Learning Value Functions in Interactive Evolutionary Multiobjective Optimization

Jürgen Branke *Member, IEEE,* Salvatore Greco, Roman Słowiński *Senior Member, IEEE*, Piotr Zielniewicz

***Abstract*—This paper proposes an interactive multi-objective evolutionary algorithm (MOEA) that attempts to learn a value function capturing the users' true preferences. At regular intervals, the user is asked to rank a single pair of solutions. This information is used to update the algorithm's internal value function model, and the model is used in subsequent generations to rank solutions incomparable according to dominance. This speeds up evolution towards the region of the Pareto front most desirable to the user. We take into account the most general additive value function as a preference model, and we empirically compare different ways to identify the value function that seems to be most representative with respect to the given preference information, different types of user preferences, and different ways to use the learned value function in the MOEA. Results on a number of different scenarios suggest that the proposed algorithm works well over a range of benchmark problems and types of user preferences.**

***Index Terms*—Evolutionary Multiobjective Optimization, Interactive Procedure, Preference Learning, Ordinal Regression**

I. Introduction

REAL life decision problems usually involve consideration of multiple conflicting objectives. For example, in portfolio optimization, one would like to maximize return but minimize risk, and in resource-constrained project scheduling, one might want to minimize project duration and resource consumption. In such cases, usually there is no single solution which simultaneously optimizes all objectives. Instead, without any additional preference information, there is a set of Pareto-optimal[1] solutions (also called Pareto front) which have to be considered equivalent.

Multi-objective Evolutionary Algorithms (MOEAs) usually try to approximate the entire Pareto front. This allows the decision maker (DM, sometimes also called user) to look at all the generated solutions and identify the most preferred. However, it may be beneficial to integrate preference information into the MOEA for the following reasons.

1) Instead of a diverse set of solutions, many of them clearly irrelevant to the DM, a search bias based on the DM's partial preferences will provide a more suitable sample of all Pareto optimal alternatives. It could either be a smaller set of only the most preferred solutions, or a more fine-grained resolution of the most preferred parts of the Pareto front.
2) By focusing the search onto the most preferred part of the objective space, we expect the optimization algorithm to find these solutions more quickly.
3) As the number of objectives increases, it becomes more and more difficult to identify the complete Pareto front. This is partly because of the increasing number of Pareto optimal solutions, but primarily because with an increasing number of objectives, an increasing portion of all feasible solutions becomes non-dominated, rendering dominance as selection criterion useless [1]. Partial user preferences re-introduce the necessary selection pressure.

Based on the above considerations, the goal is no longer to generate a good approximation of all Pareto optimal solutions, but a small set of solutions that contains the DM's preferred solution with the highest probability.

If the DM is involved in the multiobjective optimization process, then the preference information provided by the DM can be used to focus the search on the most preferred part of the Pareto front. This idea stands behind Interactive Multiobjective Optimization (IMO) methods proposed a long time before Evolutionary Multiobjective Optimization (EMO) has emerged (see, e.g., [2], [3], [4]).

Recently, it became clear that merging the IMO and EMO methodologies should be beneficial for the multiobjective optimization process [5]. There are many ways to integrate user preferences into MOEAs. In this paper, we proposes the Necessary preference enhanced Evolutionary Multiobjective Optimizer (NEMO) framework, which combines evolutionary multiobjective optimization with an interactive procedure based on so-called ordinal regression (described in the next Section). Ordinal regression usually builds preference models compatible with preference information from holistic comparisons of solutions.

It has first been applied to EMO in a methodology called NEMO (*N*ecessary preference-based *E*volutionary *M*ultiobjective *O*ptimizer) presented in [6], [7]. But there are also other ways of combining EMO and ordinal regression, and here we propose to categorize the combinations into the following three variants:

- NEMO-0: a single compatible value function is used to rank solutions in the population. Note that there is usually more than one compatible value function, so the question arises (and is addressed in this paper) which one to pick.

J. Branke works at Warwick Business School, University of Warwick, United Kingdom (email: juergen.branke@wbs.ac.uk)

S. Greco works at the Department of Economics and Business, University of Catania, Italy, (email: salgreco@unict.it)

R. Słowiński is with the Institute of Computing Science, Poznań University of Technology, and the Systems Research Institute, Polish Academy of Sciences, Poland (email: roman.slowinski@cs.put.poznan.pl)

P. Zielniewicz works at the Institute of Computing Science, Poznań University of Technology, Poland (email: piotr.zielniewicz@cs.put.poznan.pl

[1]A solution is Pareto-optimal (also called efficient or non-dominated) if there is no other feasible solution which would be at least as good on all objectives while being strictly better on at least one objective.

- NEMO-I: the whole set of compatible value functions is considered and the dominance relation used in NSGA-II to rank solutions is replaced by the necessary preference relation of robust ordinal regression. This is the proposal put forward in [6], [7].
- NEMO-II: the whole set of compatible value functions is also considered, but differently from NEMO-I, the solutions in the population are ranked according to a score calculated as the max-min difference of values between a given solution and all other solutions in the population, for the whole set of compatible value functions.

The previously proposed NEMO-I approach, while theoretically very elegant, has two drawbacks. First, at least with the very flexible additive monotonic value function model used in [6], [7] and also here, a lot of preference information is required to learn a useful model. Second, it requires a substantial computational overhead, as $O(n^2)$ linear programs have to be solved in every iteration to rank the n individuals, restricting its practical use to problems with very expensive fitness function evaluations. NEMO-0 avoids both of these problems, but leaves the question of how to pick the most helpful value function among all compatible value functions. In this paper, we focus on NEMO-0, describe its methodology and the procedure, compare various ways of selecting a value function, benchmark it against competitive models from the literature and discuss empirical results.

The paper is organized as follows. The next section provides a brief introduction to ordinal regression, followed by an overview of existing interactive EMO/IMO hybrids in Section III. Section IV describes the basic steps of NEMO-0. Some empirical results are reported in Section V. The paper concludes with a summary and some ideas for future research.

II. Ordinal Regression

To explain ordinal regression, we consider a Multiple Criteria Decision Aiding (MCDA) problem (for a comphrehensive collection of state-of-the-art surveys see [8]) concerning a finite set of alternatives $A = \{a_1, \ldots, a_m\}$, evaluated on n criteria (called objectives in optimization) from family $F = \{g_1, \ldots, g_n\}$, with $g_i \colon A \to \mathbb{R}, i = 1, \ldots, n$. Let $I = \{1, \ldots, n\}$ denote the set of criteria indices. A criterion $g_i \in F$ can be related to preferences in the following ways:

- g_i can be a gain-type criterion, i.e., increasing with respect to the preferences, so that for any $a \in A$, increasing $g_i(a)$ will make a more preferred;
- g_i can be a cost-type criterion, i.e., decreasing with respect to the preferences, so that for any $a \in A$ increasing $g_i(a)$ will make a less preferred
- g_i can be a mixed-type criterion, i.e. increasing or decreasing with respect to the preferences in different parts of its value set

For the sake of simplicity, in the following we shall consider only gain-type criteria and cost-type criteria.

Let G_i denote the value set (scale) of criterion g_i, $i \in I$. Consequently, the Cartesian product of all G_i's,

$$G = \prod_{i=1}^{n} G_i,$$

represents the objective space, and $x \in G$ denotes a vector profile of an alternative in this space. We consider a weak preference relation $\succsim$ on X which means, for each pair of vectors, $x, y \in G$,

$$x \succsim y \;\Leftrightarrow\; \text{“}x \text{ is at least as good as } y\text{”}.$$

This weak preference relation can be decomposed into its asymmetric and symmetric parts, as follows,

1) $x \succ y \equiv [x \succsim y \text{ and not } y \succsim x] \Leftrightarrow$ “x is preferred to y”, and
2) $x \sim y \equiv [x \succsim y \text{ and } y \succsim x] \Leftrightarrow$ “x is indifferent to y”.

From a pragmatic point of view, it is reasonable to assume that $G_i \subseteq \mathbb{R}$, for $i = 1, \ldots, n$, which does not exclude G_i from being a number-coded ordinal scale. More specifically, we shall assume that the evaluation scale on each criterion g_i is bounded, such that $G_i = [\alpha_i, \beta_i]$, where α_i, β_i, $\alpha_i < \beta_i$ are the worst and the best (finite) evaluations in case g_i is a gain-type criterion, and, the best and the worst evaluations in case g_i is a cost-type criterion, respectively. Thus, $g_i : A \to G_i$, $i \in I$. Therefore, each alternative $a \in A$ is associated with an evaluation vector denoted by $\mathbf{g}(a) = (g_1(a), g_2(a), \ldots, g_n(a)) \in G$.

With respect to set A, taking into account criteria from G, many MCDA methodologies have been proposed to suggest to the DM answers to one of the following questions: (i) what is the subset of the best alternatives in A, (ii) how to assign alternatives from A to pre-defined and preference ordered classes, or (iii) how to rank the alternatives from A from the best to the worst.

Among many preference models considered in the literature, the most popular is some value function $U : G \to \mathbb{R}$ representing the weak preference relation $\succsim$ in the sense that, for all $x, y \in G$

$$U(x) \geq U(y) \text{ if and only if } x \succsim y.$$

Like other preference models, the value function U of a DM is unknown a priori. A direct elicitation of this function from a DM is counterproductive in real-world decision aiding situations because of the high cognitive effort required. Eliciting indirect preferences in the form of holistic pairwise comparisons of some reference or training alternatives is much less demanding of cognitive effort. This kind of preference information is given as decision examples. A reverse search of the preference model from decision examples is done by so-called *ordinal regression* (also called disaggregation-aggregation approach). The preference model found by ordinal regression is *compatible* with the given preference information, i.e., it restores the holistic pairwise comparisons made by the DM. Finally, it is used on the whole set A of alternatives in order to compare them and obtain a recommendation for the decision problem at hand, which within NEMO is the ranking of solutions of a multiple objective optimization problem and the selection of a set of solutions considered as the best.

