INTERNATIONAL TRANSACTIONS IN OPERATIONAL RESEARCH

WILEY

INTERNATIONAL TRANSACTIONS IN OPERATIONAL RESEARCH

Intl. Trans. in Op. Res. 28 (2021) 3513–3535
DOI: 10.1111/itor.12704

An interactive algorithm for multiobjective ranking for underlying linear and quasiconcave value functions

Diclehan Tezcaner Öztürk[a,*] and Murat Köksalan[b,c]

[a] *Department of Industrial Engineering, Hacettepe University, Ankara, Turkey*
[b] *Department of Industrial Engineering, Middle East Technical University, Ankara, Turkey*
[c] *Ross School of Business, University of Michigan, Ann Arbor, MI, USA*
E-mail: diclehanozturk@hacettepe.edu.tr [Tezcaner Öztürk]; koksalan@metu.edu.tr [Köksalan]

Received 18 July 2018; received in revised form 17 May 2019; accepted 7 July 2019

Abstract

We develop interactive algorithms to find a strict total order for a set of discrete alternatives for two different value functions: linear and quasiconcave. The algorithms first construct a preference matrix and then find a strict total order. Based on the ordering, they select a meaningful pair of alternatives to present the decision maker (DM) for comparison. We employ methods to find all implied preferences of the DM, after he or she makes a preference. Considering all the preferences of the DM, the preference matrix is updated and a new strict total order is obtained until the termination conditions are met. We test the algorithms on several instances. The algorithms show fast convergence to the exact total order for both value functions, and eliciting preference information progressively proves to be efficient.

Keywords: interactive; strict total order; discrete alternative; convex cone; multiple criteria; transitivity; linear value function; quasiconcave value function

1. Introduction

Ordering a set of alternatives from "best" to "worst" considering multiple criteria has many applications in real life. Ranking countries, universities, MBA programs, places to visit are some examples. There are different approaches addressing this problem in the literature. Korhonen and Soismaa (1981) developed an algorithm that first estimates the weights of a linear value function and then produces a ranking of alternatives. Korhonen (1986) extended this approach considering a hierarchy of criteria and allowing for qualitative criteria. Köksalan and Tuncer (2009) and Ekiz and Tuncer Şakar (2020) used data envelopment analysis (DEA)-based scalarization for ranking alternatives with multiple criteria. Köksalan et al. (2010) developed a mathematical model that results in a flexible ranking based on a linear value function. They allow the weights assigned to each criterion to vary in favor of each alternative and observe how much the ranks can change. Karsu et al.

*Corresponding author.

(2018) developed an algorithm that attempts to distribute a "resource" among different parties in an equitable manner. They treat parties as criteria and different possible ways of distributing the resource as solution alternatives. They incorporate decision maker's (DM) preferences to obtain a partial or strict order of alternatives.

Multicriteria sorting is a variation of the ranking problem. In this problem, rather than producing a ranking, alternatives are sorted into predefined, preference-ordered classes. A variety of interactive approaches have been developed for the sorting problem. Köksalan and Ulu (2003), Köksalan and Özpeynirci (2009), Buğdacı et al. (2013), and Köksalan et al. (2017) developed approaches to sort alternatives into classes assuming that the DM's preferences are consistent with an underlying linear value function. Ulu and Köksalan (2001, 2014) sort alternatives under a more general underlying quasiconcave value function. Sobrie et al. (2019) develop an outranking-based algorithm to structure a sorting model.

Placing all alternatives to a strict preference order is a difficult problem. In the absence of a known value function, it requires substantial preference information from the DM. Dehnokhalaji et al. (2014) produce an estimated preference order (ranking) of alternatives for a DM who has a quasiconcave value function, and a set of preference information of the DM is available *a priori*. The accuracy of their ranking is sensitive to the available information, as expected.

In this paper, we consider ranking all alternatives without any prior information on the preferences of the DM. We develop theory and interactive algorithms to find the ranking of alternatives under multiple criteria for the cases of underlying linear and quasiconcave value functions separately. We develop procedures to choose the pair of alternatives to present the DM for his or her preference at each iteration in order to improve the efficiency of the algorithm. We also develop mechanisms to capture implied preferences (based on transitivity and the properties of the underlying value function) every time we obtain new preference information from the DM. The algorithms keep acquiring information progressively and guarantee obtaining the exact total order consistent with the underlying value function of the DM at termination. The algorithms can also be terminated prematurely with an estimated preference order. We conduct computational experiments to demonstrate the performances of the algorithms. We compare our algorithms with that of Dehnokhalaji et al. (2014) and demonstrate that it produces rankings closer to the exact total order for the same amount of preference information.

The paper unfolds as follows. We give some preliminary information in Section 2 and develop interactive algorithms for both value functions in Section 3. We conduct experiments and discuss their results in Section 4, and conclude in Section 5.

2. Background

Consider a discrete, finite, deterministic multiple criteria evaluation problem where a DM compares a set of n alternatives with respect to m criteria, given on a ratio scale. The set Ω of alternatives includes vectors $X_t = (x_{t1}, x_{t2}, \ldots, x_{tm}) \in \Re^m, t \in N = \{1, 2, \ldots, n\}$, with elements $x_{tk} > 0$, for all k. Without loss of generality, we assume that more is better in each criterion. We take most of the below definitions from Dehnokhalaji et al. (2014) directly as we will build upon their work.

Definition 1. *A vector $X^* \in \Omega \subset \Re^m$ is nondominated iff (if and only if) there does not exist another $X \in \Omega$ such that $X \geq X^*$ and $X \neq X^*$.*

Definition 2. *The function $f : \Omega \to \Re$, $X^* \in \Omega \subset \Re^m$ is called a value function if it has the following properties:*

i. $f(X^*) > f(X)$*, if X^* dominates X;*
ii. $f(X^*) > f(X)$*, iff X^* is preferred to X;*
iii. $f(X^*) \geq f(X)$*, if X^* is at least as preferred as X.*

Property 1 implies that function f is strictly increasing in set Ω.

We use the symbol "$\succ$" to denote the relationship "is preferred to" and "$\succcurlyeq$" to denote "is at least as preferred as." The DM's preferences are expressed by the set $P = \{(X_r, X_s) | X_r \succ X_s, r, s \in N\}$. Thus, P defines a strict partial order in S (an asymmetric, transitive, binary relation over S). We define a preference subset, a preference cone, and a preference polyhedron as follows.

Definition 3. *Let $X_1, \ldots, X_{k-1}, X_k, \ldots, X_l$ be l (distinct) points with the property that $X_t \succ X_k$ for $t = 1, \ldots, l$ and $t \neq k$. Then*

i. the set including alternatives $X_1, \ldots, X_{k-1}, X_k, \ldots, X_l$ is called a preference subset of P and is denoted by $\{X_1, \ldots, X_l; X_k\}$;
ii. the cone with vertex X_k is defined as

$$C_k = \left\{ X | X = X_k + \sum_{t=1, t\neq k}^{l} \mu_t \left(X_k - X_t\right), \mu_t \geq 0, t = 1, \ldots, l, t \neq k \right\},$$

and is called a preference cone. Let $T_k = \{1, \ldots, k, \ldots, l\}$ be the set of indices of alternatives that generate cone C_k.
iii. The polyhedron spanned by the points $X_1, \ldots, X_l$ is defined as

$$H\left(X_1, \ldots, X_l\right) = \left\{ X | X = \sum_{t=1}^{l} \mu_t X_t, \sum_{t=1}^{l} \mu_t = 1, \mu_t \geq 0, t = 1, \ldots, l \right\},$$

and is called a preference polyhedron.

For any $Z \in C_k$, $Z \neq X_k$, it can be shown that $f(X_t) > f(X_k) \geq f(Z)$ for $t = 1, \ldots, l$ and $t \neq k$, which implies that $X_k \succcurlyeq Z$ and $X_t \succ Z$, $t \neq k$. Point Z or any point V dominated by Z is called cone dominated. Also, if $Y \in H(X_1, \ldots, X_l)$, then from the definition of quasiconcavity it follows that $f(Y) \geq f(X_k)$ and this in turn implies that $Y \succcurlyeq X_k$ (for more details, see Korhonen et al., 1984).

Definition 4. *A binary relation R is a strict total order over A iff the following statements hold for all $a, b, c \in A$:*

*i. If $(a, b) \in R$ then $(b, a) \notin R$ (*asymmetric*).*
*ii. If $(a, b) \in R$ and $(b, c) \in R$ then $(a, c) \in R$ (*transitive*).*
*iii. If $a \neq b$, then either $(a, b) \in R$ or $(b, a) \in R$ (*connected*).*

3. Interactive algorithms to obtain a strict total order

We present a general structure (interactive algorithm) to find the strict total order of a DM and then develop specific algorithms for the cases where the DM's preferences are consistent with linear and quasiconcave value functions.

Similar to Dehnokhalaji et al. (2011, 2014), we construct and update a preference matrix $P^* = [p^*_{ij}]$, where p^*_{ij} is a preference measure of alternative X_i over alternative X_j, throughout the algorithm. The values of p^*_{ij} vary between 0 and 1, where 0 (1) indicates that X_i is at most (at least) as preferred as X_j. Larger values of p^*_{ij} indicate that X_i is closer to being preferred to X_j. Dehnokhalaji et al. (2011, 2014) develop several procedures for determining p^*_{ij} for a DM whose preferences are consistent with a quasiconcave value function. We build upon their procedures for quasiconcave value functions, and develop new procedures for linear value functions.

