A New Paradigm in Interactive Evolutionary Multiobjective Optimization

Bhupinder Singh Saini[1][0000−0003−2455−3008], Jussi Hakanen[1][0000−0001−9579−8657], and Kaisa Miettinen[1][0000−0003−1013−4689]

University of Jyvaskyla, Faculty of Information Technology, P.O. Box 35 (Agora), FI-40014 University of Jyvaskyla, Finland
{bhupinder.s.saini,jussi.hakanen,kaisa.miettinen}@jyu.fi

Abstract. Over the years, scalarization functions have been used to solve multiobjective optimization problems by converting them to one or more single objective optimization problem(s). This study proposes a novel idea of solving multiobjective optimization problems in an interactive manner by using multiple scalarization functions to map vectors in the objective space to a new, so-called preference incorporated space (PIS). In this way, the original problem is converted into a new multiobjective optimization problem with typically fewer objectives in the PIS. This mapping enables a modular incorporation of decision maker's preferences to convert any evolutionary algorithm to an interactive one, where preference information is directing the solution process. Advantages of optimizing in this new space are discussed and the idea is demonstrated with two interactive evolutionary algorithms: IOPIS/RVEA and IOPIS/NSGA-III. According to the experiments conducted, the new algorithms provide solutions that are better in quality as compared to those of state-of-the-art evolutionary algorithms and their variants where preference information is incorporated in the original objective space. Furthermore, the promising results require fewer function evaluations.

Keywords: Interactive methods · Achievement scalarizing functions · Evolutionary algorithms · Preference information · Decision maker

1 Introduction

Many multiobjective optimization problems (MOPs) are encountered in real life applications. Due to the conflicting nature of objectives in these problems, often there does not exist a single optimal solution. Instead, there exists a set of *Pareto optimal solutions* which represent trade-offs among the various objectives.

One family of methods, known as *a posteriori* methods, solve MOPs by finding a set of solutions which adequately represents the entire set of Pareto optimal solutions [18]. Evolutionary algorithms (EAs) have been employed for this with a varying degree of success. An example of such methods is NSGA-II [6], which works well for solving problems with a low number of objectives, but the performance degrades as the number of objectives increases [9]. Recent a posteriori EAs have tackled this problem in various ways [14].

However, increasing the number of objectives brings forth new challenges. As the number of objectives increases, the number of solutions required to adequately represent the set of Pareto optimal solutions (which may have an infinite number of solutions) increases exponentially [9, 14]. Regardless of the number of objectives, only one or few of these solutions are useful to a *decision maker* (DM) who wants to find and implement the desirable solution. Hence, when using a posteriori methods, computational resources are wasted on finding solutions that are not relevant. If objective function evaluations require time-consuming simulations or physical experiments, this problem is compounded and may lead to a waste of monetary resources as well. Moreover, these algorithms leave the task of choosing the final solution to the DM. As each solution is a vector in a high-dimensional objective space, comparing potentially thousands of solutions is a difficult task. This process can set a high cognitive load on the DM.

As DMs are experts in their domain, they usually have opinions or *preferences* regarding which solutions are desirable or undesirable to them. The preference information may be elucidated in the form of desirable objective function values, ranking of importances of objectives, pair-wise comparison of solutions and many other techniques [16]. Recent advances in EAs try to incorporate this information to limit the scope of search of the EA. As the DM may learn new information about the problem during the solution process, allowing them to change their preferences during the solution process is desirable [11]. Methods which allow such change are known as *interactive* methods [17–19, 30]. Ideally, this leads to less waste of resources as only solutions that are preferable to the DM are focused upon. Moreover, as only a small subset of the Pareto optimal solutions is to be represented at a time, the number of solutions to be shown to the DM is smaller, hence reducing the cognitive load. However, many interactive EAs have problems ranging from addition of hyperparameters to lack of diversity in the population, which can impair the optimization process [1, 10].