The ordinal regression paradigm emphasizes the discovery of intentions as an interpretation of actions rather than as a priori position, which was called by March the posterior rationality [9]. It has been known for at least fifty years in

the field of multidimensional analysis. It is also concordant with the induction principle used in machine learning. This paradigm has been applied within the two main MCDA approaches: those using a value function as preference model [10], [11], [12], [13], and those using an outranking relation as preference model [14], [15]. This paradigm has also been used since the mid nineties in MCDA methods involving a new, third family of preference models - a set of dominance decision rules induced from rough approximations of holistic preference relations [16].

The most well known ordinal regression methodology is the UTA (UTilités Additives) method proposed by Jacquet-Lagrèze and Siskos [12], which aims at inferring one or more additive value functions from a given ranking on a reference set of alternatives A^R. The method uses linear programming to construct the function so that the ranking obtained on A^R is as consistent as possible with the given preferences.

The criteria aggregation model in UTA is assumed to be an additive value function of the following form: for any $a \in A$,

$$U[\mathbf{g}(a)] = \sum_{i=1}^{n} u_i[g_i(a)] \tag{1}$$

subject to monotonicity and normalization constraints:

$$\begin{cases} \sum_{i=1}^{n} u_i(\beta_i) = 1 \\ u_i(g_i(a)) \geq u_i(g_i(b)) \text{ if } g_i(a) > g_i(b) \\ \quad \text{and } g_i \text{ is a gain-type criterion,} \\ u_i(g_i(a)) \geq u_i(g_i(b)) \text{ if } g_i(a) < g_i(b) \\ \quad \text{and } g_i \text{ is a cost-type criterion,} \\ \qquad\qquad \forall a, b \in A, i = 1, \ldots, n \\ u_i(\alpha_i) = 0, \quad i = 1, \ldots, n. \end{cases} \tag{2}$$

where $u_i, i = 1, \cdots, n$, are monotonic real valued functions, named marginal value or utility functions, normalized between 0 and 1. α_i and β_i are the worst and the best considered values of criterion g_i in case of a gain-type criterion, and the best and the worst considered values in case of a cost-type criterion, respectively. In the original version of UTA, the marginal value functions are supposed to be piecewise-linear, with a pre-defined number of linear pieces. This assumption has been relaxed in [17], [18], [19], where u_i are allowed to be general monotonic.

Both the marginal and the comprehensive value functions have the monotonicity property which, in the case of the comprehensive value function, induces the following property: for any $a, b \in A$,

$$\begin{cases} U[\mathbf{g}(a)] > U[\mathbf{g}(b)] \Leftrightarrow a \succ b & \text{(preference)} \\ U[\mathbf{g}(a)] = U[\mathbf{g}(b)] \Leftrightarrow a \sim b & \text{(indifference)} \end{cases} \tag{3}$$

Since linear programming does not permit to deal with constraints expressing that a quantity should be strictly greater than another one, the first of the above constraints has to be rewritten as

$$U[\mathbf{g}(a)] \geq U[\mathbf{g}(b)] + \varepsilon \Leftrightarrow a \succ b \tag{4}$$

with ε being a small positive quantity.

The UTA method infers one additive value function U compatible with the preference of the DM, i.e. satisfying constraints (3), together with monotonicity and normalization constraints (2).

Of course, the existence of such a preference model assumes preferential independence of the criteria for the DM [20]. In case this is not satisfied, i.e. the preferences $\succsim$ of the DM cannot be represented by the value function (1), [12] suggests to select the value function U minimizing the sum of deviation errors in (3) or minimizing the number of ranking errors in the sense of Kendall or Spearman distance.

Usually, among the many sets of parameters of a preference model compatible with the preference information, only one specific set is used to give a recommendation on a set of alternatives. For example, among many value functions representing pairwise comparisons of some alternatives made by the DM, only one value function is finally used to recommend the best choice or rank the alternatives. In case only one instance of the compatible preference model is considered we shall speak of *Classical Ordinal Regression*.

In the literature, several methods have also been suggested for selecting one value function that can be considered as particularly representative and that we call in the following the representative value function:

- *MDVF:* [21] suggests to select the *Most Discriminating Value Function* (MDVF) being the one that maximizes the value of ε in (4).
- *MSCVF:* [22] suggests to select the *Minimal Slope Change Value Function* (MSCVF), that is the value function U that minimizes ρ satisfying constraints (1), (2) and (3) and constraints

$$\begin{cases} \frac{u_i(x_i^{A,k}) - u_i(x_i^{A,k-1})}{x_i^{A,k} - x_i^{A,k-1}} - \frac{u_i(x_i^{A,k-1}) - u_i(x_i^{A,k-2})}{x_i^{A,k-1} - x_i^{A,k-2}} \leq \rho \\ \frac{u_i(x_i^{A,k-1}) - u_i(x_i^{A,k-2})}{x_i^{A,k-1} - x_i^{A,k-2}} - \frac{u_i(x_i^{A,k}) - u_i(x_i^{A,k-1})}{x_i^{A,k} - x_i^{A,k-1}} \leq \rho \\ 3 \leq k \leq t_i \end{cases} \tag{5}$$

  where $x_i^{A,1} < x_i^{A,2} \ldots < x_i^{A,t_i}$ are the values assumed by alternatives $a \in A$ with respect to criterion $g_i, i = 1, \ldots, n$ (of course, $x_i^{A,1} = \alpha_i$ and $x_i^{A,t_i} = \beta_i$ in case of gain-type criterion, and $x_i^{A,1} = \beta_i$ and $x_i^{A,t_i} = \alpha_i$ in case of cost-type criterion).
- *MSVF:* As a new proposal, in our experiments we shall consider the *Maximal Sum of the Scores Value Function* (MSVF). This value function maximizes the sum of the scores assigned to alternatives from A, i.e., the value function U that maximizes $\sum_{a \in A} U[\mathbf{g}(a)]$.

Note that the MSCVF and MSVF approaches require to specify an ε for the constraints **??**. We simply identify the maximum compatible ε^m first with the LP termed PRVF below, and then use $\varepsilon = 0.001\varepsilon^m$ to determine the MSCVF or MSVF.

Since the choice of one among many sets of parameters of the preference model compatible with the preference information is rather arbitrary, *Robust Ordinal Regression* (ROR) has been proposed with the aim of taking into account all the

sets of parameters compatible with the preference information given by the DM [18], [17], [19].

As a result of considering the whole set of compatible instances of the preference model, ROR takes into account *necessary preference relation* that holds for two alternatives $a, b \in A$ if and only if a is at least as good as b according to *all* instances of the preference model compatible with the preference information.

The necessary preference relation can be considered as *robust* with respect to the preference information because a given pair of alternatives compares in the same way whatever the instance of the preference model compatible with the preference information.

Example. Let us suppose that there are four solutions, called a_1, a_2, a_3 and a_4, evaluated by four objective functions g_1, g_2, g_3 and g_4 to be minimized (cost-type criteria). For the sake of simplicity, we suppose that each objective function g_i can take one of four values for each solution a_h, i.e. $g_i(a_h) \in \{1, 2, 3, 4\}$, $i = 1, 2, 3, 4$; $h = 1, 2, 3, 4$. The performance matrix of the four solutions is shown in Table I.

TABLE I
PERFORMANCE MATRIX OF THE FOUR SOLUTIONS

	g_1	g_2	g_3	g_4
a_1	4	2	2	4
a_2	3	1	1	3
a_3	1	3	1	3
a_4	2	4	4	2

Now, suppose that the DM evaluates the solutions in accordance with her value function having the following form:

$$U_{DM}(a_i) =$$

$$Max\{(5-g_1(a_i))(5-g_4(a_i))+0.5, (5-g_2(a_i))(5-g_3(a_i))\}.$$

If for the DM a_i is preferred to a_j, then

$$U_{DM}(a_i) > U_{DM}(a_j),$$

while if a_i is indifferent with a_j, then

$$U_{DM}(a_i) = U_{DM}(a_j),\ i, j = 1, 2, 3, 4.$$

In our example, the DM has the following preferences on the set of the four solutions:

$$a_4 \succ a_1 \succ a_3 \sim a_2$$

We also assume that this preference information has been given gradually, in two iterations. In the first iteration, the DM provided the preference information in the following order:

1) solution a_4 is preferred to solution a_1,
2) solution a_1 is preferred to solution a_2,
3) solution a_4 is preferred to solution a_2.

We want to verify whether it is possible to represent preferences 1)-3) using an additive value function, i.e., a function that evaluates the solutions according to the following formula: for $a_h, h = 1, 2, 3, 4$,

$$U(a_h) = u_1(g_1(a_h))+u_2(g_2(a_h))+u_3(g_3(a_h))+u_4(g_4(a_h)),$$

where $u_i(\cdot), i = 1, 2, 3, 4$, is a non-increasing marginal value function and $U(a_h) > U(a_k)$ if a_h is preferred to a_k, $h, k = 1, 2, 3, 4$. In order to obtain one value function $U(\cdot)$ representing the above preferences, one has to solve the following Linear Programming (LP) problem (PRVF - Preference Representing Value Function)

$$\max \varepsilon$$

$$\text{(PRVF)} \begin{cases} u_i(g_i(a_h)) \geq u_i(g_i(a_k)) \\ \quad \text{if } g_i(a_h) < g_i(a_k), i = 1, 2, 3, 4;\ h, k = 1, 2, 3, 4 \\ U(a_4) \geq U(a_1) + \varepsilon \\ U(a_1) \geq U(a_2) + \varepsilon \\ U(a_4) \geq U(a_2) + \varepsilon \\ u_i(4) = 0 \quad i = 1, 2, 3, 4 \\ u_1(1) + u_2(1) + u_3(1) + u_4(1) = 1 \end{cases}$$

whose unknown variables are $u_i(g_i(a_h)), i = 1, 2, 3, 4$ and $h = 1, 2, 3, 4$, and ε. If the solution of (PRVF) is positive (as in our case where max $\varepsilon = 0.33$) there exist value functions $U(\cdot)$ representing the DM's preferences; we call them compatible value functions.

The most discriminating value function, denoted by $U^{MDVF}(\cdot)$, is that compatible value function which enhances the DM's preferences by maximizing the minimal difference between the values of solutions a_h and a_k, whenever a_h is preferred to a_k.