In each iteration, our algorithm asks the DM to compare a pair of alternatives and produces a strict total order using all previous preferences. The algorithm continues until a termination criterion is satisfied.

We next give a general interactive algorithm that can be applied to any underlying value function. We later develop it for the special cases of linear and quasiconcave value functions separately.

A general interactive algorithm

Step 0. Initialization. Find p^*_{ij} for $i = 1, \ldots, n$, $j = 1, \ldots, n$ and construct P^*.
Step 1. Total order. Obtain a strict total order using P^*. If the termination criterion is satisfied, stop. Otherwise, go to Step 2.
Step 2. Selection. Select two alternatives, X_{i^*} and X_{j^*}.
Step 3. Comparison. Ask the DM to compare alternatives X_{i^*} and X_{j^*}. Update P^*. Go to Step 1.

There are multiple ways of conducting each step. We next explain the specifics of our method for linear and quasiconcave value functions separately. In order to cover the necessary background corresponding to each case, we briefly discuss the relevant aspects of Dehnokhalaji et al. (2014).

3.1. An interactive algorithm for underlying linear value functions

We next develop various mechanisms of the interactive algorithm for underlying weighted linear value functions. We identify the weight sets for which each of a pair of alternatives is favored over the other. We quantify the likelihood of X_i being preferred to X_j based on the relative hypervolume of the feasible weights that favor X_i over X_j.We present our procedure (*Partitioning the weight space*) and the weight-hypervolume computations (*H-V algorithm*) next. This procedure forms the basis of constructing the preference matrix in Steps 0 and 3 of the general interactive algorithm. In Step 1, we obtain a strict total order solving an integer programming problem developed by Bowman and Colantoni (1973). To update the preference matrix in Step 3, we use Algorithm *Lin* explained next. This algorithm first extracts all implied preferences from the comparison of alternatives X_{i^*}

and X_{j^*}, and then partitions the feasible weight space between all alternative pairs X_i and X_j to calculate p^*_{ij}.

We next explain each mechanism in detail.

3.1.1. Partitioning the weight space

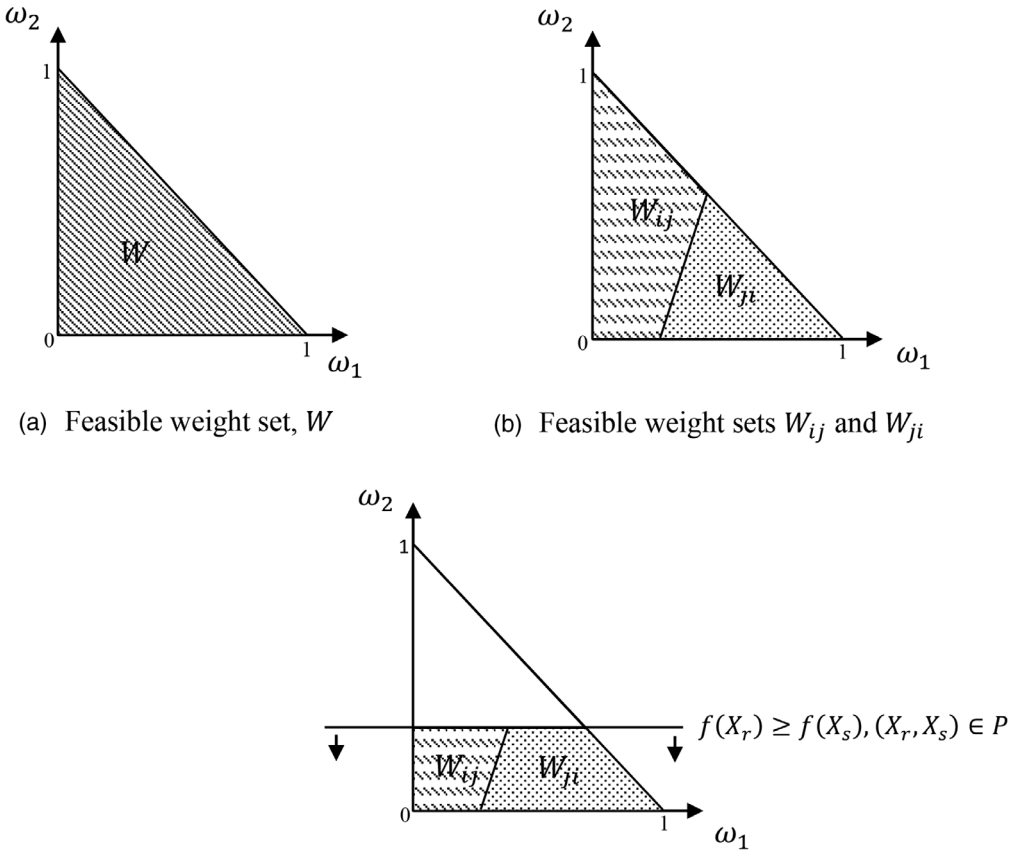
Let $X_t = (x_{t1}, x_{t2}, \ldots, x_{tm})$, $\omega = (\omega_1, \omega_2, \ldots, \omega_m)$, and $f(X_t) = \sum_k \omega_k x_{tk}$ be the form of the DM's underlying value function, where ω_k denotes the weight of criterion k. The possible weight values have to be consistent with the preference information provided by the DM, that is, $\omega \in W = \{\omega : \omega_k \geq 0 \, \forall k, \sum_k \omega_k = 1, f(X_r) \geq f(X_s) \text{ for all } (X_r, X_s) \in P\}$. If we had exact information on ω_k values, then constructing the exact strict total order would be straightforward. However, at the beginning of the interactive algorithm, we do not have any preference information. The only information we have on the values of the weights is their scaling and normalization: without loss of generality, we scale weight values between 0 and 1 and set their sum to 1. As we progress, using the responses of the DM, we impose further restrictions on the possible values of weights. The more we restrict the weights, the better we estimate the p^*_{ij} values.

Let $W_{ij} = \{\omega : \omega \in W, f(X_i) \geq f(X_j)\}$. The optimality region for each alternative is a $(m-1)$-polytope in R^{m-1}, as given in Theorem 2 of Kim et al. (2006). We show an example plot of the weight space in Fig. 1 for the 3-objective case. The graph shows ω_1 and ω_2 values, and $\omega_3 = 1 - \omega_1 - \omega_2$. The shaded area in Fig. 1a represents the original feasible weights (W) prior to obtaining any preference information. For two alternatives (say X_i and X_j), there are three different possibilities: (a) $W_{ij} = \emptyset$, $W_{ji} = W$, (b) $W_{ij} = W$, $W_{ji} = \emptyset$, and (c) $W_{ij}, W_{ji} \neq \emptyset$. Let $V(W_{ij}) = v_{ij}$ denote the hypervolume of W_{ij} (area in the 3-objective case shown in Fig. 1). Note that, if any weight vector in the feasible weight space is equally likely, then we expect the likelihood that X_i is preferred to X_j to be proportional to the hypervolume of weights favoring X_i over X_j relative to the total hypervolume of the feasible weights. That is $p^*_{ij} = \frac{v_{ij}}{V(W)}$ and $p^*_{ji} = 1 - p^*_{ij}$, where $V(W) = v_{ij} + v_{ji}$. Figure 1b demonstrates the weight sets for which X_i and X_j are preferred to each other prior to any preference information. In Fig. 1c, a preference constraint is imposed on the weight space ($f(X_r) \geq f(X_s)$). In this case, W_{ij} and W_{ji} shrink as we need to consider only those portions that satisfy the imposed constraint. As we introduce additional preference constraints, we expect the feasible weight space to keep shrinking and p^*_{ij} to approach 1(0) if the DM in fact prefers $X_i(X_j)$ to $X_j(X_i)$.

We narrow down the feasible weight range each time we acquire new preference information from the DM. For example, in Fig. 1, if the DM prefers X_i to X_j, the new feasible weight set, W, is equal to W_{ij} of Fig. 1b. In this case, $p^*_{ij} = 1$ and $p^*_{ji} = 0$. We need to keep comparing alternative pairs that will be favored in some parts of the new feasible space until termination conditions are satisfied.

3.1.2. Constructing a strict total order

We discussed in Definition 4 that P^* has to be asymmetric, transitive, and connected in order to define a strict total order. An intuitive simple rule could be to rank X_i before X_j whenever $p^*_{ij} > p^*_{ji}$. However, such a rule has the risk of violating transitivity and preventing the construction of a strict total order. Model 1 developed by Bowman and Colantoni (1973) produces a strict total order



(a) Feasible weight set, W

(b) Feasible weight sets W_{ij} and W_{ji}

(c) Updated feasible weight sets W_{ij} and W_{ji}

Fig. 1. Partitioning the weight space.

among n alternatives based on $P^* = [p^*_{ij}]$ even when transitivity is violated by the above rule. As in Dehnokhalaji et al. (2014), we also use this model.