The concept of utilizing the preferences of a DM in the solution process of an MOP is very popular in the field of multiple criteria decision making [18,19]. One of the methods adopted is to use scalarization functions [18,20]. These functions utilize the preferences of the DM to map the objective function values of solutions to scalar values, hence converting the MOP to one or more single objective optimization problems. Different scalarization functions interpret the same preference information differently, and may lead to different results [20]. Different solutions can hence be obtained by solving multiple scalarization functions with the same preference information, as done in synchronous NIMBUS [21], or by slightly modifying the preference information multiple times and optimizing the same scalarization function, as done in the reference point algorithm [28].

In this paper, we explore the concept of using multiple scalarization functions to create a new space: *Preference Incorporated Space* (PIS). First, we study the mathematical properties of this new space. More specifically, we study the effect of optimizing in the PIS, introducing a new paradigm in preference based optimization: Interactive Optimization using Preference Incorporated Space (IOPIS) algorithm. The IOPIS algorithm enables us to make use of preference informa-

tion with any a posteriori EA in an interactive way, as the preference information is encoded directly in the optimization problem in the PIS. It also enables us to control the dimension of the space in which dominance is judged, equal to the number of chosen scalarization functions. We then introduce the IOPIS algorithm, a modular algorithm that takes a given number of specified scalarization functions, and uses a DM's preferences to convert a generic MOP to an MOP in the preference incorporated space. This can then be solved interactively with any appropriate non-interactive EA together with DM's preferences. As examples, we implement two versions of the new algorithm: IOPIS/RVEA and IOPIS/NSGA-III, where the new problem in the PIS is optimized using decomposition based EAs RVEA [4] and NSGA-III [5], respectively.

The rest of the paper is organized as follows. Section 2 discusses the background of multiobjective optimization, EAs, and scalarization functions. Section 3 discusses the mathematical properties of the PIS and introduces the IOPIS algorithm with a visual explanation. In Section 4, we conduct an experimental study to compare the performances of the two implementations of the IOPIS algorithm with state of the art a posteriori EAs and their interactive variants and discuss the results. Finally, we draw conclusions in Section 5. All implementations and experimental data presented in this paper are open source and publicly available at `https://desdeo.it.jyu.fi` as a part of the DESDEO framework.

2 Background

2.1 Multiobjective Optimization

An MOP can be defined as:

$$\begin{aligned} &\text{minimize } \{f_1(\mathbf{x}), \ldots, f_k(\mathbf{x})\} \\ &\text{subject to } \mathbf{x} \in S, \end{aligned} \tag{1}$$

where $\mathbf{x} = (x_1, \ldots, x_n)^T$ are vectors of decision variables belonging to the feasible set $S \subset \mathbb{R}^n$. The $k\,(\geq 2)$ objective functions f_i map vectors of S to $\mathbb{R}$. The objective function values $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_k(\mathbf{x}))$ form objective vectors in the objective space $\mathbb{R}^k$. A solution $\mathbf{x}^1 \in S$ of problem (1) is said to *dominate* another solution $\mathbf{x}^2 \in S$ (written as $\mathbf{f}(\mathbf{x}^1) \succ \mathbf{f}(\mathbf{x}^2)$) if $f_i(\mathbf{x}^1) \leq f_i(\mathbf{x}^2)$ for all $i = 1, \ldots, k$ and $f_j(\mathbf{x}^1) < f_j(\mathbf{x}^2)$ for at least one $j = 1, \ldots, k$. Pareto optimal solutions are solutions of the MOP which are not dominated by any other solution in S. For this reason, they are also referred to as *non-dominated solutions*. Sometimes, it is desirable for DMs to consider a subset of Pareto optimal solutions with bounded trade-offs [18]. Such solutions are called *properly Pareto optimal* solutions.

We can define the set of solutions of problem (1), known as a Pareto set, as:

$$PS_{OS} = \{\mathbf{x} \in S \mid \nexists_{\mathbf{x}^* \in S}\ \mathbf{f}(\mathbf{x}^*) \succ \mathbf{f}(\mathbf{x})\}, \tag{2}$$

where the subscript OS refers to the fact that the set was obtained by considering the objective vectors in the objective space. We can now define an *ideal point* and a *nadir point* of problem (1). These points represent the lower and upper