Note that the above PRVF problem is equivalent to the following (MDVF - Most Discriminant Value Function) problem:

$$\max \min_{h,k \in \{1,2,3,4\}: a_h \succ a_k} U(a_h) - U(a_k)$$

$$\text{(MDVF)} \begin{cases} u_i(g_i(a_h)) \geq u_i(g_i(a_k)) \\ \quad \text{if } g_i(a_h) < g_i(a_k), i = 1, 2, 3, 4, h, k = 1, 2, 3, 4 \\ U(a_4) \geq U(a_1) + \varepsilon \\ U(a_1) \geq U(a_2) + \varepsilon \\ U(a_4) \geq U(a_2) + \varepsilon \\ u_i(4) = 0 \quad i = 1, 2, 3, 4 \\ u_1(1) + u_2(1) + u_3(1) + u_4(1) = 1 \end{cases}$$

whose unknown variables are $u_i(g_i(a_h)), i = 1, 2, 3, 4$ and $h = 1, 2, 3, 4$. More precisely, $u_i^\star(g_i(a_h))$ and $\varepsilon^\star$ constitute an optimal solution of the PRVF problem if and only if $u_i^\star(g_i(a_h))$ form an optimal solution of the MDVF problem, and in this case

$$\varepsilon^\star = \max \min_{h,k \in \{1,2,3,4\}: a_h \succ a_k} U(a_h) - U(a_k).$$

The most discriminating value function $U^{MDVF}(\cdot)$ is presented in Table II. Here, the table is a complete representation of the value function. For a continuous scale, we would simply use linear interpolation to assign values to further alternatives. Table III presents values assigned to solutions a_1, a_2, a_3 and a_4 by the value function $U^{MDVF}(\cdot)$.

TABLE II
MOST DISCRIMINATING VALUE FUNCTION

$g_i(a) =$	1	2	3	4
$u_1(\cdot)$	0.67	0.67	0	0
$u_2(\cdot)$	0	0	0	0
$u_3(\cdot)$	0.33	0.33	0	0
$u_4(\cdot)$	0	0	0	0

TABLE III
VALUES ASSIGNED TO THE SOLUTIONS BY THE MOST DISCRIMINANT VALUE FUNCTION

	$u_1(g_1(a))$	$u_2(g_2(a))$	$u_1(g_1(a))$	$u_1(g_1(a))$	$U(a)$
a_1	0	0	0.33	0	0.33
a_2	0	0	0	0	0
a_3	0.67	0	0.33	0	1
a_4	0.67	0	0	0	0.67

The min-max slope change value function $U^{MSCVF}(\cdot)$, computed using a fixed threshold $\varepsilon = 0.1$, is given in Table IV and the values assigned by $U^{MSCVF}(\cdot)$ to the solutions are presented in Table V. The min-max slope change is $\rho^\star = 0.15$. Analogous values are shown in Tables VI and VII for the maximum sum of scores value function $U^{MSVF}(\cdot)$ computed again with a fixed threshold $\varepsilon = 0.1$.

TABLE IV
MIN-MAX SLOPE CHANGE VALUE FUNCTION

$g_i(a) =$	1	2	3	4
$u_1(\cdot)$	0.72	0.33	0.09	0
$u_2(\cdot)$	0	0	0	0
$u_3(\cdot)$	0.27	0.23	0.04	0
$u_4(\cdot)$	0	0	0	0

TABLE V
VALUES ASSIGNED TO THE SOLUTIONS BY THE MIN-MAX SLOPE CHANGE VALUE FUNCTION

	$u_1(g_1(a))$	$u_2(g_2(a))$	$u_1(g_1(a))$	$u_1(g_1(a))$	$U(a)$
a_1	0	0	0.23	0	0.23
a_2	0.09	0	0.04	0	0.13
a_3	0.72	0	0.27	0	1
a_4	0.33	0	0	0	0.33

TABLE VI
MAXIMUM SUM OF SCORES VALUE FUNCTION

$g_i(a) =$	1	2	3	4
$u_1(\cdot)$	0.55	0.55	0.35	0
$u_2(\cdot)$	0	0	0	0
$u_3(\cdot)$	0.45	0.45	0	0
$u_4(\cdot)$	0	0	0	0

TABLE VII
VALUES ASSIGNED TO THE SOLUTIONS BY THE MAXIMUM SUM OF - SCORES VALUE FUNCTION

	$u_1(g_1(a))$	$u_2(g_2(a))$	$u_1(g_1(a))$	$u_1(g_1(a))$	$U(a)$
a_1	0	0	0.45	0	0.45
a_2	0.35	0	0	0	0.35
a_3	0.55	0	0.45	0	1
a_4	0.55	0	0	0	0.55

Let us suppose now that in the second iteration, the DM has added a new piece of preference information:

4) s_4 is preferred to s_3.

Thus, we have to add the constraint $U(s_4) \geq U(s_3) + \varepsilon$ to problem (PRVF), obtaining problem (PRVF1). For (PRVF1) we obtain $\max \varepsilon = 0$. Thus, even though the DM provided accurate and consistent preference information, there is no additive value function that can represent all the four items of preference information given by the DM. In other words, even the additive preference model is not flexible enough to represent the user's preferences. In order to enable representation of the most recently given preference information, we have to remove some pairwise preferences given by the DM, starting from the oldest, until we are able to represent again the remaining preferences. In this example, it is sufficient to discard the pairwise preference comparisons 1) ("solution s_4 is preferred to solution s_1") and 2) ("solution s_1 is preferred to solution s_2"). The obtained value function represents pairwise comparisons 3) ("solution s_4 is preferred to solution s_2") and 4) ("solution s_4 is preferred to solution s_3"). The most discriminating value function $U^{MD}(\cdot)$ thus obtained is presented in Table VIII while Table IX gives the values assigned by the value function $U^{MD}(\cdot)$ to solutions s_1, s_2, s_3 and s_4.

TABLE VIII
MOST DISCRIMINATING VALUE FUNCTION REPRESENTING PAIRWISE COMPARISONS 3) AND 4)

	1	2	3	4
$u_1(\cdot)$	0.5	0.5	0	0
$u_2(\cdot)$	0	0	0	0
$u_3(\cdot)$	0	0	0	0
$u_4(\cdot)$	0.5	0.5	0	0

TABLE IX
VALUES ASSIGNED TO THE SOLUTIONS BY THE MOST DISCRIMINANT VALUE FUNCTION REPRESENTING PAIRWISE COMPARISONS 3) AND 4)

	$u_1(g_1(s))$	$u_2(g_2(s))$	$u_1(g_1(s))$	$u_1(g_1(s))$	$U(s)$
s_1	0	0	0	0	0
s_2	0	0	0	0.5	0.5
s_3	0.5	0	0	0	0.5
s_4	0.5	0	0	0.5	1

III. INTERACTIVE EVOLUTIONARY MULTIOBJECTIVE OPTIMIZATION

There are various ways in which user preferences can be incorporated into EMO, and over recent years, there has been a surge of publications on this topic. The methods can be grouped, for example, according to the kind of information asked from the DM, such as a reference point (e.g., [23], [24], [25]), maximal and minimal trade-offs [26], desirability functions [27] or pairwise comparisons of solutions (e.g., [28]).

Most papers assume that partial preference information is provided a priori, i.e., before the optimization run. In principle, however, all these preference-based MOEAs can be run in a quasi interactive fashion: The DM is asked for some preference information, then the MOEA is run and a set of solutions is returned upon which the DM is allowed to alter their preference information and the run is continued. In this case, the user is fully responsible for specifying and refining preference information. Apart from the solutions maintained from one iteration to the next, there is no learning or accumulation of knowledge in these algorithms. It can be expected that the algorithm will converge to the solutions corresponding to the final preference information provided, independent of the history of the search.

Deb and Chaudhuri [29] propose an interactive decision support system called I-MODE that implements an interactive procedure built over a number of existing EMO and classical decision making methods. The main idea of the procedure is to allow the DM to interactively focus on interesting region(s) of the Pareto front. The DM has several options to focus the search on the desired regions. For example, he/she may use a weighted sum approach, a value function based approach, or trade-off information. The preference information is then used by an MOEA to generate new solutions in the most preferred regions.

Truly interactive MOEAs try to learn from the interaction with the user, and accumulate preference information over time. Some examples include the Territory Defining Algorithm by Köksalan and Karahan [30] or the interactive MOEA/D by Gong et al. [31].

Since we propose an algorithm that attempts to learn the user's value function, the following will focus on previous work based on a similar line of thought. It is categorized into approaches that attempt to find a single value function (as usually in ordinal regression), and those that work with a set of value functions (as does robust ordinal regression).

A full survey is out of the scope of this paper, and the interested reader is referred to the corresponding survey papers [32], [33], [34], [35].

A. Learning a value function model

Some approaches use the elicited preference information to derive a single value function to represent the user preferences, similar to what is usually done in classical ordinal regression. Value function models can have different complexity, ranging from simple linear functions through general additive forms, to non-parametric approaches such as artificial neural networks or support vector machines. Most approaches simply use the derived value function for ranking individuals, sometimes as secondary criterion after non-dominance, but other uses can also be found.

Phelps and Köksalan [36] proposed an interactive evolutionary algorithm that periodically asks the DM to rank pairs of solutions. Assuming a **linear** model (actually, the objectives are modified before the optimization to the squared distance from a reference value, which effectively results in **ellipsoidal** indifference curves of the value function), these preferences are turned into constraints for possible weights. For example, if solution a is preferred over b and the objectives are to be minimized (cost-type criterion), it is clear that

$$\sum_{k=1}^{n} w_k(g_k(a) - g_k(b)) < 0. \tag{6}$$

The method determines the most discriminating (in the sense of MDVF) linear value function compatible with the preference information. As explained in Section II, the most discriminating value function can be obtained by solving a linear program (LP). The resulting value function is then used for ranking individuals in the evolutionary algorithm that works as a single objective evolutionary algorithm between user interactions. If the LP is overconstrained and no feasible solution is found, the oldest preference information is discarded. Since this method uses LP to determine the value function, it is computationally efficient. Its limiting factor is the restriction on linear models.

A very similar idea is used by Barbosa and Barreto [37], but instead of an LP, a second evolutionary algorithm is used to determine a compatible linear value function.