Model 1	
$\max \sum_{i,j=1}^{n} p^*_{ij} a_{ij}$	
s.t. $a_{ij} + a_{ji} = 1$	$i, j = 1, 2, \ldots, n$
$a_{hi} + a_{ij} + a_{jh} \leq 2$	$i, j, h = 1, 2, \ldots, n$
$a_{ij} \in \{0, 1\}$	$i, j = 1, 2, \ldots, n$

3.1.3. Updating P^*

Consider that we would like to estimate the likelihood of an alternative, say X_a, to be preferred to another alternative, say X_b. We first solve Model 2 (that has an artificial objective function of a constant value zero) to see if there are any feasible weights consistent with past responses of the

DM that would make X_a at least as preferred as X_b. If Model 2 is feasible, we conclude that there are such weights.

Model 2
Max 0
s.t. $\sum_k \omega_k(x_{rk} - x_{sk}) \geq 0$ $(X_r, X_s) \in P$
$\sum_k \omega_k(x_{ak} - x_{bk}) \geq 0$
$\sum_k \omega_k = 1$
$0 \leq \omega_k \leq 1$ $\forall k$

In this case, we estimate the likelihood of X_a being preferred to X_b using the procedure suggested in Kim et al. (2006). The likelihood estimate depends on the relative hypervolume of the feasible weight space that favors X_a over X_b. For m criteria, the weight space can be represented in $m-1$ dimensions due to the normalization of the weights. Additional constraints derived from the past preferences of the DM reduce the feasible weight space. The weight space is always a polytope by construction. If a pair of alternatives can outperform each other under certain feasible weight values, we can partition the feasible polytope of weights into two half-spaces (polytopes), each favoring one of the two alternatives. We can use the relative hypervolumes of the two polytopes as measures of the likelihoods of alternatives being preferred to each other. A hypervolume computation requires integration over $m-1$ dimensions. The procedure employed by Kim et al. (2006) uses the vertices of the polytope based on Büeler et al.'s (2000) approach. We use the same approach and present its steps next.

H-V algorithm (hypervolume computation)

Step 1. Find a minimal vertex representation (V-representation) of the convex polytope.
In this representation, the polytope is defined by its vertices, y_l, $l = 1, \ldots, m-1$.

Step 2. Divide the convex polytope into r simplices using Delaunay triangulation algorithm.
Each simplice, q, is defined by a subset of the vertices, y_l^q.

Step 3. Calculate the volume of each simplice q, V_q, and add all volumes to find the total volume of the convex polytope, v_{ij}, using the following equations. Here, y_{lk}^q indicates the kth criterion value of vertex l of simplice q.

$$V_q = \frac{1}{(m-1)!} det \begin{vmatrix} 1 & y_{11}^q & \ldots & y_{1,m-1}^q \\ 1 & y_{21}^q & \ldots & y_{2,m-1}^q \\ & & \ldots & \\ 1 & y_{t1}^q & \ldots & y_{t,m-1}^q \end{vmatrix}$$

$$v_{ij} = \sum_{q=1}^{r} V_q$$

After obtaining a preference from the DM ($X_r \succ X_s$), we use Algorithm *Lin* to update P^*. We use the new preference $X_r \succ X_s$ to add a constraint and narrow down the weight space in Model 2. Based on the results of Model 2, we update the p_{ij}^* values of all alternative pairs that have not already been

set to 0 or 1. In case Model 2 is infeasible, the p^*_{ij} values are set to 0 or 1 for all relations implied with this infeasibility. If it is feasible for both $a = i, b = j$ and $a = j, b = i$, we use H-V algorithm to calculate p^*_{ij}.

Algorithm *Lin*

Step 0. Set $i = 1$ and $j = 2$.
Step 1. Set $a = i, b = j$, and $c = 1$.
Step 2. If $p^*_{ab} \in \{0, 1\}$, go to Step 5. Otherwise, solve Model 2. If Model 2 is infeasible, set $p^*_{ba} = 1$ and $p^*_{ab} = 0$. For all $t = 1, \ldots, n$, if $p^*_{tb} = 1$ and $p^*_{ta} \neq 1$, set $p^*_{ta} = 1$ and $p^*_{at} = 0$. For all $t = 1, \ldots, n$ and for all $u = 1, \ldots, n$, if $p^*_{ta} = 1$, $p^*_{au} = 1$ and $p^*_{tu} \neq 1$, set $p^*_{tu} = 1$ and $p^*_{ut} = 0$. Go to Step 5. If Model 2 is feasible, go to Step 3.
Step 3. Set $c \leftarrow c + 1$. If $c = 3$, go to Step 4. Otherwise, set $a = j$ and $b = i$, and go to Step 2.
Step 4. Calculate v_{ij} using the H-V algorithm. Set $p^*_{ij} = \frac{v_{ij}}{V(W)}$. Set $p^*_{ji} = 1 - p^*_{ij}$. Go to Step 5.
Step 5. Set $j \leftarrow j + 1$. If $j = n + 1$, go to Step 6. Otherwise, go to Step 1.
Step 6. Set $i \leftarrow i + 1$. If $i = n$, terminate the algorithm. Otherwise, set $j = i + 1$ and go to Step 1.

Consider a strict total order $X_{j_1} \succ X_{j_2} \succ \cdots \succ X_{j_N}$. Let R_i denote the index of the alternative at rank i ($R_i = j_i$) and O_j denote the rank of alternative X_j ($O_{j_i} = i$). We terminate the algorithm when a predefined precision condition that is specified by the two parameters, ε and *limit*, is satisfied. The algorithm keeps improving the preference relations between pairs of alternatives by getting additional preference information from the DM as necessary. We next present the steps of the interactive algorithm.

Algorithm *IntLin*

Step 0. Initialization.
Set the values of ε and *limit*. Set $p^*_{ii} = 0.5, i \in N$, $p^*_{ij} = \frac{v_{ij}}{V(W)}$, $p^*_{ji} = \frac{v_{ji}}{V(W)}$ for all $(i, j), i \neq j$.
Step 1. Total order.
Solve Model 1 to find a strict total order. If p^*_{ij} values of at least ε% of the consecutively-ranked alternatives are greater than *limit,* stop. That is, if at least ε% of $p^*_{R_k, R_{k+1}}$, $k = 1, \ldots, n - 1$, are greater than *limit,* stop. Otherwise, go to Step 2.
Step 2. *Selection*.
Identify alternative pairs whose relative preference measures are inconsistent with their relative ranks in the produced strict total order in Step 1, and place them in set I, that is, $I = \{(i, j) | p^*_{ij} > p^*_{ji}, O_i > O_j\}$. If $I \neq \emptyset$, let $(i^*, j^*) = argmax_{(i,j)\in I}\{|p^*_{ij} - p^*_{ji}|\}$. If $I = \emptyset$, let $(i^*, j^*) = argmin_{i=1,\ldots,n, j=1,\ldots,n, i\neq j}\{|p^*_{ij} - p^*_{ji}|\}$.
Step 3. Comparison.
Ask for comparison between alternatives X_{i^*} and X_{j^*}, denote the index of the preferred and inferior alternatives as r and s, respectively. Update P^* using Algorithm *Lin* and update W, $W = W_{rs}$. Go to Step 1.

In Step 0, we start with a feasible weight set, W, that includes all possible weights (see Fig. 1a for a 3-criteria example). The initial hypervolume of W, $V(W)$, is 0.5 for this example (since all weights are scaled between 0 and 1). To find the hypervolume v_{ij} for each $(i, j), i \neq j$, we divide the region

by a hyperplane, introducing the constraint $\sum_k \omega_k(x_{ik} - x_{jk}) \geq 0$, as shown in Fig. 1b. We then find W_{ij} and W_{ji} for all (i, j), $i \neq j$, as in Fig. 1b. In Step 2, if there are alternative pairs having relative preference measure values inconsistent with their relative ranks in the produced strict total order in Step 1, we choose one of those pairs to present the DM in order to rectify this inconsistency. We choose the pair that has the largest inconsistency in their relative preference measures. If relative preference measures of all pairs are consistent with their ranks, then we choose the pair having the minimum difference in the relative preference measures. Our motivation here is to disclose the true preference relation of a pair for which we have the least information.

3.2. An interactive algorithm for underlying quasiconcave value functions

In this section, we develop the specifics of the interactive algorithm for underlying quasiconcave value functions. We quantify the likelihood of X_i being preferred to X_j by measuring how far X_i is from being provably inferior to X_j and how far X_i is to being provably preferred to X_j. For these two measures, we utilize the previous preferences and implied preferences (as captured by the constructed cones) of the DM. We explain how we update the cones in Algorithm *Cone_update* and how we utilize the cone information in constructing P^* in Algorithm *Qsq*. Algorithm *Qsq* is inspired by the algorithm developed in Dehnokhalaji et al. (2014) that is explained next. We then discuss the other mechanisms in detail.

3.2.1. Background on constructing P^*

Dehnokhalaji et al. (2014) develop Algorithms *PM* and *(i,j)* to construct a strict total order among alternatives using an available set of preferences of the DM. Their approach consists of two stages: constructing the preference matrix $P^* = [p^*_{ij}]$ and constructing a strict total order using P^*.

In the first stage, they calculate p^*_{ij}, for all $i, j \in N$, using Algorithm *(i,j)* defined in Dehnokhalaji et al. (2014). They use two measures: (a) shortest radial[1] distance-to-domination, $|\alpha_{ij}|$, and (b) shortest radial distance-to-dominate, $|\beta_{ij}|$, to determine the value of p^*_{ij}. These measures can be interpreted as the smallest deterioration percentage of alternative X_i to be dominated by X_j and the smallest improvement percentage of alternative X_i to dominate X_j, respectively, on the radial ray passing through alternative X_i in the objective function space.