bounds of the ranges of the objective function values among the Pareto optimal solutions, respectively. The ideal point $\mathbf{z}^* = (z_1^*, \ldots, z_k^*)$ can be calculated as $z_i^* = \min_{x \in S} f_i(\mathbf{x})$. The nadir point $\mathbf{z}^{nad} = (z_1^{nad}, \ldots, z_k^{nad})$ can be calculated as $z_i^{nad} = \max_{x \in PS_{OS}} f_i(\mathbf{x})$. It should be noted that calculating the nadir point requires the calculation of the PS_{OS}. Hence, the calculation of the nadir point is tricky in problems with more than two objectives and needs to be estimated [8,18]. Any objective vector $\mathbf{z}$ is defined to be *achievable* if $\mathbf{z}$ belongs to the set:

$$T = \{\mathbf{z} \in \mathbb{R}^k \mid \exists_{\mathbf{x} \in S}\ \mathbf{f}(\mathbf{x}) \succ \mathbf{z} \text{ or } \mathbf{f}(\mathbf{x}) = \mathbf{z}\}. \tag{3}$$

By definition, the nadir point is an achievable point, while the ideal point is not.

2.2 Evolutionary Algorithms

Decomposition-based methods such as NSGA-III [5], RVEA [4], and many variants of MOEA/D [31] have become popular in the evolutionary multiobjective optimization community. These methods decompose the objective space into sections using directional vectors called reference vectors, reference points, or weights. For simplicity, in what follows, we will be using the term reference vectors (RVs). These RVs, usually spread uniformly in the objective space, represent individual single-objective optimization problems. The RVs are typically generated using a simplex lattice design, and the number of RVs is equal to $\binom{l+k-1}{k-1}$, where l is a parameter controlling the density of the RVs. Subsets of the population which lie in the decomposed region associated with an RV (in the objective space) evolve in the direction of that RV based on scalar fitness values calculated using the RV and their objective function values.

As mentioned in the introduction, EAs which approximate the entire Pareto front exhibit many downsides. Methods have been proposed to get around those downsides by incorporating the preferences of the DM in an interactive fashion, see, e.g. [23, 26, 30]. One of the ways to incorporate a DM's preferences in decomposition-based EAs is to manipulate the spread of the RVs to account for the preferences [4,15]. In many such methods, the DM is required to provide their preferences in the form of a *reference point* in the objective space [12, 26, 27]. The components of a reference point are desirable values of each objective function, which may or may not be achievable. Then, uniformly spread RVs are translated towards this point. This translation introduces a new scalar hyperparameter which controls the final spread of the RVs around the reference point. This method introduces a few new problems, though. Firstly, the effect of changing the value of the newly introduced hyperparameter may be difficult for a DM to understand. But an appropriate value for this hyperparameter is important as it has been observed that a small spread of RVs may lead to a degradation in population diversity, which prohibits the convergence of the EA [1, 10].

2.3 Achievement Scalarizing Functions

As mentioned, scalarization functions are functions that map a vector to a real-valued scalar. The weighted sum function and the Chebyshev function used by

MOEA/D and the angle-penalized distance function used by RVEA are examples of scalarization functions [4,31]. To be regarded as a good scalarization function, it must have some desirable properties [25]. Firstly, the solutions obtained by optimizing the scalarization function should be Pareto optimal. Secondly, these solutions should be satisfactory according to the preferences of a DM, if the preferences are feasible. Finally, any Pareto optimal solution should be discoverable by changing the preferences provided by the DM.

Unfortunately, no single scalarization function satisfies the three conditions concurrently [25]. However, if we relax the conditions to only account for properly Pareto optimal solutions, rather than all Pareto optimal solutions, then all three conditions can be satisfied by some scalarization functions. In this paper, we focus on a subclass of scalarization functions, known as achievement scalarizing functions (shortened to achievement function) [29]. An achievement function is a continuous function $s : \mathbb{R}^k \to \mathbb{R}$. Achievement functions are characterized by either being strictly increasing and order-representing, or strongly increasing and order-approximating [29]. We will focus on the latter kind as they satisfy all three relaxed desirable properties.

