumar D (2001) Rank aggregation methods for the web. In: Proceedings of the 10th international conference on world wide web (pp. 613–622)
D'Ambrosio A, Mazzeo G, Iorio C, Siciliano R (2017) A differential evolution algorithm for finding the median ranking under the kemeny axiomatic approach. Comput Oper Res 82:126–138
Emond EJ, Mason DW (2000) A new technique for high level decision support (Tech. Rep.). Ottawa, Canada: Department of National Defence, Operational Research Division
Emond EJ, Mason DW (2002) A new rank correlation coefficient with application to the consensus ranking problem. J Multi-Criteria Decis Anal 11(1):17–28
Fürnkranz J, Hüllermeier E (2011) Preference learning: An introduction. In: Fürnkranz J, Hüllermeier E (eds) Preference learning. Springer, Berlin Heidelberg, pp 1–17
Gad AG (2022) Particle swarm optimization algorithm and its applications: a systematic review. Arch Comput Methods Eng 29(5):2531–2561. https://doi.org/10.1007/s11831-021-09694-4
Good I (1975) The number of orderings of n candidates when ties are permitted. Fibonacci Quarterly 13:11–18
Grbovic M, Djuric N, Guo S, Vucetic S (2013) Supervised clustering of label ranking data using label preference information. Mach Learn 93:191–225
Heiser WJ, D'Ambrosio A (2013) Clustering and prediction of rankings within a kemeny distance framework. B. Lausen, D. Van den Poel, & A. Ultsch (Eds.), Algorithms from and for nature and life (pp. 19–31). Cham: Springer International Publishing
Imran M, Hashim R, Khalid NEA (2013) An overview of particle swarm optimization variants. Proc Eng 53:491–496. https://doi.org/10.1016/j.proeng.2013.02.063
Irurozki E, Calvo B, Lozano JA (2016) PerMallows: An R package for mallows and generalized mallows models. J Stat Softw 71(12):1–30. https://doi.org/10.18637/jss.v071.i12
Kaufman L, Rousseeuw PJ (2008) Partitioning Around Medoids (Program PAM). Finding Groups in Data (pp. 68–125). John Wiley & Sons, Inc
Kemeny J, Snell J (1962) Preference rankings: An axiomatic approach. J. Kemeny & J. Snell (Eds.), Mathematical models in the social sciences (p. 9-23). New York: Blaisdell
Kendall MG (1938) A new measure of rank correlation. Biometrika 30(1/2):81–93

Kennedy J, Eberhart R (1995) Particle swarm optimization. Proceedings of icnn'95 - international conference on neural networks (Vol. 4, p. 1942-1948)
Lee PH, Yu P (2010) Distance-based tree models for ranking data. Comput Stat Data Anal 54(6):1672–1682
Maechler M, Rousseeuw P, Struyf A, Hubert M, Hornik K (2023) cluster: Cluster analysis basics and extensions [Computer software manual]. Retrieved from https://CRAN.R-project.org/package=cluster (R package version 2.1.6 – For new features, see the 'NEWS' and the 'Changelog' file in the package source))
Mandhani B, Meila M (2009, 16–18 Apr). Tractable search for learning exponential models of rankings. D. van Dyk & M. Welling (Eds.), Proceedings of the twelfth international conference on artificial intelligence and statistics (Vol. 5, pp. 392–399). Hilton Clearwater Beach Resort, Clearwater Beach, Florida USA: PMLR
Marden JI (1996) Analyzing and modeling rank data. Chapman & Hall, London
Mattei N, Walsh T (2013) Preflib: A library of preference data http://preflib.org. In: Proceedings of the 3rd international conference on algorithmic decision theory (adt 2013). Springer
Meissner M, Schmuker M, Schneider G (2006) Optimized particle swarm optimization (opso) and its application to artificial neural network training. BMC Bioinform 7(1):125. https://doi.org/10.1186/1471-2105-7-125
Mollica C, Tardella L (2017) Bayesian plackett-luce mixture models for partially ranked data. Psychometrika 82(2):442–458
Müllensiefen D, Hennig C, Howells H (2018) Using clustering of rankings to explain brand preferences with personality and socio-demographic variables. J Appl Stat 45(6):1009–1029
Murphy TB, Martin D (2003) Mixtures of distance-based models for ranking data. Comput Stat Data Anal 41(3–4):645–655
Parker H, Singh R, Badal PS (2023) Rankaggsigfur: Polynomially bounded rank aggregation under kemeny's axiomatic approach [Computer software manual]. Retrieved from https://CRAN.R-project.org/package=RankAggSIgFUR (R package version 1.0.0)
Patil D, Gupta R (2023) Gis-based multi-criteria decision-making for ranking potential sites for centralized rainwater harvesting. Asian J Civ Eng 24(2):497–506
Plaia A, Buscemi S, Sciandra M (2021) Consensus among preference rankings: a new weighted correlation coefficient for linear and weak orderings. Adv Data Anal Classific 15(4):1015–1037. https://doi.org/10.1007/s11634-021-00442-x
Plaia A, Sciandra M (2019) Weighted distance-based trees for ranking data. Adv Data Anal Classif 13:427–444
Poli R, Kennedy J, Blackwell T (2007) Particle swarm optimization. Swarm Intell 1(1):33–57. https://doi.org/10.1007/s11721-007-0002-0
Thompson GL (1993) Generalized permutation polytopes and exploratory graphical methods for ranked data. Ann Stat 21(3):1401–1430
Tibshirani R, Walther G, Hastie T (2001) Estimating the number of clusters in a data set via the gap statistic. J R Stat Soc Ser B 63(2):411–423
Vitelli V, Fleischer T, Ankill J, Arjas E, Frigessi A, Kristensen VN, Zucknick M (2023) Transcriptomic pan-cancer analysis using rank-based bayesian inference. Mol Oncol 17(4):548–563. https://doi.org/10.1002/1878-0261.13354
Vitelli V, Sørensen Ø, Crispino M, Frigessi A, Arjas E (2018) Probabilistic preference learning with the mallows rank model. J Mach Learn Res 18(158):1–49
Yoo Y, Escobedo AR (2021) A new binary programming formulation and social choice property for kemeny rank aggregation. Decis Anal 18(4):296–320
Young H (1995) Optimal voting rules. J Econ Persp 9(1):51–64
Young H, Levenglick A (1978) A consistent extension of condorcet's election principle. SIAM J Appl Math 35(2):285–300
Zhang Y, Wang S, Ji G (2015) A comprehensive survey on particle swarm optimization algorithm and its applications. Math Probl Eng 2015:931256. https://doi.org/10.1155/2015/931256
Zhang Y, Zhang W, Pei J, Lin X, Lin Q, Li A (2012) Consensus-based ranking of multivalued objects: A generalized borda count approach. IEEE Trans Knowl Data Eng 26(1):83–96
Zuwairie I (2014) Advances in particle swarm algorithms in asynchronous, discrete and multi-objective optimization. In: 2014 5th international conference on intelligent systems, modelling and simulation (p. 5-6)