In order to find the initial values of α_{ij} and β_{ij}, they solve and use the optimal objective function values of Models 3 and 4, respectively.

Model 3	Model 4
$\max \omega_{ij}$	$\min \gamma_{ij}$
s.t. $X_j \geq (1 + \omega_{ij})X_i$	s.t. $X_j \leq (1 + \gamma_{ij})X_i$

Let solutions to Models 3 and 4 yield ω^*_{ij} and γ^*_{ij}, where the former is the shortest radial distance of X_i to the quadrant dominated by X_j (made of all points dominated by X_j) and the latter is the

[1]Radial means that the objective values of the solution are changed proportionately.

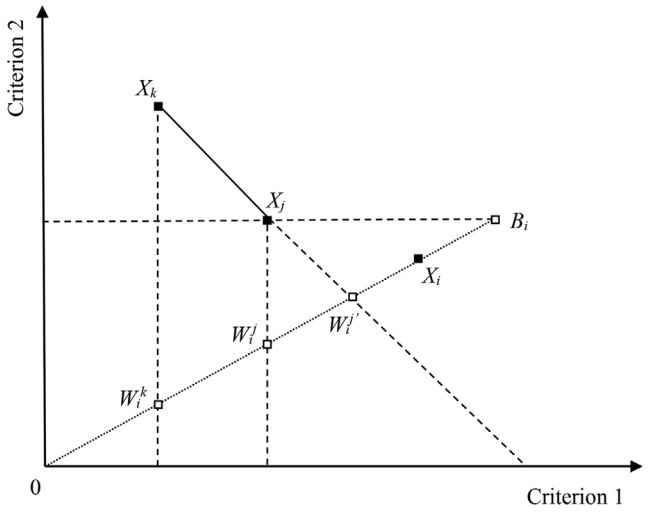


Fig. 2. Finding preference measure of X_i over X_j.

shortest radial distance of X_i to the quadrant dominating X_j (made of all points dominating X_j). Then $\alpha_{ij} = \omega_{ij}^*$ and $\beta_{ij} = \gamma_{ij}^*$. We illustrate the measures for X_i and X_j in Fig. 2. X_i should move to point W_i^j in order to be dominated by X_j, and should move to point B_i in order to dominate X_j. Here, $|\omega_{ij}^*|$ corresponds to $\frac{W_i^j X_i}{OX_i}$ and $|\gamma_{ij}^*|$ corresponds to $\frac{B_i X_i}{OX_i}$.

When incorporating preference information, they discuss two cases separately.

Case 1: Finding p_{ij}^* where X_j is the vertex of a cone.

After setting the initial values of α_{ij} and β_{ij}, they check if these values can be improved based on the available preference information and update these values if necessary. Assume that there is a single pairwise comparison information available from the DM. That is, for some specific k, $j \in N$ the preference set is $P = \{(X_k, X_j) | X_k \succ X_j\}$ and leads to cone C_j. They find the radial distance of X_i to the cone-dominated region of C_j and to the region that dominates any convex combination of the cone generators of C_j. If the radial distance of X_i to the cone-dominated region of C_j, ρ_{ij}^* is less than $|\alpha_{ij}|$, they set $\alpha_{ij} = \rho_{ij}^*$. Similarly, if the radial distance of X_i to the region that dominates any convex combination of the cone generators of C_j, φ_{ij}^* is less than $|\beta_{ij}|$, they set $\beta_{ij} = \varphi_{ij}^*$. The values of ρ_{ij}^* and φ_{ij}^* are the optimal objective function values for Models 5 and 6, respectively. Model 5 constructs a cone having vertex X_j and the remaining generators X_t, $t \in T_j - \{j\}$ that are preferred to X_j. If ρ_{ij}^* is positive, then X_i is dominated by cone C_j and all cone generators, including X_j, are preferred to X_i. If ρ_{ij}^* is zero, then X_j is at least as preferred as X_i, and X_t, $t \in T_j - \{j\}$ are preferred to X_i. If ρ_{ij}^* is negative, then C_j does not provide a conclusive preference information about X_i; however, ρ_{ij}^* can be used to set the value of p_{ij}^*. Model 6, on the other hand, checks if X_i can be represented as a convex combination of cone generators, $t \in T_j$. If φ_{ij}^* is negative, then X_i dominates some convex combinations of cone generators and hence is preferred to the vertex of cone C_j, X_j. If φ_{ij}^* is zero, then X_i is a convex combination of the cone generators of C_j. If φ_{ij}^* is positive, there is no conclusive preference information but its magnitude serves as a measure of how close X_i is to the

closest point that dominates convex combinations of cone generators. This is used in determining p^*_{ij}.

Model 5	Model 6
$\max \rho_{ij}$	$\min \varphi_{ij}$
s.t. $X_j + \sum_{t\in T_j-\{j\}} \mu_t(X_j - X_t) \geq (1+\rho_{ij})X_i$	s.t. $\sum_{t\in T_j} \mu_t X_t \leq (1+\varphi_{ij})X_i$
$\mu_t \geq 0, t \in T_j - \{j\}$	$\sum_{t\in T_j} \mu_t = 1, \mu_t \geq 0, t \in T_j$

In Fig. 2, $W_i^{j'}$ shows the closest point to X_i in the radial direction that is cone dominated by C_j. Since point $W_i^{j'}$ is closer to X_i than point W_i^j, $|\alpha_{ij}|$ is set to ρ^*_{ij}. The optimal solution of Model 6 turns out to be equal to that of Model 4 for the case in Fig. 2 and there is no need to update β_{ij}.

Case 2: Finding p^*_{ik} where alternative X_k is a nonvertex generator of cone C_j and X_i is not a generator of cone C_j

In Fig. 2, X_k is a nonvertex cone generator and X_i is not a cone generator. Initially, W_i^k is found as the point on the radial ray passing through X_i, which is closest to the dominating region of X_k. Using cone C_j, Model 5 yields that the closest point to X_i which is cone dominated in the radial direction from X_i is $W_i^{j'}$, with the optimal objective function value ρ^*_{ij}. Since X_j is at least as preferred as $W_i^{j'}$ and X_k is preferred to X_j, it is known that X_k is preferred to $W_i^{j'}$. Since $\rho^*_{ij} < |\alpha_{ik}| = \frac{W_i^k X_i}{OX_i}$, they set $\alpha_{ik} = \rho^*_{ij}$.

After calculating the α_{ij} and β_{ij} values, the preference measure of X_i over X_j, p^*_{ij}, is set to $\frac{|\alpha_{ij}|}{|\alpha_{ij}|+|\beta_{ij}|}$. After constructing P^*, they form the strict total order in the second stage using Model 1.

3.2.2. Progressive construction of P^*

In this section, we describe our approach to update P^* progressively based on the preference information obtained from the DM. We first develop the necessary theory for constructing P^* and updating the cones.

Theorem 1. *Let $T'_j = T_j \cup \{i\}$. If $\varphi^*_{ij} \leq 0$ in Model 6, then $C'_j \equiv C_j$.*

Proof. $\varphi^*_{ij} \leq 0$ implies $X_i \succcurlyeq X_j$. Suppose we evaluate alternatives X_k and X_j. We first solve Models 5 and 6 to find the preference measure of X_k over X_j. Let the dual of Model 5 be Model 5′. Setting $\bar{\nu} = -\bar{\delta}$ and adding the redundant constraint $\bar{\nu}(X_j - X_j) \geq 0$, we obtain Model 5″.

Model 5′	Model 5″
$\min \rho_{kj} = \bar{\delta}(X_k - X_j)$	$\min \rho_{kj} = \bar{\nu}(X_j - X_k)$
s.t. $\bar{\delta}(X_j - X_t) \geq 0\ t \in T_j - \{j\}$	s.t. $\bar{\nu}(X_t - X_j) \geq 0\ t \in T_j$
$-\bar{\delta}X_k = 1$	$\bar{\nu}X_k = 1$
$\bar{\delta} \leq 0$	$\bar{\nu} \geq 0$

If alternative X_i is also included in the cone, we would additionally have the following constraint in Model 5′′: $\bar{v}(X_i - X_j) \geq 0$.

Since $\varphi_{ij}^* \leq 0$ in Model 6, we know that X_i is a convex combination or dominates some convex combinations of the generators of cone C_j; $X_i \geq \sum_{t \in T_j} \mu_t X_t$, for some μ_t satisfying $\sum_{t \in T_j} \mu_t = 1, \mu_t \geq 0, t \in T_j$.

Since $\bar{v}X_t \geq \bar{v}X_j$ for all $t \in T_j$ from Model 5′′ then $\bar{v} \sum_{t \in T_j} \mu_t X_t \geq \bar{v}X_j$, for any μ_t satisfying $\sum_{t \in T_j} \mu_t = 1, \mu_t \geq 0, t \in T_j$. Since $X_i \geq \sum_{t \in T_j} \mu_t X_t$, for some μ_t, we have $\bar{v}X_i \geq \bar{v} \sum_{t \in T_j} \mu_t X_t \geq \bar{v}X_j$ for $\sum_{t \in T_j} \mu_t = 1, \mu_t \geq 0 \; \forall t \in T_j$. Therefore, $\bar{v}(X_i - X_j) \geq 0$ is redundant in Model 5′′.