Deb et al. [38] derive a **polynomial** value function model. The user is shown a set of (five in the paper) evenly spread Pareto-optimal solutions and asked to (at least partially) rank them. Then, the most discriminating value function is determined. However, because a polynomial value function model is used, fitting the value function model to the specified preferences involves solving a non-linear optimization problem, and the authors propose to use sequential quadratic programming. Once a most discriminating value function has been identified, this information is used not directly for ranking, but for separating the objective space into two areas: All individuals with an estimated value (according to the approximated value function) better than the solution ranked *second* by the DM are assumed to dominate all the solutions with an estimated value worse than the solution ranked second. If both solutions lie either above or below the estimated indifference curves of the value function through the second best individual, they are compared based on the usual Pareto dominance. The authors additionally use the approximated value function to perform a local single-objective optimization starting with the solution ranked best by the DM. If this local improvement step is not able to improve the solution's value by at least a certain margin, it is concluded that the algorithm has found the most preferred solution and the optimization is stopped. The approach is particularly interesting for its innovative use of the value function. The drawbacks are the computational overhead involved in sequential quadratic programming, and the larger number of solutions to be considered by the DM in every step.

Todd and Sen [39] use **artificial neural networks** to represent the DM's value function. Periodically, they present the DM with a set of solutions and ask for a score between 0 and 1. The set of solutions is chosen such that they represent a broad variety regarding the approximated value function, in particular, the estimated best and worst individual of the population are always included. Information from several user interactions is accumulated after normalizing preference scores. While an artificial neural network is a very flexible value function model, it is again time-consuming to train, the resulting value function is not explicit (which might be useful, e.g., to communicate with the user), and a user may find it very difficult to score solutions (rather than rank them) consistently over the run.

Another model that allows to represent complex value functions are **support vector machines** (SVM). SVMs have the additional advantage of being able to trade-off model accuracy and model complexity. Battiti and Passerini [40] use SVMs in the setting of an interactive MOEA, more specifically NSGA-II. Periodically, the DM is presented with a set of solutions and asked to (at least partially) rank them. This information

is then used to train the SVM, with cross-validation employed to select an appropriate kernel. The derived approximate value function is then used to replace the crowding distance in NSGA-II by sorting individuals in the same non-dominance rank based on their value according to the learned value function. The solutions shown to the DM are the best according to the approximated value function, or randomly selected non-dominated solutions in the first step. The paper examines the influence of the number of solutions shown to the DM (assuming full ranking) and the number of interactions with the DM. The results suggest that a relatively large number of solutions need to be ranked for the SVM to learn a useful value function (around 10-20), but only two interactions with the DM seem sufficient to come very close to results that would have been obtained had the DM's true value function been known from the beginning. The authors recommend not starting interaction until the MOEA has found a reasonable coverage of the entire Pareto front, which somewhat defeats the purpose of narrowing down the search early on. In [41], the approach's robustness to incorrect (noisy) DM preferences is examined and it is shown that the algorithm can cope well with noise, in particular if the number of solutions ranked by the DM is large.

The NEMO-0 approach in this paper allows for a flexible value function model (additive monotonic) that can still be computed efficiently with LP. Also, while most of the above approaches rely on the MDVF principle (implicitly also SVM relies on this idea, albeit in a different space), we examine various alternatives and show MSVF actually to be a better choice.

B. Using a set of value functions

Rather than deriving a single value function, and analogous to robust ordinal regression, one may acknowledge that there are usually several value functions compatible with the elicited user preferences.

Jaszkiewicz [42] samples the preference function used in each generation from the set of compatible preference functions (in this case **linear** weightings are assumed). The proposed approach uses the value function also for local search. In the interactive version, preference information from pairwise comparisons of solutions is used to reduce the set of possible weight vectors.

Greenwood et al. [28] suggested the imprecise value function approach which considers *all* compatible **linear** value functions *simultaneously*. The procedure asks the user to rank a few alternatives, and from this derives constraints for the weightings of the objectives consistent with the given ordering. Then, to compare any two solutions a and b, simultaneously all linearly weighted additive value functions are considered which are consistent with the ordering on the initially ranked solutions. A preference of a over b is inferred if a is preferred to b for all such value functions. Again, this can be checked by linear programming. The approach can be seen as a simplified form of robust ordinal regression (cf. Section II), based on the assumption of simple linear value functions. Overall, the method requires to solve one or two LPs for each pair of solutions in the population. In order to make sure that there is at least one compatible value function the authors suggest to use a mechanism from [43] which removes a minimal set of the DM's preference statements to make the weight space non-empty.

Branke et al. [44] use the expected value as indicator in an indicator-based EA i.e., a solution is evaluated by the loss in expected value the DM is able to get if this solution would be absent from the population. To calculate the expected value, it is assumed that the DM has a linear value function of the form $U(a) = \lambda g_1(a) + (1-\lambda)g_2(a)$, and λ is unknown but follows a uniform distribution over $[0, 1]$. The expected marginal value (EMV) of a solution a is then the value difference between the best and second best solution, integrated over all value functions where solution a is best. Without preference information, the result of using this indicator is a natural focus of the search on so-called "knees", i.e., convex regions with strong curvature. In these regions, an improvement in either objective requires a significant worsening of the other objective, and such solutions are often preferred by DMs [45]. Additional explicit user preferences can be taken into account by allowing the user to specify the probability distribution for λ [32]. Obviously, any distribution over any space of value functions could be considered with this approach, and Wagner et al. [46] use similar ideas but Chebycheff value functions.

In the approach by Korhonen et al. [47], the value function model is only implicit. Under the assumption of **quasi-concave** value functions, specified preferences between solutions can be generalized to preference cones [47]. This idea is used by Fowler et al. [48] to partially rank the non-dominated solutions in an MOEA. The DM is asked to consider a set of six solutions and specify the best and worst. From this information, six preference cones are derived (five 2-point cones involving the best and any of the other solutions, and one 6-point preference cone specifying that five solutions are better than the worst). The solutions shown to the DM are selected from the set of non-dominated solutions that can not already be ranked with the existing cones.

Branke et al. [6], [7] proposed a method called NEMO (Necessary preference enhanced Evolutionary Multiobjective Optimizer) which we classify as NEMO-I approach in this paper. It uses pairwise comparisons of solutions to learn user preferences. Similar to the imprecise value function approach by Greenwood et al. [28], it simultaneously considers the set of *all* value functions compatible with the elicited preference information. But rather than being restricted to linear value functions, it allows for **piecewise-linear** [7] or **general monotonic additive** [6], [7] value functions. This is possible because it is based on robust ordinal regression (cf. Section II). It can take into account a preference ranking of solutions, such as "a is preferred over b", but also intensities of preferences, such as "a is preferred over b more than c over d". A solution a is necessarily preferred over solution b, if it is preferred according to *all* value functions compatible with the elicited preference information. As in [28], whether one solution is necessarily preferred over another can be detected by solving at most two linear programs. The main difference is that the variables are not weights, but values of characteristic points

of marginal value functions. NEMO-I replaces the use of the dominance relation in the non-dominance sorting step of NSGA-II by the necessary preference relation.

IV. The NEMO-0 approach

With all the theory of ordinal regression explained, it is easy to explain NEMO-0, the algorithm we focus on in this paper. NEMO-0 is based on NSGA-II [49]. That means it uses Pareto-non-dominance ranking as primary criterion to rank individuals. Non-dominance ranking ranks individuals by iteratively determining the non-dominated solutions in the population (non-dominated front), assigning those individuals the next best rank, and removing them from the population. The result is a partial ordering, favoring individuals closer to the Pareto front.

NEMO-0 differs from NSGA-II primarily by also making use of preference information from pairwise comparisons to order individuals within the same non-dominance rank. Every q generations (where q is a user-specified p arameter), the DM is asked to compare two solutions a and b, resulting in preference information (either $a \succ b$ or $b \succ a$). After the stage of preference elicitation, a representative value function compatible with all elicited preference information is computed as explained in Section II. As class of possible value functions, we chose the very flexible c lass o f a dditive monotonic functions. A representative value function of this type is then used as secondary criterion to order all individuals with the same non-dominance rank, thus replacing the crowding-distance calculation usually used in NSGA-II. The information from the pairwise comparison is appended to the previously provided preference information. If there is no value function compatible with the current preference information, we must discard the inconsistent information. The complete algorithm is outlined in Algorithm 1.

We made a number of design decisions.

- We examined all of the different ways to select a representative value function discussed in Section II. Their performance will be compared in Section V-A.
- Instead of the two-stage ranking using Pareto-non-dominance ranking and the value function as secondary criterion, we also considered using the inferred value function only. The benefit of the two-stage ranking is demonstrated in Section V-B).
- The selection of the solutions to be compared may have an impact on the information gained as well as the effort required from the DM. As several other papers before us, we here follow the simple approach of comparing randomly selected non-dominated solutions, but would like to point out that an intelligent selection mechanism may further enhance the performance not only of NEMO, but also other comparison-based interactive EMO algorithms.
- There are various possibilities to make sure there is always a compatible value function. One sensible way would certainly be to ask the user to resolve any inconsistencies in the preference information once they are detected. As we assume an artificial user, in case there is no value function compatible with the preference information, following Phelps and Köksalan [36], we simply discard the information from the oldest pairwise comparison provided by the user until the space of compatible value functions becomes again non-empty.

So, while NEMO-I considered the entire set of compatible value functions, NEMO-0 only determines a single representative value function and thus falls into the category of Section III-A. This makes it much faster than NEMO-I, which in each generation has to solve a number of LPs that is quadratic in the population size.

Compared to the other approaches in Subsection III-A, we use a very flexible value function model (arbitrary monotonic additive value functions), much more flexible than for example the linear model used by Phelps and Köksalan [36] or also the quadratic model used by Deb et al. [38]. On the other hand, we accumulate preference information over the run, and only require relatively little input from the user (comparison of pairs of solutions) while other approaches such as the ones by Battiti and Passerini [40] or Deb et al. [38] require to rank much larger sets of solutions.

Overall, NEMO-0 is most similar to the way Phelps and Köksalan [36] use preferences, which we primarily use as a benchmark in this paper. NEMO-0 differs by using a more flexible value function model (additive monotonic instead of linear), by examining different ways to determine the representative value function, and by using non-dominance ranking in addition to the representative value function.