Theorem 1. *[29] Let us consider $\mathbf{z}^1, \mathbf{z}^2 \in \mathbb{R}^k$ such that $\mathbf{z}^1 \succ \mathbf{z}^2$. Then for any order-approximating achievement function $s : \mathbb{R}^k \to \mathbb{R}$, we have*

$$s(\mathbf{z}^1) < s(\mathbf{z}^2). \tag{4}$$

From Theorem 1, it can be concluded that solving the following problem:

$$\begin{array}{c} \text{minimize } s(\mathbf{f}(\mathbf{x})) \\ \text{subject to } \mathbf{x} \in S \end{array} \tag{5}$$

will lead to a Pareto optimal solution of problem (1) [28, 29].

A general formulation of an (order-approximating) achievement function is:

$$s(\mathbf{f}(\mathbf{x}), \bar{\mathbf{z}}) = \max_{i=1,\ldots,k} \left[\frac{f_i(\mathbf{x}) - \bar{z}_i}{\mu_i} \right] + \rho \sum_{i=1}^{k} \left(\frac{f_i(\mathbf{x}) - \bar{z}_i}{\mu_i} \right), \tag{6}$$

where ρ is a small positive scalar and μ_i are positive scalars and $\bar{\mathbf{z}} \in \mathbb{R}^k$ is a reference point provided by the DM [24]. Minimizing $s(\mathbf{f}(\mathbf{x}), \bar{\mathbf{z}})$ has the effect of optimizing problem (1) by sliding a cone along the line $\bar{\mathbf{z}} + \lambda\boldsymbol{\mu}$, where $\lambda \in \mathbb{R}$, so that a minimum (> 0) number of solutions lie in the cone [20]. Bounds of the trade-offs in the solutions obtained by (5) can be controlled by changing ρ [28].

The general formulation (6) represents achievement functions that can take preferences in other forms, not just reference points [24]. Different achievement functions differ in how $\boldsymbol{\mu}$ is set, which means they are optimizing along different directions, albeit starting from the same reference point $\bar{\mathbf{z}}$. Hence, they may lead to different solutions even if the same reference point is provided to them. For the implementation of the IOPIS algorithm, we focus on the GUESS [2] and STOM [22] scalarization functions (based on e.g., [3, 20]). For the GUESS function, $\mu_i = z_i^{nad} - \bar{z}_i$ and for the STOM function, $\mu_i = \bar{z}_i - z_i^*$. As $\mu_i > 0$ for all

$i = 1, \ldots, k$, it follows that for these two achievement functions, $z_i^* < \bar{z}_i < z_i^{nad}$ for all $i = 1, \ldots, k$. Another achievement function of note is the achievement scalarizing function (ASF) used in the reference point method [28], which is used in the experimental study section. For ASF, $\mu_i = z_i^{nad} - z_i^*$.

3 Optimization in Preference Incorporated Space

3.1 Properties of Preference Incorporated Space

Let there be a set of achievement functions $\mathbf{s} = \{s_1, \ldots, s_q\}$ with $q \geq 2$. Then we can define a PIS as the set $\{\mathbf{s}(\mathbf{f}(\mathbf{x}), \bar{\mathbf{z}}) \in \mathbb{R}^q\}$, and a new MOP in the PIS as:

$$\begin{aligned} &\text{minimize } \mathbf{s}(\mathbf{f}(\mathbf{x}), \bar{\mathbf{z}}) = \{s_1(\mathbf{f}(\mathbf{x}), \bar{\mathbf{z}}), \ldots, s_q(\mathbf{f}(\mathbf{x}), \bar{\mathbf{z}})\} \\ &\text{subject to } \mathbf{x} \in S. \end{aligned} \tag{7}$$

Two solutions $\mathbf{x}^1$ and $\mathbf{x}^2$ can now be compared in two spaces. As stated in Section 2.1, a solution $\mathbf{x}^1$ is said to dominate another solution $\mathbf{x}^2$ *in the objective space* if $\mathbf{f}(\mathbf{x}^1) \succ \mathbf{f}(\mathbf{x}^2)$. A solution $\mathbf{x}^1$ is said to dominate $\mathbf{x}^2$ *in the PIS* if $\mathbf{s}(\mathbf{f}(\mathbf{x}^1), \bar{\mathbf{z}}) \succ \mathbf{s}(\mathbf{f}(\mathbf{x}^2), \bar{\mathbf{z}})$. Similar to (2), we can define the solutions to problem (7), i.e., the Pareto set obtained by optimizing in the PIS as:

$$PS_{PIS} = \{x \in S \mid \nexists_{x^* \in S} \ \mathbf{s}(\mathbf{f}(\mathbf{x}^*), \bar{\mathbf{z}}) \succ \mathbf{s}(\mathbf{f}(\mathbf{x}), \bar{\mathbf{z}})\}. \tag{8}$$