Similarly, let the dual of Model 6 be Model 6′. Setting $\bar{\tau} = -\bar{\sigma}$, we obtain Model 6′′.

Model 6′	Model 6′′
$\max \varphi_{kj} = \bar{\sigma} X_k + \pi$	$\max \varphi_{kj} = \pi - 1$
s.t. $\bar{\sigma} X_t + \pi \leq 0 \; t \in T_j$	s.t. $\pi \leq \bar{\tau} X_t \; t \in T_j$
$-\bar{\sigma} X_k = 1$	$\bar{\tau} X_k = 1$
$\bar{\sigma} \leq 0, \pi$ unrestricted in sign	$\bar{\tau} \geq 0, \pi$ unrestricted in sign

If alternative X_i is also included in the cone, we have the following additional constraint: $\pi \leq \bar{\tau} X_i$.

Since $\pi \leq \bar{\tau} X_t$ for all $t \in T_j$ from Model (6′′), $\pi \leq \bar{\tau} \sum_{t \in T_j} \mu_t X_t$ for any μ_t satisfying $\sum_{t \in T_j} \mu_t = 1, \mu_t \geq 0, t \in T_j$. Since $X_i \geq \sum_{t \in T_j} \mu_t X_t$ for some μ_t satisfying $\sum_{t \in T_j} \mu_t = 1, \mu_t \geq 0$, $t \in T_j$, $\pi \leq \bar{\tau} X_i$. The additional constraint, $\pi \leq \bar{\tau} X_i$, is always satisfied and hence is a redundant constraint in Model 6′′.

For both Models 5′′ and 6′′, we do not obtain any additional information from the implied preference $X_i \succcurlyeq X_j$, and $C_j' \equiv C_j$. □

Theorem 1 implies that there is no gain in including an alternative that is a convex combination or dominates some convex combinations of cone generators (and hence is found to be at least as preferred as the vertex of that cone) as a cone generator. Therefore, we do not include X_i that is a convex combination or dominates some convex combinations of cone generators as a new cone generator in T_j.

We next show that if X_i is a convex combination or dominates some convex combinations of the generators of cone C_j, and if X_j is a nonvertex generator of cone C_k, then X_i is also a convex combination of or dominates some convex combinations of the generators of cone C_k. Thus, since including X_i as a cone generator in C_j is not necessary due to Theorem 1, then including X_i as a cone generator in C_k is also not necessary.

Corollary 1. *If* $\sum_{t \in T_j} \mu_t X_t \leq X_i$, *for some* $\sum_{t \in T_j} \mu_t = 1, \mu_t \geq 0, t \in T_j$ *and if* $j \in T_k$ *then* $\sum_{t \in T_k} \mu_t X_t \leq X_i$, *for some* $\sum_{t \in T_k} \mu_t = 1, \mu_t \geq 0, t \in T_k, j \neq k$.

Proof. Since $j \in T_k$, $T_j \subset T_k$. If $\varphi_{ij}^* \leq 0$ in Model 6, then $\varphi_{ik}^* \leq 0$ in Model 6 as well. □

We develop Algorithm *Qsq* that processes all relevant cones and updates P^* after obtaining new preference information (e.g., $X_r \succ X_s$) from the DM. Algorithm *Qsq* requires to update cones regularly and we develop this phase separately (as Algorithm *Cone_update*) in order to simplify

presentation. We first present Algorithm *Cone_update*. Let U be the set of indices of the vertexes of the cones that are assigned new nonvertex generators. Let(i, j) be the indexes of alternative pairs such that $j \in U$, $i \in N$ and $0 < p^*_{ij} < 1$. We place all such pairs in a set, Q. Both sets, U and Q, will be constructed by Algorithm *Cone_update*.

Algorithm *Cone_update*

Step C.1. Set $p^*_{rs} = 1$ and $p^*_{sr} = 0$. Set $Q = \emptyset$.
Step C.2. Update $T_s \leftarrow T_s \cup \{r\}$. Set $U = \{s\}$.
Step C.3. For all $i = 1, \ldots, n$, if $p^*_{ir} = 1$ and $p^*_{is} \neq 1$, set $p^*_{is} = 1$ and $p^*_{si} = 0$. If $i \in T_r$, set $T_s \leftarrow T_s \cup \{i\}$ and set $U \leftarrow U \cup \{s\}$.
Step C.4. For all $i = 1, \ldots, n$ and for all $j = 1, \ldots, n$, if $p^*_{is} = 1$, $p^*_{sj} = 1$ and $p^*_{ij} \neq 1$, set $p^*_{ij} = 1$ and $p^*_{ji} = 0$. If $i \in T_s$, set $T_j \leftarrow T_j \cup \{i\}$ and set $U \leftarrow U \cup \{j\}$.
Step C.5. Set $Q \leftarrow Q \cup \{(i, j) : j \in U, i \in N, 0 < p^*_{ij} < 1\}$.

In the first step, we set the preference measures based on DM's response. In Step C.2, we include alternative X_r as a generator to the cone with vertex X_s. Due to transitivity, in Steps C.3 and C.4, we update all related cones. We first update the preference values of all alternatives that are preferred to X_r but not yet preferred to X_s. If those alternatives are also cone generators of C_r, they are included as cone generators of cone C_s as well. We consider a similar case in Step C.4. In both steps, we only include the alternatives that provide additional information as cone generators (i.e., we do not include those that are redundant due to Corollary 1). The indexes of the cones that are updated are included in set U in Steps C.2–C.4. In the last step, we include the alternative pairs whose preference values have the potential to be improved to set Q, based on the updated cones. Algorithm *Qsq* requires C.1–C.5 to be executed at the start, C.2–C.5 to be executed whenever a cone-dominated alternative is identified, and C.3–C.5 to be executed whenever an alternative that dominates convex combinations of the generators of a cone is identified. Each case leads to further updating of P^*.

We next present Algorithm *Qsq* that updates all cones and P^* after obtaining preference information ($X_r \succ X_s$) from the DM. The algorithm starts with the current values of p^*_{ij} and the optimal objective function values of Models 3 and 4 (ω^*_{ij} and γ^*_{ij}, respectively). If X_j was a vertex of a cone in a previous iteration, then we also retrieve the optimal objective function values of Models 5 and 6 (ρ^*_{ij} and φ^*_{ij}, respectively) from that iteration.

Algorithm *Qsq*

Step 1. Use Algorithm *Cone_update* to update all related cones.
Step 2. Set $i \leftarrow a$, $j \leftarrow b$, where $(a, b) \in Q$. Set $Q \leftarrow Q - \{(a, b)\}$. Go to Step 2.1.
Step 2.1. Solve Model 5 to check for the status of X_i relative to cone C_j.

- If $\rho^*_{ij} \geq 0$, then X_i falls into cone C_j and hence $X_j \succcurlyeq X_i$. Set $p^*_{ji} = 1$ and $p^*_{ij} = 0$. Set $r = j$, $s = i$, execute Steps C.2–C.5 of Algorithm *Cone_update*. If $Q = \emptyset$, go to Step 3; otherwise, go to Step 2.
- If $\rho^*_{ij} < 0$, go to Step 2.2.

Step 2.2. Solve Model 6 to check for the status of X_i relative to the generators of cone C_j.

- If $\varphi_{ij}^* \leq 0$, then $X_i \succcurlyeq X_j$. Set $p_{ij}^* = 1$ and $p_{ji}^* = 0$. Set $r = i, s = j, U = \emptyset$, execute Steps C.3–C.5 of Algorithm *Cone_update*. If $Q = \emptyset$, go to Step 3; otherwise, go to Step 2.
- If $\varphi_{ij}^* > 0$, go to Step 3 if $Q = \emptyset$ and go to Step 2, otherwise.

Step 3. For all $i = 1, \ldots, n$ and $j = 1, \ldots, n$, $i \neq j$, if $0 < p_{ij}^* < 1$ and if X_j is not a vertex of a cone, set $\alpha_{ij} = \omega_{ij}^*$ and $\beta_{ij} = \gamma_{ij}^*$. If $0 < p_{ij}^* < 1$ and if X_j is a vertex of a cone, update α_{ij} and β_{ij} as follows:

$$\alpha_{ij} \leftarrow \max\left\{\omega_{ij}^*, \rho_{ij}^*\right\}$$

$$\beta_{ij} \leftarrow \min\left\{\gamma_{ij}^*, \varphi_{ij}^*\right\}$$

Step 4. For all $i = 1, \ldots, n$, $j = 1, \ldots, n$ and $h = 1, \ldots, n$, $i \neq j$, $i \neq h$, if $p_{jh}^* = 1$, $0 < p_{ij}^*, p_{ih}^* < 1$, and $\alpha_{ij} < \alpha_{ih}$, set $\alpha_{ij} = \alpha_{ih}$.

Step 5. For all $i = 1, \ldots, n$, $j = 1, \ldots, n$, $i \neq j$, if $0 < p_{ij}^* < 1$, set $p_{ij}^* = \frac{-\alpha_{ij}}{\beta_{ij} - \alpha_{ij}}$.