V. Experimental results

An empirical evaluation of interactive EMO methods is challenging, because the test environment has to include a model of the user behavior. We use an artificial user which applies a pre-specified value function for decision making. Obviously, this value function is not known to NEMO, but only used by the artificial user (DM) for comparing two solutions when preferences are elicited.

As it is usual in EMO, we assume here without loss of generality that the objective functions $g_1, \ldots, g_n$ are to be minimized (cost-type criteria). Two user's value functions are used in our tests:

- **Linear** – assumes the artificial user has a value function that for the minimized objective functions is the opposite of a linear weighting of those objectives. More specifically, we use $U(a) = -((w_1 g_1(a) + \cdots + w_n g_n(a)))$ and, consequently, the user's goal is to minimize $U^-(a) = w_1 g_1(a) + \ldots + w_n g_n(a)$. The $w_1, \ldots, w_n$ parameters depend on the problem and are defined below.
- **Chebycheff –** assumes the artificial user has a value function that for the minimized objective functions and the reference point in zero is the opposite of a Chebycheff function of these objectives. More specifically, the user's goal is to maximize $U(a) = -\max\{w_1 g_1(a), \ldots, w_n g_n(a)\}$ which is equivalent to to minimize $U^-(a) = \max\{w_1 g_1(a), \ldots, w_n g_n(a)\}$. Again, the parameters $w_1, \ldots, w_n$ depend on the problem and are defined below.

Algorithm 1 Basic NEMO-0

Generate initial solutions randomly.
Elicit DM's preferences. {*Present to the DM a pair of non-dominated solutions and ask for a preference comparison*}
Determine the representative value function consistent with the preference information.
Determine the non-dominance ranking. {*As in NSGA-II*}
Within each non-dominance rank, sort individuals according to representative value function. {*The representative value function replaces the crowding distance in NSGA-II*}
repeat
  Mating selection and offspring generation.
  if Time to ask the DM **then**
    Elicit DM's preferences.
    Determine the representative value function.
  end if
  Determine non-dominance ranking.
  Within each non-dominance rank, sort individuals according to representative value function.
  Environmental selection.
until Stopping criterion met
Return all non-dominated solutions in the population.

For the sake of simplicity, with a slight abuse of the terminology, when we speak of a user's value functions we refer to their opposite forms, and thus we aim at minimizing the linear value function $U^-(a) = w_1 g_1(a) + \ldots + w_n g_n(a)$ or the Chebycheff-like value function $U^-(a) = \max\{w_1 g_1(a), \ldots, w_n g_n(a)\}$. When showing the plots of convergence (Figures 1-2, and Tables XII-XIII), the terms "Convergence indicator" and "Convergence curve" mean the value of $U^-(\cdot)$.

We look at two performance measures over time:

1) *Best-in-population value:* The value of the best solution in the final population. This assumes that the final population is returned to the DM, and the DM is able and willing to spend the effort to identify the most preferred solution among the set.
2) *Average population value:* The average value of all the solutions in the final population. This puts more emphasis on whether an algorithm was able to focus the search appropriately, as solutions with much lower value than the best one hurt performance.

NEMO-0 is set up such that in every k-th generation, it randomly selects two individuals from the current population, which do not dominate each other, and receives as feedback the preference relation for the selected pair of solutions by the artificial user acting according to the assumed value function. The information from the DM is then used to update the internal preference model. Additionally, we used a buffer of preferences to reduce the amount of accumulated preference information to at most 50 pairwise comparisons. The use of this buffer significantly reduces the computation time for more complex problems. The preference elicitation is performed every generation (elicit-interval=1), every 10 generations (elicit-interval=10) or every 50 generations (elicit-interval=50). A typical run of 300 generations takes between 4 seconds (for 2D problems and elicit-interval 10) to 15 seconds (for 4D problems and elicit-interval 1), i.e., the computational overhead is insignificant for most real-world applications.

All our results have been averaged over 100 independent runs. Since we do not expect the parameter settings of the algorithm to influence the relative performance of the various algorithms, we mostly rely on default settings. We use a real-valued representation, generate offspring by simulated binary crossover with crossover probability of 0.9 and $\eta_c = 1$, and Gaussian mutation with mutation probability 0.03 and step size $\sigma = 0.01$. Constraints are ensured by reflecting on the boundary, and mating selection is done by tournament selection. We run the algorithm for a pre-specified number of generations (problem dependent). The population size has been set to 32, a value smaller than usual in MOEA. However, we no longer aim at finding a set of solutions representative for the entire Pareto front, but only a single solution. Therefore, returning a large population size and asking the DM to look at such a large number of alternatives does not seem to be desirable.

We compared the efficiency of NEMO-0 with that of the preference model proposed by Phelps and Köksalan [36]. However, we did not use an exact re-implementation of [36], because we are primarily interested in the learning and use of the preference model. In our implementation of this algorithm, we therefore used only the fitness score proposed in [36], but the same selection and mutation operators in the evolutionary process as for NEMO-0 and NSGA-II. This algorithm is henceforth denoted by P&K.

In this section, we empirically investigate NEMO-0 and show the following.

1) We compare different definitions of what constitutes the best concept of representative value function for NEMO-0.
2) We show that NEMO-0 works for finding solutions in different regions of the Pareto front.
3) We show that NEMO-0 works for different types of user preferences.
4) We compare NEMO-0 with NSGA-II and the preference model proposed in [36] on a variety of benchmark functions and show that preference elicitation is particularly useful in many-objective problems.

A. Comparing different concepts of the representative value function

For an initial comparison of the considered concept of the representative value function for NEMO-0, we use ZDT1 (a simple 2-objective benchmark with convex Pareto front [50]) with linear value function and DTLZ2 (a concave Pareto front [51]) with 4 objectives (4D) with Chebycheff user value function. Each one is used once with a true preference function that has the most preferred solution in the middle of the Pareto front (Figure 1 (a) and (c) labelled "middle"), and once with a preference function that has the most preferred solution towards the lower right of the Pareto front (Figure 1 (b) and (d) labelled "extreme"). The parameters of the user's true value function are summarized in Table X. We compare the three different ways to determine the representative value function as discussed in Section II, i.e., minimum slope change (MSCVF), maximum discrimination (MDVF), and maximum sum of utilities (MSVF). The convergence plots for the value of the best solution in the population for these functions are presented in Figure 1.

TABLE X
PARAMETERS OF THE USER'S VALUE FUNCTION

	w_1	w_2	w_3	w_4
ZDT1 Linear (middle)	0.60	0.40	–	–
ZDT1 Linear (extreme)	0.15	0.85	–	–
DTLZ2-4D Cheb. (middle)	0.30	0.15	0.20	0.35
DTLZ2-4D Cheb. (extreme)	0.65	0.10	0.15	0.10

As can be seen, all concepts of the representative value function work more or less equally well if the DM's value function is linear and the Pareto front is convex. Also, the elicitation interval does not seem to have much influence, so it seems that very little preference information is needed to focus the search for such simple problems (Figure 1a,b). For DTLZ2-2D with Chebycheff user value function, more differences are visible. The minimum slope change value function (MSCVF) is clearly worse, in particular for larger elicitation intervals. Also the maximum discrimination concept (MDVF) with elicit-interval=10 struggles, and has a performance quite similar to NSGA-II.

For DTLZ2-4D, i.e., as soon as we move to more objectives, the differences become very large. The maximum sum of marginal utilities as representative value function (MSVF) performs best for both settings of the user preferences, and for both settings of the elicitation interval. The maximum discrimination concept (MDVF) sometimes also works quite well, although seems to get stuck at suboptimal solutions for elicit-interval=10 and struggles with the case where the most preferred solution is an extreme solution. NSGA-II is quite slow in the beginning and seems to converge to a much worse value level than most of the other approaches. The minimum slope change idea (MSCVF) is worst, and even worse than NSGA-II for the larger elicitation interval. Its curve also seems more erratic than the others, often deteriorating substantially after an initial phase of "normal" convergence. The reason may be that penalizing slope changes leads to mostly linear value function approximations that struggle with the concave Pareto front in DTLZ2 and sometimes lead the population away from the most preferred solution.

In all cases except the simplest ones, the runs with elicit-interval=10 are substantially slower than the corresponding variant with elicit-interval=1, demonstrating the benefit preference information can have in speeding up the optimization.

In the following experiments, we use only the best performing maximum sum of marginal values concept (MSVF) to determine the representative value function.

B. Benefit of two-tier ranking

The representative value function computed provides a complete ordering on the population. This is what Phelps and Köksalan [36] use for selection. NEMO-0 instead uses a two-tier ranking, with standard non-dominance Pareto ranking as first criterion (as in NSGA-II), and the representative value function is only used to rank individuals with the same non-dominance rank. The reason is that the representative value function represents only one out of potentially many value functions compatible with the provided preference information, and completely relying on it may temporarily misguide the search. As can be seen in Figure 2, this two-tier ranking is essential for NEMO-0. If the representative value function is used as the only ranking tool, convergence suffers substantially, in particular in cases with little preference information (elicit-interval=10).

C. Robustness with respect to user preference

In the next set of experiments we use ZDT1 with linear and ZDT4 with Chebycheff user value function, each with three different preference parameters displayed in Table XI, to show how quickly and how accurately the population focuses on the corresponding areas of the Pareto front.