We modify the desirable properties of scalarization functions as stated in [25] to reflect properties related to the PIS as:

1. Pareto optimal solutions in the PIS remain Pareto optimal in the objective space.
2. Pareto optimal solutions in the PIS follow the preference given by the DM in the objective space.
3. Any properly Pareto optimal solution of problem (1) can be discovered by changing the reference point of problem (7).

It can be shown that the first condition is true regardless of the choice or number of the achievement functions.

Theorem 2. *Let PS_{PIS} be the set of Pareto optimal solutions of problem* (7)*. Let PS_{OS} be the set of Pareto optimal solutions of problem* (1)*. Then,*

$$PS_{PIS} \subset PS_{OS}. \tag{9}$$

Proof. Suppose $\mathbf{x} \in PS_{PIS}$ but $\mathbf{x} \notin PS_{OS}$. Therefore, there exists some $\mathbf{x}^*$ such that $\mathbf{f}(\mathbf{x}^*) \succ \mathbf{f}(\mathbf{x})$. Thus, according to Theorem 1, $s^i_{\bar{\mathbf{z}}}(\mathbf{f}(\mathbf{x}^*)) < s^i_{\bar{\mathbf{z}}}(\mathbf{f}(\mathbf{x}))$ for all $i \in \{1, \ldots, q\}$. Hence, $\mathbf{s}_{\bar{\mathbf{z}}}(\mathbf{f}(\mathbf{x}^*)) \succ \mathbf{s}_{\bar{\mathbf{z}}}(\mathbf{f}(\mathbf{x}))$, which contradicts $\mathbf{x} \in PS_{PIS}$. □

The set PS_{PIS} represents the trade-offs between the values of the various achievement functions in problem (7). As the different achievement functions are different interpretations of the same preference information obtained from a

DM, the solutions in the set PS_{PIS} represent the trade-offs between those interpretations. Hence, it can be said that solutions obtained by solving problem (7) follow the preferences given by the DM. Moreover, as PS_{PIS} includes solutions which minimize individual achievement functions present in PIS, and as any properly Pareto optimal solution in the objective space can be found using the achievement functions by changing the reference points, it follows that the third condition also holds. Note that these results are valid for all order-approximating scalarization functions, and not just STOM and GUESS functions.

Solving the MOP in the PIS has a few benefits compared to solving the MOP in the objective space. Firstly, we can control the dimension of the PIS, which is equal to the number of achievement functions chosen. This means that, given some number (≥ 2) of achievement functions, we can use any multiobjective EA (or biobjective EA, as PIS can be a two dimensional space) to solve problem (7), regardless of the number of objectives in the original problem. Secondly, controlling the dimension of the optimization problem also gives us an indirect control over the number of function evaluations needed by the EA during the optimization process. This is because the number of solutions required to adequately represent the set of Pareto optimal solutions increases with increasing the dimension of the objective space (for problem (1)) or PIS (for problem (7)). Hence, choosing fewer achievement functions than k is an easy way to reduce the number of function evaluations needed by an EA to solve an optimization problem. Thirdly, by incorporating the preference information in the PIS, we gain the ability to use any non-interactive EA in an interactive fashion. This modularity enables easy use of well-tested EAs without needing to change them to enable interaction with a DM.