In the first step, we update all cones using the stated preference of the DM. If new generators have been added to the cone that has vertex X_j, in Step 2, we update the objective function values of Models 5 and 6 for all pairs $(i, j) \in Q$. In Step 2.1, we solve Model 5 to update ρ_{ij}^*. In this step, $\rho_{ij}^* \geq 0$ implies that alternative X_j is at least as preferred as alternative X_i. In this case, we update cone C_i by adding X_j as well as all alternatives preferred to X_j as nonvertex cone generators to T_i. If $\rho_{ij}^* < 0$, we go to Step 2.2 to update φ_{ij}^*. In this step, we may find an exact preference relation between X_i and X_j. In this case, we do not include alternative X_i as a cone generator since it provides redundant information (see Theorem 1). Therefore, in case $\varphi_{ij}^* \leq 0$, we go to Step C.3 in Algorithm *Cone_update*. At each execution of Algorithm *Cone_update*, we identify index pairs (i, j) of alternatives whose p_{ij}^* values have the potential to be updated based on the cones and place them in set Q. At the end of Step 2, we have the ρ_{ij}^* and φ_{ij}^* values for all $(i, j) \in Q$. In Step 3, we set the shortest radial distance-to-domination (α_{ij}) and the shortest radial distance-to-dominate (β_{ij}) values for all alternative pairs. The shortest radial distance-to-domination equals either ω_{ij}^* (if X_j is not a vertex of a cone) or the maximum of ω_{ij}^* and ρ_{ij}^* (if X_j is the vertex of a cone). Similarly, the shortest radial distance-to-dominate equals either γ_{ij}^* (if X_j is not a vertex of a cone) or the minimum of γ_{ij}^* and φ_{ij}^* (if X_j is the vertex of a cone). In this step, we start from scratch as if there is no relation between X_i and X_j, even if α_{ij} or β_{ij} had been updated in previous iterations, as those values could be obsolete due to recent preference information. To demonstrate this, assume that X_j was a nonvertex generator of a cone at some iteration for which X_i was not a cone generator. In this case, assume that α_{ij} value was set using the said cone. In the current iteration, with new preference information X_i might also become a nonvertex generator of the same cone. In this case, the old cone information cannot be used to set the α_{ij} value and thus we set α_{ij} to its initial value, ω_{ij}^*.

In Step 4, we check if we can find a better estimate for the shortest radial distance-to-domination. We demonstrate how we update α_{ij} in Fig. 3.

Suppose that, at an iteration in the interactive algorithm, we only have the cone information $T_h = \{f, h\}$. Since X_j dominates a convex combination of the cone generators of C_h, Model 6 results

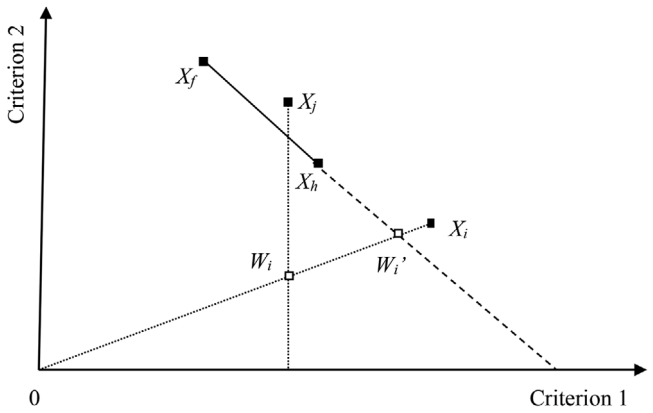


Fig. 3. Illustration of Algorithm *Qsq*—Step 4.

in $\varphi^*_{jh} < 0$. Therefore, we set p^*_{jh} to 1 and p^*_{hj} to 0. However, we do not include X_j as a cone generator to C_h since adding this implied preference leads to redundant information (see Theorem 1).

Suppose we try to find p^*_{ij} next. According to Model 3, the closest point to X_i in the radial direction that is dominated by X_j is W_i. However, the closest point to X_i in the radial direction that is dominated by cone C_h is W_i', and we update the shortest radial distance-to-domination, α_{ij}, to $\frac{W_i'X_i}{OX_i}$. In Step 4, we use available information to update the shortest distance-to-domination using all available preferences together with all dominance and cone-dominance relations.

In Step 5 of Algorithm *Qsq*, we update all preference measures.

3.2.3. Using the interactive algorithm for quasiconcave value functions

We next consider each step of the interactive algorithm and explain the specifics for implementing it for quasiconcave value functions. Steps 1 and 2 are same as the linear case but Steps 0 and 3 use information derived for quasiconcave value functions.

Algorithm *IntQsq*

Step 0. Initialization.

Set the values of ε and limit. Set $T_j = \{j\}$, $j \in N$. Set $p^*_{ii} = 0.5$, $i \in N$. For all (i, j), $i \neq j$, solve Models 3 and 4. Set $\alpha_{ij} = \omega^*_{ij}$, $\beta_{ij} = \gamma^*_{ij}$, and $p^*_{ij} = \frac{-\alpha_{ij}}{\beta_{ij}-\alpha_{ij}}$.

Step 1. Total order.

Solve Model 1 to find a strict total order. If the p^*_{ij} values of at least ε% of the consecutively-ranked alternatives are greater than *limit,* stop. That is, if at least ε% of $p^*_{R_k R_{k+1}}$, $k = 1, \ldots, n-1$, are greater than *limit,* stop. Otherwise, go to Step 2.

Step 2. Selection.

Identify alternative pairs whose relative preference measures are inconsistent with their relative ranks in the produced strict total order in Step 1 and place them in set I, i.e., $I = \{(i, j) | p^*_{ij} > p^*_{ji}, O_i > O_j\}$. If $I \neq \emptyset$, let $(i^*, j^*) = argmax_{(i,j)\in I}\{|p^*_{ij} - p^*_{ji}|\}$. If $I = \emptyset$, let $(i^*, j^*) = argmin_{i=1,\ldots,n, j=1,\ldots,n, i\neq j}\{|p^*_{ij} - p^*_{ji}|\}$.

Step 3. Comparison.

Ask for comparison between alternatives X_{i^*} and X_{j^*}, denote the index of the preferred and inferior alternatives as r and s, respectively. Update P^* using *Algorithm Qsq*. Go to Step 1.

3.3. Differences from existing approaches

For the linear underlying value function case, we utilize Büeler et al.'s (2000) approach as in Kim et al. (2006) to calculate the relative hypervolumes of the weight space between pairs of alternatives. Kim et al. (2006) use the calculated values in a totally different context. They develop a measure based on relative hypervolumes to compare the qualities of different solution sets. In contrast, we use relative hypervolumes to construct an exact total order of alternatives. We create mechanisms for obtaining preference information progressively to speed up the convergence of our interactive algorithm and use the available preference information to derive all implied preferences exploiting the properties of a linear underlying value function.

For the quasiconcave underlying value function, we utilize some of the procedures developed by Dehnokhalaji et al. (2014). Algorithm *Qsq* is a variation of Algorithm (*i,j*) of Dehnokhalaji et al. (2014). Algorithm *Qsq* is invoked by our interactive algorithm to revise P^* every time new preference information is obtained. Algorithm (*i,j*) is static and creates a preference matrix corresponding to a set, P, of available preferences. In Algorithm (*i,j*), relationships between all alternative pairs are extracted based on preference and cone-dominance information. In Algorithm *Qsq*, we first update the preference cones utilizing the new preference information. Then we reevaluate and update the preference measures of all alternative pairs that could have been influenced by the new preference information. Another difference is that Algorithm *Qsq* utilizes the cone-dominance information more effectively than Algorithm (*i,j*) does. More specifically, in Algorithm (*i,j*), additional information for α_{ij} only comes from the cones for which X_j is a nonvertex generator, whereas Algorithm *Qsq* considers a more general case by considering the dominance or cone-dominance relations of X_j with other alternatives as well. This allows Algorithm *Qsq* to update more distance-to-domination values compared to Algorithm (*i,j*). Additionally, we try to choose the alternative pair for the DM to compare in such a way to improve the efficiency of the algorithm.

3.4. Scaling and setting the parameters

The alternatives are evaluated by their criteria values, and different criteria typically have different ranges; in some cases the differences could be substantial. In order to imply meaningful information, these criteria should be scaled to have similar ranges. Scaling could be done by normalizing the values using the minimum and maximum values of the nondominated points.

The termination condition parameters, ε and *limit,* can be set by the DM before running the algorithm. If the DM wishes to guarantee obtaining the exact total order he/she should set the values to 100 and 1, respectively. This results in a preference matrix with all indices 0 or 1 for $i \neq j$ at termination. Alternatively, the algorithm can be terminated at an intermediate step with ε and *limit,* values less than 100 and 1, respectively. Higher values of ε and *limit* will result in rankings closer to the exact total order at the expense of more computations and preference information.

4. Computational results

We present our results in three parts. First, we compare the results of our algorithms *IntLin* and *IntQsq* with those of Dehnokhalaji et al. (2014; we refer to their algorithm as *D-et al.*). In the second part, we consider the progress of our algorithms' rankings as we increase the number of comparisons. Finally, we evaluate the results of varying the termination conditions on randomly generated problems.

4.1. Comparison with D-et al.

In *D-et al.*, they test the strict total order for four different cases: 10- and 20-alternative problems, each with 3 and 4 criteria. For each case, they generate 25 different sets of preferences obtained from the DM (simulated by an underlying value function), each set having either 9 or 15 pairwise comparisons for 10-alternative problems and 20 or 40 pairwise comparisons for 20-alternative problems.