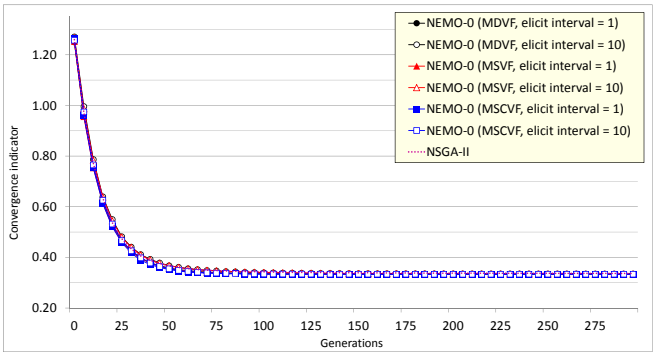
TABLE XI
PARAMETERS OF THE USER'S VALUE FUNCTION

	w_1	w_2
ZDT1 Linear (middle)	0.60	0.40
ZDT1 Linear (extreme-1)	0.15	0.85
ZDT1 Linear (extreme-2)	0.85	0.15
ZDT4 Chebysheff (middle)	0.60	0.40
ZDT4 Chebysheff (extreme-1)	0.20	0.80
ZDT4 Chebysheff (extreme-2)	0.95	0.05

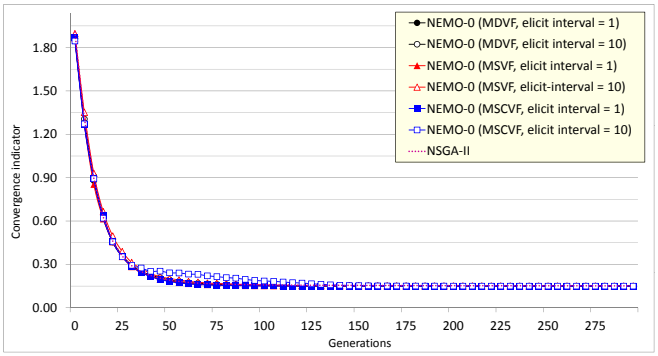
Figure 3 shows the typical result of NEMO-0 after 50, 100, 200 and 300 generations. The Pareto front is marked yellow. As can be seen, NEMO-0 very quickly converges to the user-preferred solution on the Pareto front. For the more difficult ZDT-4 function, NEMO-0 reaches a similar value after 200 generations that NSGA-II reaches after 300 generations, confirming that incorporation of preference information can not only focus the search on the most interesting region, but also get there more quickly.

D. Comparison of NEMO-0 with other approaches

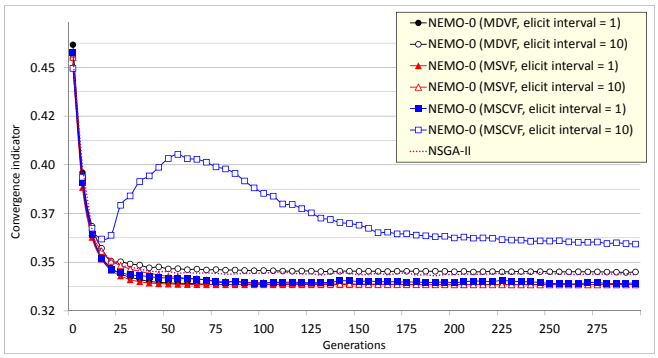
In this section, we compare our approach with NSGA-II and Phelps and Köksalan's preference model (P&K). We consider the following problems: ZDT2 and ZDT4 in 2D, DTLZ2 in



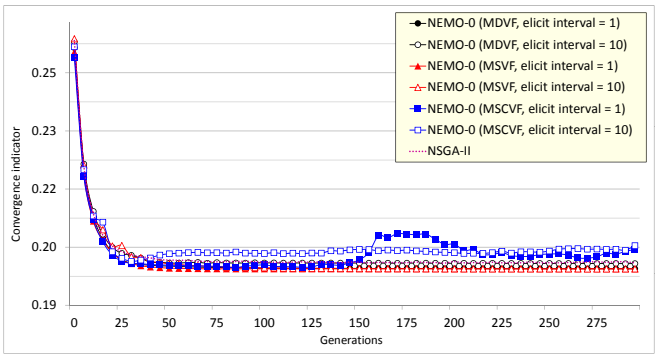
(a) ZDT1-2D Linear (middle)



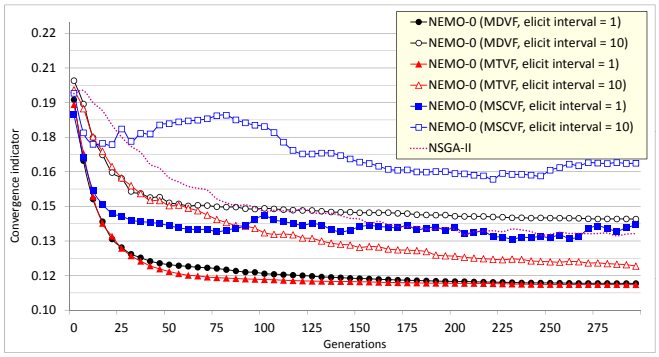
(b) ZDT1-2D Linear (extreme)



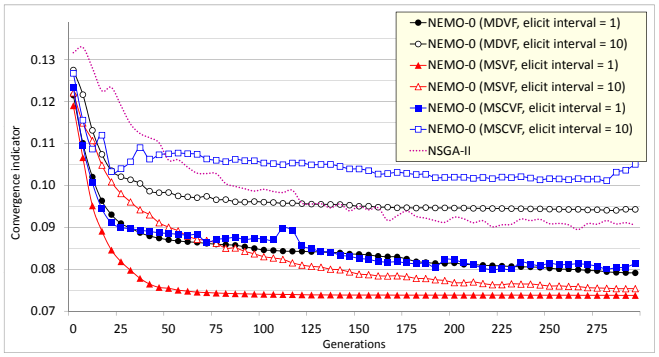
(c) DTLZ2-2D Chebyshev (middle)



(d) DTLZ2-2D Chebyshev (extreme)



(e) DTLZ2-4D Chebyshev (middle)



(f) DTLZ2-4D Chebyshev (extreme)

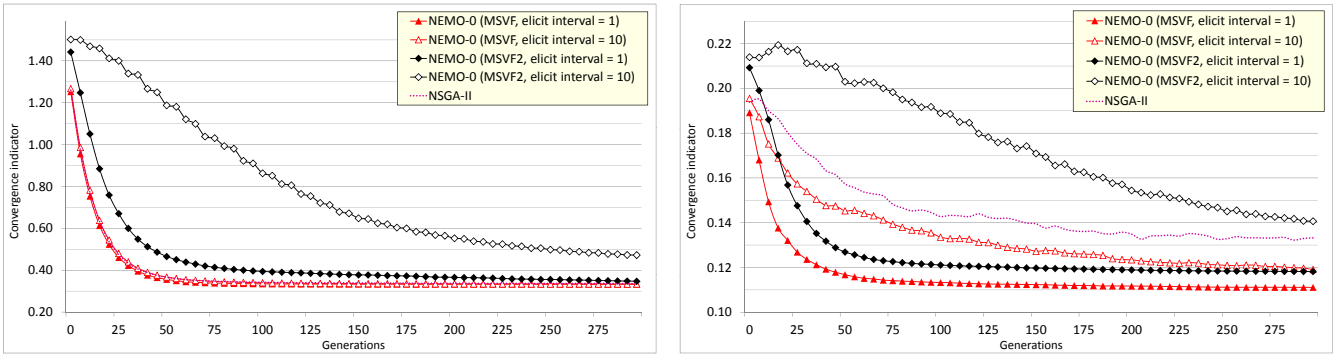
Fig. 1. Comparison of different concepts of the representative value function, for various test problems and user preferences. Plots show best-of-population value vs. generations.

3D, DTLZ2 in 4D and WFG1 in 5D (for a description of the latter one, see [52]), all with Chebycheff user value function.

Rather than comparing the value obtained after a specific number of generations, we calculate the area under the convergence curve (average over generations) as a measure of the overall performance of the algorithm. For WFG-5D, we ran the algorithm for 15,000 generations, while for all other problems, the limit was 300 generations. We report on the area under the convergence curve for NEMO-0 and P&K relative to the area under the curve for NSGA-II for all tested functions (i.e., values > 1 mean worse performance than NSGA-II, while values < 1 mean better performance than NSGA-II). These results are presented in Table XII (for the value of the best individual in the population) and Table XIII (for the average value of individuals in the population).

As can be seen, overall, NEMO-0 with elicit-interval=1 clearly works best in all cases.

Looking first at the value of the best individual in the population (Table XII), for ZDT-2 the differences are rather small and preference elicitation does not seem to help much. In fact, the P&K model performs even worse than NSGA-II, probably because the linear preference model is not very useful for ZDT-2's concave Pareto front. On ZDT-4, there is a clear benefit of using preference information for both preference-



(a) ZDT1-2D Linear (middle)

(b) DTLZ2-4D Chebyshev (middle)

Fig. 2. Comparing NEMO-0 two-tier ranking (MSVF) vs. NEMO-0 one-tier ranking (MSVF2). Plots show best-of-population value vs. generations.

based approaches. With elicit-interval=1, NEMO-0 is significantly better than P&K, while their performance is very similar for elicit-interval=10. For DTLZ2, NEMO-0 again shows clear advantages, and the advantage is substantially larger in the case of 4 objectives compared to the case of 3 dimensions, which supports our conjecture that preference information is particularly helpful in higher dimensional problems. Finally, for the WFG-1 problem in 5D, NSGA-II is performing quite poorly, which is not particularly surprising not only because it is a hard benchmark problem, but also because NSGA-II is known to work poorly in the case of more than 3 dimensions. So, relative to NSGA-II, NEMO-0 and the P&K preference model yield much better solutions despite the fact that they are based on NSGA-II, only additionally taking into account preference information. Again this demonstrates how the integration of preference information can overcome the challenges associated with many-objective problems. On the other hand, the amount of preference information needs to be sufficiently large. While for elicit-interval=50, the preference-based approaches still focus on the most interesting region as can be seen from the average values in Table XIII, the value of the best found solution of the preference-based solutions is not better than of NSGA-II. The difference between NEMO-0 and P&K in Table XII for each combination of test problem and elicit-interval are significant according to a Mann-Whitney-U test with 5% significance level. The only exception is ZDT4 with elicit-interval=10, where the difference is non-significant.

The results on the average value of individuals in the population confirm the above observations. All preference-based approaches outperform NSGA-II under all scenarios, partly by a huge margin, because Pareto-optimal individuals far away from the preferred region and thus with low value are dropped from the population. On the other hand, the relative difference between NEMO-0 and P&K are less clear, although all differences are still statistically significant. For ZDT4, P&K is better with respect to the average value of all individuals in the population, although it was worse or equal with respect to the best individual in the population. This may indicate that NEMO-0 is finding better solutions, but P&K is converging to a more narrow region.