3.2 The IOPIS algorithm

The IOPIS algorithm takes a formulation of the optimization problem of the form (1) as input. The algorithm also takes as its input a set of achievement functions $\mathbf{s}$. The ideal point $\mathbf{z}^*$ and the nadir point $\mathbf{z}^{nad}$ of the problem are also taken as inputs. As the calculation of the nadir point can be tricky in problems with more than two objectives, approximate values of the nadir point (and ideal point) can also be used. The solutions are generated between these two points. Hence, the DM can use their expertise to give the approximate values of the points within which to search. The interactive solution process begins when these points are shown to the DM. The following four steps are repeated iteratively until the DM has received a satisfactory solution:

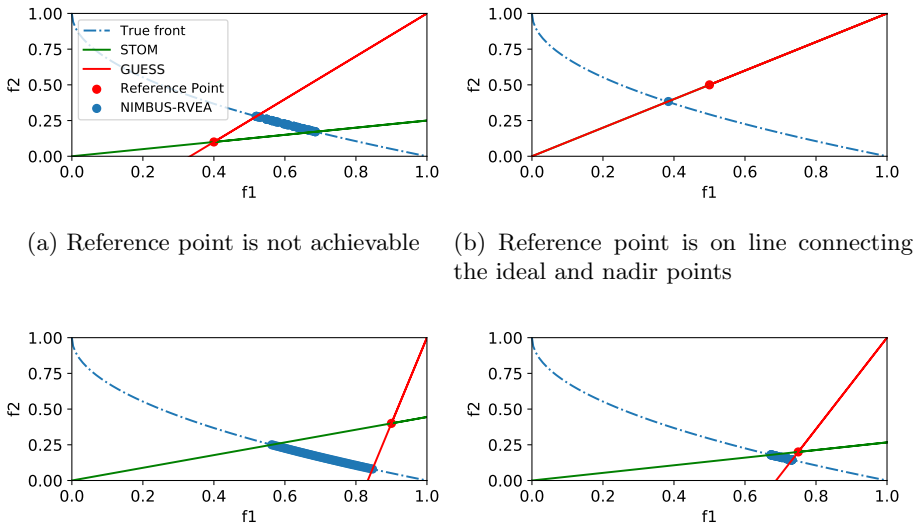
1. *Preference elicitation*: The DM is asked to give their preferences as a reference point based on the information currently available to them.
2. *Problem creation*: Using the original objectives, the known estimates of the ideal and nadir points, the reference point, and the set of achievement functions, a new optimization problem is created in the PIS, as shown in (7).
3. *Problem solution*: Solve the problem created in the previous step with an EA. If this is the first iteration of the algorithm, start the EA with a new

population, generated in a manner specific to the selected EA. In subsequent iterations, the population from the previous iteration is used as the starting population.

4. *Display solutions*: Display the solutions obtained in step 3 to the DM. The DM can indicate the maximum number of solutions to be shown at a time in step 4. If the number of solutions generated in step 3 exceeds the limit, e.g., clustering can be applied before displaying solutions.

3.3 Visual Interpretation

Even though the IOPIS algorithm is designed for optimization problems with more than two objectives, a biobjective problem is easily visualizable to demonstrate the algorithm. Here we use the ZDT1 problem [32] to study the effect of the choice of the reference point on the solutions returned by one implementation of the IOPIS algorithm. In this implementation, STOM and GUESS scalarization functions are used with NSGA-III to solve the resulting MOP in the PIS.



(a) Reference point is not achievable

(b) Reference point is on line connecting the ideal and nadir points

(c) Reference point is achievable

(d) Reference point is close to the front

Fig. 1. Solutions obtained for various reference points for the ZDT1 problem.

Four different reference points are given to the algorithm. Each subfigure in Figure 1 shows the corresponding objective vectors returned by the IOPIS algorithm. In each subfigure, the blue dashed curve represents the true Pareto front of the problem and the red point is the reference point. The green line represents the direction along which the STOM function optimizes, whereas the red line represents the direction along which the GUESS function optimizes.