We solve the same four problems with *IntQsq* using the same underlying value functions to be maximized (Dehnokhalaji et al., 2014):

Three criteria:

Linear: $f(X) = f(x_1, x_2, x_3) = 0.5x_1 + 0.3x_2 + 0.2x_3$.

Quadratic: $f(X) = f(x_1, x_2, x_3) = -\frac{(1.2x_1-1000)^2+(x_2-1000)^2+(x_3-1000)^2}{1000}$.

Four criteria:

Linear: $f(X) = f(x_1, x_2, x_3, x_4) = 0.3x_1 + 0.25x_2 + 0.25x_3 + 0.2x_4$.

Quadratic: $f(X) = f(x_1, x_2, x_3, x_4) = -\frac{(1.2x_1-1000)^2+(x_2-1000)^2+(x_3-1000)^2+(0.6x_4-1000)^2}{1000}$.

We implemented our algorithm in C++ using CPLEX for Models 1–6 setting ε to 100 and *limit* to 1.0.

We terminate *IntQsq* after exactly the same number of comparisons used in *D-et al.* for each case, to have a fair comparison between the two algorithms. We test the correlation between the exact total order and the order we obtain at termination using Kendall's Tau (Daniel, 1990, p. 365). Kendall's Tau (τ) values range between -1 and 1, where bigger values indicate better match with the exact total order. We report the results of *D-et al.* and *IntQsq* in Table 1. In the fifth column, we present 95% confidence intervals of the rank correlations of 25 different preference sets (reported by Dehnokhalaji et al., 2014). The corresponding results when the same problems are solved with *IntQsq* are given in the sixth column. Only for the 10-alternative problem with 3 criteria and 9 comparisons, the τ value of *IntQsq* falls inside the 95% confidence interval of Dehnokhalaji et al.'s (2014) results. Except this case, the rank correlations of *IntQsq* are larger than the upper bounds of the 95% confidence intervals in all cases. For the case of 15 comparisons and 10 alternatives with 3 and 4 criteria, *IntQsq* already finds the exact order before the 15th comparison, and either terminates before the 15th comparison or continues since all p^*_{ij} values are not equal to 0 or 1 until the 15th comparison.

Table 1
Rank correlations (τ) between the true and generated total orders—*IntQsq*

Number of criteria	Number of alternatives	Number of pairwise comparisons	Value function	*D-et al.*—95% confidence interval for rank correlations	*IntQsq*—rank correlations
3	10	9	Linear	[0.653,0.775]	0.689
			Quadratic	[0.663,0.743]	0.867
		15	Linear	[0.766,0.825]	1.000
			Quadratic	[0.782,0.845]	1.000
	20	20	Linear	[0.621,0.683]	0.737
			Quadratic	[0.610,0.667]	0.695
		40	Linear	[0.758,0.802]	0.905
			Quadratic	[0.788,0.828]	0.842
4	10	9	Linear	[0.740,0.812]	0.911
			Quadratic	[0.695,0.775]	0.867
		15	Linear	[0.787,0.864]	1.000
			Quadratic	[0.751,0.826]	1.000
	20	20	Linear	[0.589,0.673]	0.758
			Quadratic	[0.609,0.672]	0.726
		40	Linear	[0.761,0.819]	0.926
			Quadratic	[0.783,0.827]	0.874

Table 2
Rank correlations (τ) between the true and generated total orders—*IntLin*

Number of criteria	Number of alternatives	Number of pairwise comparisons	*D-et al.*—95% confidence interval for rank correlations	*IntLin*—rank correlations
3	10	9	[0.653,0.775]	1.000
		15	[0.766,0.825]	1.000
	20	20	[0.621,0.683]	1.000
		40	[0.758,0.802]	1.000
4	10	9	[0.740,0.812]	0.911
		15	[0.787,0.864]	1.000
	20	20	[0.589,0.673]	0.979
		40	[0.761,0.819]	1.000

Had we known the underlying value function is linear, we could have used our *IntLin* algorithm in order to converge faster. To observe the results under these conditions, we also solve the problems that are evaluated with a linear underlying value function using *IntLin*. In order to make the hypervolume computations over the weight space with its built in functions, we implemented the algorithm in MATLAB in this case. We call CPLEX from MATLAB whenever we need to solve Models 1 and 2. We report the results in Table 2. For 3-criteria problems, *IntLin* terminates in 8 and 15 comparisons for 10- and 20-alternative problems, respectively, after obtaining 0 or 1 value for each off-diagonal element of P^*. Hence, it guarantees that the resulting order is the exact total order. The results are similar for 4-criteria test problems. For these problems, we find the exact total orders in 13 and 22 questions for 10- and 20-alternative problems, respectively. The τ values are very close

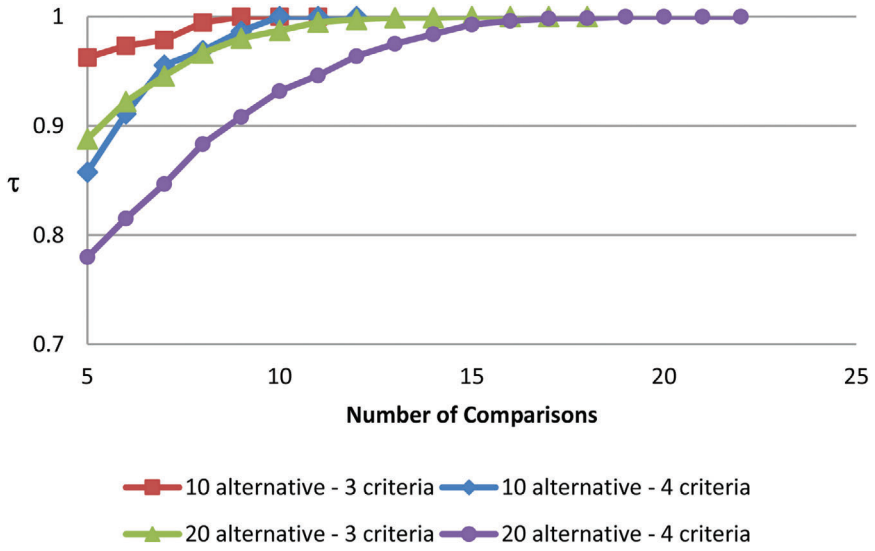


Fig. 4. Progress of τ values with comparisons—*IntLin*.

to 1.0 in 9 questions for 10-alternative, and in 20 questions for 20-alternative problems. Comparing the results with those of *D-et al.*, we observe that our rank correlations are always larger than the upper bounds of the 95% confidence intervals of the corresponding rank correlations obtained from *D-et al.*

The results show that the interactive algorithms perform well on all problems, approximating the total order within a reasonable number of comparisons.

4.2. *Progress of τ with number of comparisons in the interactive algorithm*

In this section, we show the test results on the progress of τ values as we acquire more information from the DM. Specifically, we monitor the τ values for the ranking obtained each time the DM makes a comparison. For this purpose, we generate 25 different 10- and 20-alternative sets with both 3- and 4-criteria. The criteria scores are generated from a discrete uniform distribution between 1 and 100. For all alternative sets, we make sure that all alternatives are nondominated and the underlying value function scores are distinct. To achieve these, we discard the generated alternative if it is dominated by or its value score is identical to a previously generated alternative.

We use the same underlying linear and quadratic value functions as in the previous section to simulate the responses of the DM.

The results for *IntLin* can be seen in Fig. 4. We start reporting the rank correlations at the fifth comparison as there is a need for a minimum amount of preference information to produce a meaningful order. For 10-alternative, 3- and 4-criteria problems, the algorithm terminates in at most 11 and 12 comparisons, respectively. The corresponding values are 18 and 22, for 20-alternative, 3- and 4-criteria problems, respectively. The average τ values are above 0.75 at the fifth comparison for all problems and monotonically increase with each additional comparison, approaching to 1 quickly.

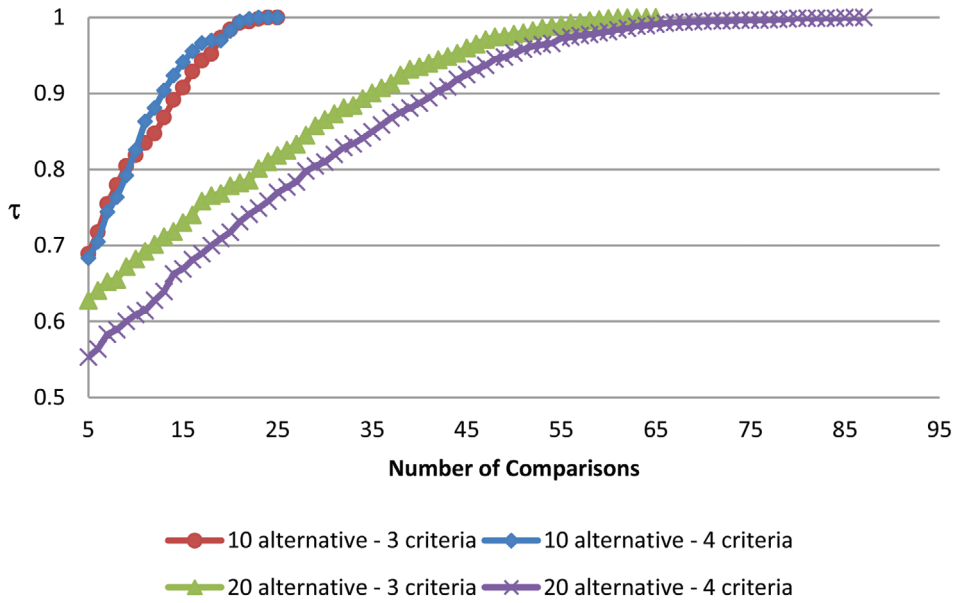


Fig. 5. Improvement in τ values with comparisons—*IntQsq*.