VI. Conclusion

We presented a general framework for combining MOEAs with interactive preference information and ordinal regression. This framework encompasses three variants, all using pairwise comparisons to interactively elicit user preferences:

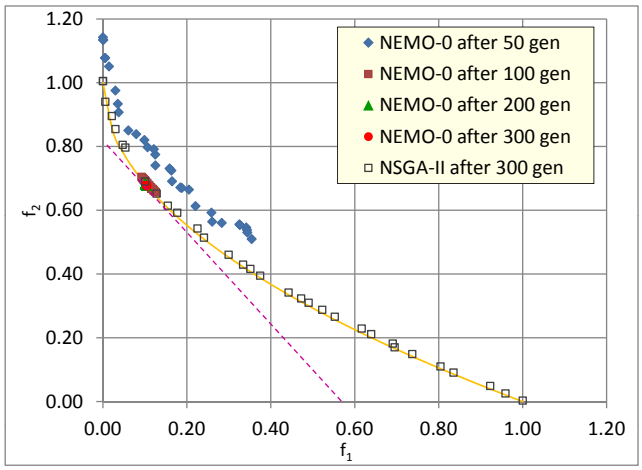
- the approach explored in this paper, NEMO-0, which combines the advantages of the well known EMO method NSGA-II with ordinal regression by learning a representative value function,
- a previously proposed method called NEMO-I, which uses robust ordinal regression to determine whether one solution is necessarily preferred to another, and
- a variant of NEMO-I, called NEMO-II, which is much faster as it requires only one LP to be solved per solution, rather than per pair of solutions.

The main advantages of NEMO-0 studied here are the following:

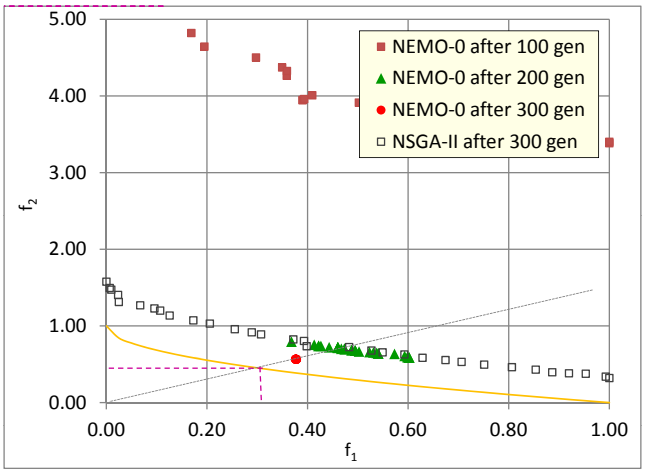
1) It models the user preferences in terms of very general additive value functions,
2) It uses a preference information expressed in a simple and intuitive way (comparisons of solutions), and requires relatively few interactions.

Our empirical results show that the proposed NEMO-0 method works as expected and is able to converge faster to the user-preferred solutions than NSGA-II without taking the user preferences into account. It also significantly outperforms a previous approach by Phelps and Köksalan [36], in particular in cases where user preferences are non-linear. Because we only have to solve one linear program per user interaction, the approach runs much faster than our previously proposed NEMO-I. There are various avenues for future work. Obviously, after having presented the general framework and NEMO-II, we should also explore this algorithm empirically. For all three approaches in the NEMO framework, we should investigate to adapt the number of interactions with the user, the times of interactions, and the pairs of solutions shown to the user.

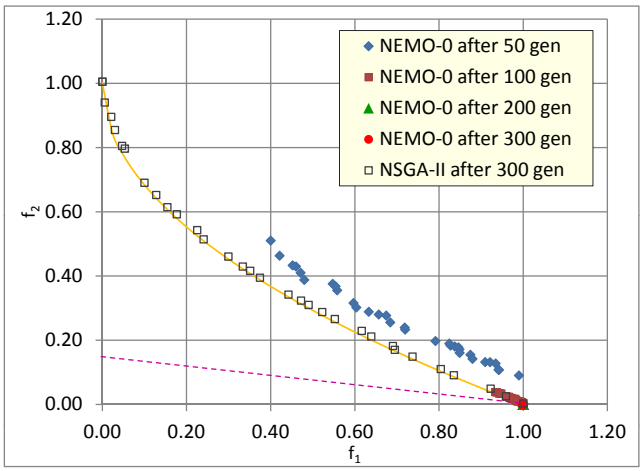
Acknowledgements: The third and fourth author wish to acknowledge financial support from the Polish National Science Center, grant no. 155534.



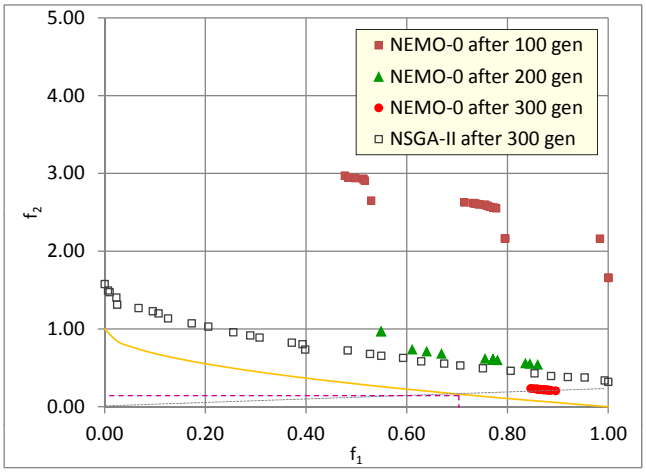
(a) ZDT1 Linear (middle)



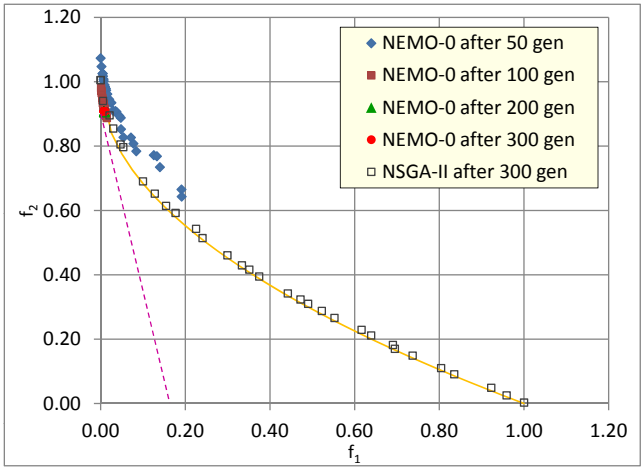
(b) ZDT4 Chebysheff (middle)



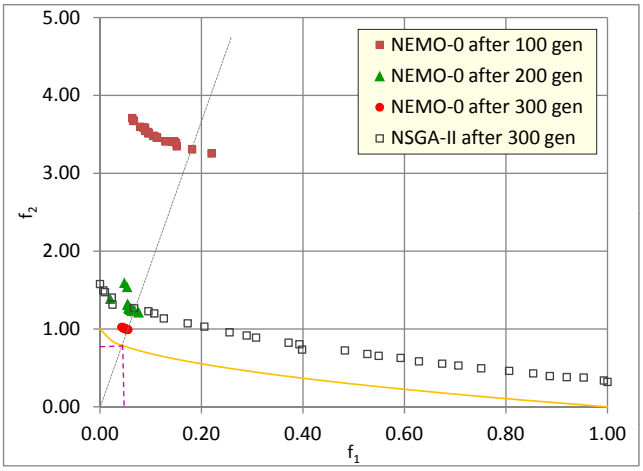
(c) ZDT1 Linear (extreme-1)



(d) ZDT4 Chebysheff (extreme-1)



(e) ZDT1 Linear (extreme-2)



(f) ZDT4 Chebysheff (extreme-2)

Fig. 3. Population of NEMO-0 after 50, 100, 200 and 300 generations in objective space.

TABLE XII
AREA UNDER CONVERGENCE CURVE (VALUE OF BEST INDIVIDUAL IN THE POPULATION).

	ZDT2-2D	ZDT4-2D	DTLZ2-3D	DTLZ2-4D	WFG1-5D
NEMO-0 (elicit-interval=1)	**0.970** $\pm$ 0.015	**0.905** $\pm$ 0.104	**0.907** $\pm$ 0.005	**0.791** $\pm$ 0.015	**0.451** $\pm$ 0.023
P&K (elicit-interval=1)	1.012 $\pm$ 0.025	0.935 $\pm$ 0.125	1.011 $\pm$ 0.037	0.997 $\pm$ 0.123	0.494 $\pm$ 0.019
NEMO-0 (elicit-interval=10)	0.995 $\pm$ 0.027	0.946 $\pm$ 0.111	0.956 $\pm$ 0.017	0.900 $\pm$ 0.032	0.672 $\pm$ 0.075
P&K (elicit-interval=10)	1.126 $\pm$ 0.073	0.942 $\pm$ 0.114	1.157 $\pm$ 0.073	1.344 $\pm$ 0.301	0.822 $\pm$ 0.066
NEMO-0 (elicit-interval=50)	——	——	——	——	1.039 $\pm$ 0.085
P&K (elicit-interval=50)	——	——	——	——	1.167 $\pm$ 0.161

TABLE XIII
AREA UNDER CONVERGENCE CURVE (AVERAGE VALUE OF ALL INDIVIDUALS IN THE POPULATION).

	ZDT2-2D	ZDT4-2D	DTLZ2-3D	DTLZ2-4D	WFG1-5D
NEMO-0 (elicit-interval=1)	**0.783** $\pm$ 0.012	0.916 $\pm$ 0.092	**0.520** $\pm$ 0.006	**0.452** $\pm$ 0.010	**0.119** $\pm$ 0.006
P&K (elicit-interval=1)	0.801 $\pm$ 0.019	**0.848** $\pm$ 0.092	0.566 $\pm$ 0.025	0.569 $\pm$ 0.094	0.138 $\pm$ 0.006
NEMO-0 (elicit-interval=10)	0.822 $\pm$ 0.020	0.957 $\pm$ 0.096	0.635 $\pm$ 0.042	0.644 $\pm$ 0.044	0.183 $\pm$ 0.023
P&K (elicit-interval=10)	0.888 $\pm$ 0.054	0.854 $\pm$ 0.087	0.656 $\pm$ 0.049	0.764 $\pm$ 0.164	0.218 $\pm$ 0.018
NEMO-0 (elicit-interval=50)	——	——	——	——	0.283 $\pm$ 0.024
P&K (elicit-interval=50)	——	——	——	——	0.297 $\pm$ 0.040

REFERENCES

[1] H. Ishibuchi, N. Tsukamoto, and Y. Nojima. Evolutionary many-objective optimization: A short review. In *Congress on Evolutionary Computation*, pages 2424–2431. IEEE, 2008.