1. The reference point is not achievable (Figure 1a): There is no solution that achieves values close to the reference point. Minimizing each achievement function individually returns the solution corresponding to the point of intersection of the line representing that achievement function and the Pareto front. As can be seen, the solutions returned by the algorithm include those solutions, and nondominated solutions in between.
2. Reference point is on the line joining the ideal and nadir point (Figure 1b): Due to the nature of the chosen achievement functions, only a single solution is returned by the algorithm. This is because if the reference point is on the line connecting the ideal and nadir, both achievement functions optimize along the same line. This behaviour can be changed by choosing a different set of achievement functions to form the PIS, or by shifting the reference point slightly to increase the diversity of the solutions.
3. The reference point is achievable and dominated (Figure 1c): The algorithm returns a set of solutions that satisfy the given reference point. As in the first case, optimal solutions of the individual achievement functions are included.
4. The reference point is close to the front (Figure 1d): Bringing the reference point closer to the front has the effect of reducing the spread of the solutions returned by the algorithm, hence solutions are returned in a narrower region.

The spread of the solutions is controlled by the position of the reference point. A DM who does not know very well what is realistic may provide a reference point far from the front. In such cases, the algorithm will return a diverse set of solutions (with an exception and possible resolutions discussed in point 2. above). After being provided with such solutions, the DM will have more knowledge about the trade-offs involved among the solutions, and may want to fine-tune their search in a narrow region. This is easily accomplished by providing a reference point closer to the now known region of the front. This methodology of control is similar to the one proposed in the reference point method [28].

4 Numerical Results

4.1 Experimental Setup

In this study, two versions of the interactive IOPIS algorithm were implemented. IOPIS/NSGA-III uses NSGA-III to solve the problem in the PIS, while IOPIS/RVEA uses RVEA. In both implementations, the STOM and GUESS functions are used as achievement functions to form the PIS. These algorithms were compared against a posteriori RVEA [4] and NSGA-III [5]. Interactive versions of the two a posteriori algorithms (iRVEA and iNSGA-III) were also implemented and included in this study. The details of iRVEA can be found in [12] and iNSGA-III was implemented in a similar manner. RVEA and NSGA-III were chosen for this study as they have been shown to work well in problems with $k > 2$ [4, 5]. Even though the problem in the PIS here is biobjective, the implementations of the IOPIS algorithm use RVEA and NSGA-III to ensure that only the effect of optimizing in the PIS is reflected in the results, and not the choice of the EA.

The algorithms were compared using the DTLZ{2-4} [7] and WFG{1-9} [13] problems, with 3-9 objectives each. The number of variables was kept as $10+k-1$, as recommended in [7]. For the IOPIS EAs, each component of the nadir point was randomly generated from a halfnormal distribution with the underlying normal distribution centered around 1 and having a scale of 0.15, then being scaled up by a factor equal to the true nadir point components (1 for the DTLZ problems, varying values for the WFG problems). This led to the generation of nadir points with components up to 50% worse than the true nadir point. This was done to test the performance of the IOPIS EAs in cases where only approximate values of the nadir point are available. The true ideal point was provided to the IOPIS EAs, as the calculation of it is relatively simpler.

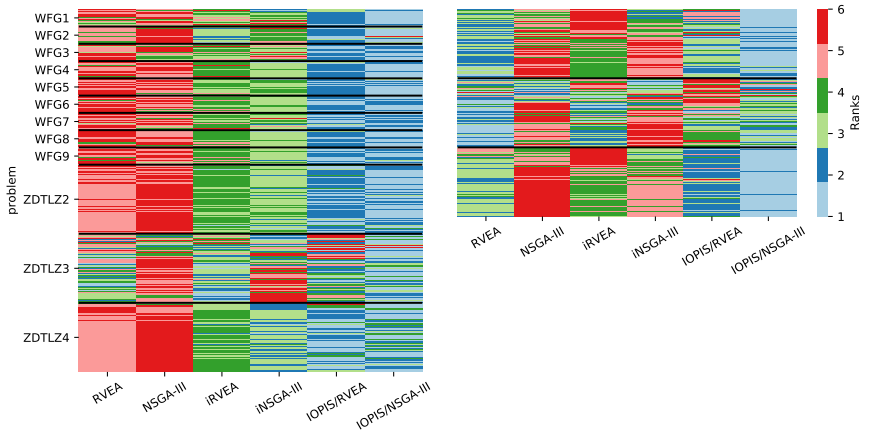
Each EA was run for four iterations. For each EA, an iteration consisted of a constant number of generations: One of {100, 150, 200, 250} for the DTLZ problems, 100 for the WFG problems (The reason for using only 100 generations per iteration for WFG problems will be discussed in the next subsection.). All other hyperparameters, such as the number of solutions or algorithm specific hyperparameters, were set to values recommended in their respective papers. In each iteration, all interactive EAs received a common reference point randomly generated in a hyperbox with the ideal and nadir points as opposing vertices. The non-interactive EAs were ran through the iterations uninterrupted. Hence 336 tests with the DTLZ problems[1] and 252 tests with the WFG problems[2] were conducted for each of the six EAs. The algorithms were compared based on the optimality and the preferability of the solutions returned at the end of each iteration, and the number of function evaluations conducted.