We find the rankings of the same alternative sets using *IntQsq*. We use the quadratic underlying value functions given in the previous section to simulate the preferences of the DM. The average τ values with varying number of comparisons are given in Fig. 5. As in the linear case, the average τ values increase monotonically with each comparison and reach above 0.9 after 15, 13, 35, and 42 comparisons for 10-alternative 3-criteria, 10-alternative 4-criteria, 20-alternative 3-criteria, and 20-alternative 4-criteria problems, respectively.

4.3. Results for different termination parameters

The termination of the interactive algorithm is based on two parameters: *limit* (the minimum p_{ij} value of consecutively ranked alternatives) and ε (the minimum percentage of the consecutively ranked alternatives satisfying *limit*). If ε and *limit* are set to 100 and 1.0, respectively, then the algorithm guarantees to find the exact total order.

In this section, we solve the 25 problem sets explained in the previous section, each time terminating the algorithm using $\varepsilon = 100$ and varying the *limit* values between 0.6 and 1.0. We do not consider smaller *limit* values than 0.6, since smaller p_{ij} values are not meaningful to indicate preference of X_i over X_j.

We present the results for linear and quasiconcave underlying value functions in Tables 3 and 4, respectively. For all cases, the average rank correlations are greater than 0.90. If $\tau = 0.90$ is acceptable (which gives a ranking very close to the exact ranking), it is sufficient to set $\varepsilon = 100$ and *limit* $=$ 0.6 in general. For these termination parameters, the number of comparisons is rather small in all cases.

Table 3
Rank correlations (τ) between the true and generated total orders and number of comparisons made if the algorithm terminates when 100% of the consecutive $p^*_{R_k, R_{k+1}}$ values are above *limit—IntLin*

	Average no. of comparisons				Average rank correlations			
	3-Criteria		4-Criteria		3-Criteria		4-Criteria	
limit	10 Alternatives	20 Alternatives	10 Alternatives	20 Alternatives	10 Alternatives	20 Alternatives	10 Alternatives	20 Alternatives
0.6	5.36	10.44	6.84	14.92	0.964	0.994	0.929	0.991
0.7	6.56	11.28	8.52	16.44	0.991	1.000	0.968	0.998
0.8	7.04	11.84	9.20	16.96	0.995	1.000	0.989	1.000
0.9	7.60	12.16	9.80	17.36	0.998	1.000	0.995	1.000
1.0	8.24	13.20	10.88	19.16	1.000	1.000	1.000	1.000

Table 4
Rank correlations (τ) between the true and generated total orders and number of comparisons made if the algorithm terminates when 100% of the consecutive $p^*_{R_k, R_{k+1}}$ values are above *limit—IntQsq*

	Average no. of comparisons				Average rank correlations			
	3-Criteria		4-Criteria		3-Criteria		4-Criteria	
limit	10 Alternatives	20 Alternatives	10 Alternatives	20 Alternatives	10 Alternatives	20 Alternatives	10 Alternatives	20 Alternatives
0.6	12.88	38.96	15.48	48.08	0.909	0.939	0.968	0.948
0.7	14.92	41.84	16.68	52.48	0.945	0.952	0.986	0.976
0.8	16.44	45.48	17.48	55.64	0.979	0.975	0.998	0.992
0.9	17.92	49.16	18.16	57.48	0.995	0.994	1.000	0.999
1.0	19.04	51.44	18.44	58.20	1.000	1.000	1.000	1.000

5. Conclusions

Finding a total order of alternatives under multiple criteria is a difficult problem and may require a lot of preference information. In this study, we develop interactive algorithms to find a total order while keeping the preference information at a reasonable level. We develop the necessary theory and separate interactive algorithms for linear and quasiconcave underlying value function cases.

We test the performance of the algorithms on randomly generated instances with different characteristics. The results show that interactive algorithms work well on all these problems. We observe that acquiring preference information progressively, rather than using *a priori* preference information, yields good results in estimating the rankings. This is expected since the progressive approach utilizes the past information in determining what type of information to gather next. Both interactive algorithms show fast convergence to the exact total orders. The algorithm for linear underlying value functions reaches good results with less preference information, thanks to the more restrictive form of the value function. The average performances of both algorithms monotonically improve with additional preference information in our experiments, as expected. This is a desirable property

indicating the robustness of the algorithms. Furthermore, terminating the algorithms prematurely before guaranteeing the exact total orders causes only slight deviations from the exact total orders.

There could be several variations that can be considered in different steps of the algorithms and these variations may affect the convergence of the algorithms. An important decision is the pair for which the DM's preference is asked for in each iteration. Different variations of selecting these pairs may be tried as a future research. The method we employ in constructing the P^* matrix for quasiconcave underlying value functions measures radial distances as in data envelopment analysis. New measures to represent the distance-to-dominate and distance-to-domination values can be considered.

The interactive algorithms can be applied to other value functions and function-specific rules can be developed. The interactive algorithm developed for the linear case can be used as an approximation for the cases where the underlying value function is nonlinear. When the structure of the underlying value function is not known in advance, a combination of the two algorithms can be developed, and the answers of the DM may guide in selecting the algorithm to use.

References

Bowman, V.J., Colantoni, C.S., 1973. Majority rule under transitivity constraints. *Management Science* 19, 9, 1029–1041.

Büeler, B., Enge, A., Fukuda, K., 2000. Exact volume computation for polytopes: a practical study. *Polytopes—Combinatorics and Computation* DMV Seminar, Vol. 29. Birkhäuser, Basel, pp. 131–154.

Buğdacı, A.G., Köksalan, M., Özpeynirci, S., Serin, Y., 2013. An interactive probabilistic approach to multi-criteria sorting. *IIE Transactions* 45, 10, 1048–1058.

Daniel, W.W., 1990. *Applied Nonparametric Statistics*. PWS-KENT, Boston, MA.

Dehnokhalaji, A., Korhonen, P. J., Köksalan, M., Nasrabadi, N., Wallenius, J., 2011. Convex cone-based partial order for multiple criteria alternatives. *Decision Support Systems* 51, 2, 256–261.

Dehnokhalaji, A., Korhonen, P. J., Köksalan, M., Nasrabadi, N., Tezcaner Öztürk, D., Wallenius, J., 2014. Constructing a strict total order for alternatives characterized by multiple criteria: an extension. *Naval Research Logistics* 61, 2, 155–163.

Ekiz M. K., Tuncer Şakar, C., 2020. A new DEA approach to fully rank DMUs with an application to MBA programs. *International Transactions in Operational Research* 27, 4, 1886–1910.

Karsu, Ö., Morton A., Argyris N., 2018. Capturing preferences for inequality aversion in decision support. *European Journal of Operational Research* 264, 2, 686–706.

Kim, B., Gel, E. S., Fowler, J. W., Carlyle, W. M., Wallenius, J., 2006. Evaluation of nondominated solution sets for k-objective optimization problems: an exact method and approximations. *European Journal of Operational Research* 173, 2, 565–582.

Köksalan, M., Bilgin Özpeynirci, S., 2009. An interactive sorting method for additive utility functions. *Computers & Operations Research* 36, 9, 2565–2572.

Köksalan, M., Büyükbaşaran, T., Özpeynirci, Ö., Wallenius, J., 2010. A flexible approach to ranking with an application to MBA programs. *European Journal of Operational Research* 201, 2, 470–476.

Köksalan, M., Mousseau, V., Özpeynirci, S., 2017. Multi-criteria sorting with category size restrictions. *International Journal of Information Technology & Decision Making* 16, 1, 5–23.

Köksalan, M., Tuncer, C., 2009. A DEA-based approach to ranking multi-criteria alternatives. *International Journal of Information Technology & Decision Making* 8, 1, 29–54.

Köksalan, M., Ulu C., 2003. An interactive approach for placing alternatives in preference classes. *European Journal of Operational Research* 144, 2, 429–439.

Korhonen, P., 1986. A hierarchical interactive method for ranking alternatives with multiple qualitative criteria. *European Journal of Operational Research* 24, 2, 265–276.

Korhonen, P., Soismaa, M., 1981. An interactive multiple criteria approach to ranking alternatives. *Journal of the Operational Research Society* 32, 7, 577–585.

Korhonen, P., Wallenius, J., Zionts, S., 1984. Solving the discrete multiple criteria problem using convex cones. *Management Science* 30, 11, 1336–1345.

Sobrie, O., Mousseau, V., Pirlot, M., 2019. Learning monotone preferences using a majority rule sorting model. *International Transactions in Operational Research* 26, 5, 1786–1809.

Ulu, C., Köksalan, M.M, 2001. An interactive procedure for selecting acceptable alternatives in the presence of multiple criteria. *Naval Research Logistics* 48, 7, 592–606.

Ulu, C., Köksalan, M., 2014. An interactive approach to multicriteria sorting for quasiconcave value functions. *Naval Research Logistics* 61, 6, 447–457.