[2] K. Miettinen. *Nonlinear Multiobjective Optimization*. Kluwer, Dordrecht, 1999.

[3] R. Steuer. *Multiple Criteria Optimization: Theory, Computation and Application*. Wiley, New York, 1986.

[4] Ph. Vincke. *Multicriteria Decision Aid*. Wiley, Chichester, 1992.

[5] J. Branke, K. Deb, K. Miettinen, and R. Słowiński, editors. *Multiobjective Optimization: Interactive and Evolutionary Approaches*, volume 5252 of *LNCS*. Springer, Berlin, 2008.

[6] J. Branke, S. Greco, R. Słowiński, and P. Zielniewicz. Interactive evolutionary multiobjective optimization using robust ordinal regression. In M. Ehrgott et al., editors, *International Conference on Evolutionary Multi-Criterion Optimization*, volume 5467 of *LNCS*, pages 554–568. Springer, 2009.

[7] J. Branke, S. Greco, R. Słowiński, and P. Zielniewicz. Interactive evolutionary multiobjective optimization driven by robust ordinal regression. *Bulletin of the Polish Academy of Sciences - Technical Sciences*, 58(3):347–358, 2010.

[8] J.R. Figueira, S. Greco, and M. Ehrgott, editors. *Multiple Criteria Decision Analysis: State of the Art Surveys*. Springer, Berlin, 2010.

[9] J. March. Bounded rationality, ambiguity and the engineering of choice. *Bell Journal of Economics*, 9:587–608, 1978.

[10] V. Srinivasan and A. D. Shocker. Estimating the weights for multiple attributes in a composite criterion using pairwise judgments. *Psychometrika*, 38:473–493, 1973.

[11] D. Pekelman and S. K. Sen. Mathematical programming models for the determination of attribute weights. *Management Science*, 20:1217–1229, 1974.

[12] E. Jacquet-Lagrèze and Y. Siskos. Assessing a set of additive utility functions for multicriteria decision making: The UTA method. *European Journal of Operational Research*, 10:151–164, 1982.

[13] Y. Siskos, V. Grigoroudis, and N. Matsatsinis. UTA methods. In F. Figueira, S. Greco, and M. Ehrgott, editors, *Multiple Criteria Decision Analysis: State-of-the-Art Surveys*, pages 297–343. Springer, Berlin, 2005.

[14] L. Kiss, J. M. Martel, and R. Nadeau. ELECCALC - an interactive software for modelling the decision maker's preferences. *Decision Support Systems*, 12:757–777, 1994.

[15] V. Mousseau and R. Słowiński. Inferring an ELECTRE TRI model from assignment examples. *Journal of Global Optimization*, 12:157–174, 1998.

[16] S. Greco, B. Matarazzo, and R. Słowiński. Rough sets theory for multicriteria decision analysis. *European Journal of Operational Research*, 129(1):1–47, 2001.

[17] J. Figueira, S. Greco, and R. Słowiński. Building a set of additive value functions representing a reference preorder and intensities of preference: Grip method. *European Journal of Operational Research*, 195(2):460–486, 2009.

[18] S. Greco, V. Mousseau, and R. Słowiński. Ordinal regression revisited: Multiple criteria ranking with a set of additive value functions. *European Journal of Operational Research*, 191(2):415–435, 2008.

[19] S. Greco, R. Słowiński, J. Figueira, and V. Mousseau. Robust ordinal regression. In S. Greco et al., editors, *Trends in Multiple Criteria Decision Analysis*, pages 273–320. Springer, 2010.

[20] R.L. Keeney and H. Raiffa. *Decisions with multiple objectives: Preferences and value tradeoffs*. J. Wiley, 1976.

[21] M. Beuthe and G. Scannella. Comparative analysis of UTA multicriteria methods. *European Journal of Operational Research*, 130(2):246–262, 1996.

[22] S. Greco, V. Mousseau, and R. Słowiński. Parsimonious preference models for robust ordinal regression. Presentation at 74 EWG-MCDA, Yverdon-les-Bains, October 2011.

[23] C. M. Fonseca and P. J. Fleming. Genetic algorithms for multiobjective optimization: Formulation, discussion, and generalization. In *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 416–423, 1993.

[24] A. Lopez Jaimes, A. Arias Montano, and C. A. Coello Coello. Preference incorporation to solve many-ojective airfoil design problems. In *Congress on Evolutionary Computation*, pages 1605–1612. IEEE, 2011.

[25] L. B. Said, S. Bechikh, and K. Ghedira. The r-dominance: A new dominance relation for interactive evolutionary multicriteria decision making. *IEEE Transactions on Evolutionary Computation*, 14(5):801–818, 2010.

[26] J. Branke, T. Kaußler, and H. Schmeck. Guidance in evolutionary multi-objective optimization. *Advances in Engineering Software*, 32:499–507, 2001.

[27] T. Wagner and H. Trautmann. Integration of preferences in hypervolume-based multiobjective evolutionary algorithms by means of desirability functions. *IEEE Transactions on Evolutionary Computation*, 14(5):688–701, 2010.

[28] G. W. Greenwood, X. S. Hu, and J. G. D'Ambrosio. Fitness functions for multiple objective optimization problems: combining preferences with Pareto rankings. In R. K. Belew and M. D. Vose, editors, *Foundations of Genetic Algorithms*, pages 437–455. Morgan Kaufmann, 1997.

[29] K. Deb and S. Chaudhuri. I-MODE: An interactive multi-objective optimization and decision-making using evolutionary methods. *Applied Soft Computing*, 10:496–511, 2010.

[30] M. Köksalan and I. Karahan. An interactive territory defining evolutionary algorithm: iTDEA. *IEEE Transactions on Evolutionary Computation*, 14(5):702–722, 2010.

[31] M. Gong, F. Liu, W. Zhang, L. Jiao, and Q. Zhang. Interactive MOEA/D for multi-objective decision making. In *Genetic and Evolutionary Computation Conference*, pages 721–728. ACM, 2011.

[32] J. Branke. Consideration of user preferences in evolutionary multi-objective optimization. In J. Branke, K. Deb, K. Miettinen, and R. Slowinski, editors, *Multiobjective Optimization - Interactive and Evo-*

lutionary Approaches, volume 5252 of *LNCS*, pages 157–178. Springer, 2008.

[33] C. A. Coello Coello, D. A. Van Veldhuizen, and G. W. Greenwood B. Lamont. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Kluwer, 2002.

[34] C. A. Coello Coello. Handling preferences in evolutionary multiobjective optimization: A survey. In *Congress on Evolutionary Computation*, volume 1, pages 30–37. IEEE, 2000.

[35] L. Rachmawati and D. Srinivasan. Preference incorporation in multiobjective evolutionary algorithms: A survey. In *Congress on Evolutionary Computation*, pages 3385–3391. IEEE, 2006.

[36] S. Phelps and M. Köksalan. An interactive evolutionary metaheuristic for multiobjective combinatorial optimization. *Management Science*, 49(12):1726–1738, 2003.

[37] H. J. C. Barbosa and A. M. S. Barreto. An interactive genetic algorithm with co-evolution of weights for multiobjective problems. In L. Spector et al., editors, *Genetic and Evolutionary Computation Conference*, pages 203–210. Morgan Kaufmann, 2001.

[38] K. Deb, A. Sinha, P.J. Korhonen, and J. Wallenius. An interactive evolutionary multiobjective optimization method based on progressively approximated value functions. *IEEE Transactions on Evolutionary Computation*, 14(5):723–739, 2010.

[39] D. S. Todd and P. Sen. Directed multiple objective search of design spaces using genetic algorithms and neural networks. In Wolfgang Banzhaf et al., editor, *Genetic and Evolutionary Computation Conference*, pages 1738–1743. Morgan Kaufmann, San Francisco, California, 1999.

[40] R. Battiti and A. Passerini. Brain-computer evolutionary multiobjective optimization: A genetic algorithm adapting to the decision maker. *IEEE Transactions on Evolutionary Computation*, 14(5):671–687, 2010.

[41] P. Campigotto and A. Passerini. Adapting to a realistic decision maker: experiments towards a reactive multi-objective optimizer. In *International Conference on Learning and Intelligent Optimization*, volume 6073 of *LNCS*, pages 338–341. Springer, 2010.

[42] A. Jaszkiewicz. Interactive multiobjective optimization with the Pareto memetic algorithm. *Foundations of Computing and Decision Sciences*, 32(1):15–32, 2007.

[43] C. White, A. Sage, and S. Dozono. A model of multiattribute decision-making and tradeoff weight determination under uncertainty. *IEEE Transactions on Systems, Man, and Cybernetics*, 14:223–229, 1984.

[44] J. Branke, K. Deb, H. Dierolf, and M. Osswald. Finding knees in multi-objective optimization. In *Parallel Problem Solving from Nature*, number 3242 in LNCS, pages 722–731. Springer, 2004.

[45] I. Das. On characterizing the 'knee' of the pareto curve based on normal-boundary intersection. *Structural Optimization*, 18(2/3):107–115, 1999.

[46] T. Wagner, H. Trautmann, and D. Brockhoff. Preference articulation by means of the R2 indicator. In *Evolutionary Multi-Criterion Optimization Conference*, volume 7811 of *LNCS*, pages 81–95. Springer, 2013.

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

[48] J. W. Fowler, E. S. Gel, M. M. Köksalan, P. Korhonen, J. L. Marquis, and J. Wallenius. Interactive evolutionary multi-objective optimization for quasi-concave preference functions. *European Journal of Operational Research*, 206:417–425, 2010.

[49] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan. A fast and elitist multi-objective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.

[50] E. Zitzler, K. Deb, and L. Thiele. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation Journal*, 8(2):125–148, 2000.

[51] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler. Scalable multi-objective optimization test problems. In *Proceedings of the Congress on Evolutionary Computation (CEC-2002)*, pages 825–830, 2002.

[52] S. Huband, P. Hingston, L. Barone, and L. While. A review of multi-objective test problems and a scalable test problem toolkit. *IEEE Transactions on Evolutionary Computation*, 10(5):477–506, 2006.