4.2 Experimental Results

The Pareto optimal solutions of the DTLZ2-4 problems form a hypersphere in the objective space, centered around the ideal point, which is the origin, and a radius of one. Hence, calculating the Euclidean norm of the objective vectors of the solutions returned by the EAs is a measure of optimality, with lower values of the norm being closer to the Pareto front. However, these values cannot be compared between problems, nor can they be compared for the same problem but with a different number of objectives. Instead, the median of the norm of the solutions returned at the end of each test was calculated for each of the six EAs. These values were then used to rank each EA from 1 to 6 for every test, lower ranks being given to better (lower) median norm values. A similar procedure was followed for comparing methods based on the preferability of the solutions returned by the EAs. The achievement function used in the reference point method (ASF) [28] was chosen as the metric of preferability. The median ASF values of the solutions were then used to rank the different EAs. The true nadir point was used in the calculation of the ASF values. For the tests involving the WFG 1-9 problems, ranks were only calculated based on median ASF values.

[1] = 3 (problems) * 7 (objectives) * 4 (generations per iteration) * 4 (iterations)
[2] = 9 (problems) * 7 (objectives) * 4 (iterations)

The Pareto fronts for these problems are not spherical, hence ranks based on median norm values are not relevant.



(a) Ranks based on median ASF values (b) Ranks based on median norm values

Fig. 2. Heatmaps of ranks of algorithms based on the median ASF value or median norm value of the solutions obtained.

Heatmaps of the ranks based on ASF and norm are shown in Figures 2a and 2b, respectively. A paired colormap was used in the creation of the heatmaps which gave ranks 1 and 2 a blue hue, ranks 3 and 4 a green hue, and ranks 5 and 6 a red hue. This choice brings forward a clear clustering in the rankings of the 6 EAs. As seen in Figure 2a, iRVEA and iNSGA-III tend to return more preferable solutions than their non-interactive counterparts. This behaviour is expected as RVEA and NSGA-III focus on the entire Pareto front, whereas iRVEA and iNSGA-III focus on a limited region. However, as seen in Figure 2b, the solutions returned by iRVEA were farther away from the Pareto front compared to RVEA. This is because as iRVEA has a much lower diversity of solutions compared to RVEA, which hampers the optimization process.

In both heatmaps, the PIS based EAs get ranks 1 or 2 in most tests, i.e., these algorithms returned solutions that were more preferable, and closer to the Pareto front than the other four algorithms. It should also be noted that IOPIS/NSGA-III performed better than IOPIS/RVEA in most cases. Further investigation of the PIS is required on this. The results obtained on the DTLZ3 problem are also interesting. RVEA returned solutions which were closer to the Pareto front, compared to the other methods. While the IOPIS EAs still outperformed RVEA based on the preferability of the solutions, RVEA outperformed an interactive method iNSGA-III. This is happening as iNSGA-III failed to converge to the Pareto front because of the lack of diversity of the solutions, and

got stuck on one of the local fronts of the problem. While there was a correlation between the problem type and the performance of the methods (IOPIS EAs got ranks one or two more often in the WFG problems compared to the DTLZ problems), there was no correlation between the performance of the method and the number of objectives. In the case of the DTLZ problems, the performance was also not dependent on the number of generations per iteration, i.e., there was no improvement in the results after a hundred generations (per iteration). This is why the number of generations per iterations was fixed to 100 for the tests involving the WFG problems.

The final metric of comparison is the number of function evaluations conducted. Given a constant number of generations, the number of function evaluations is linearly correlated with the population size, which is equal to the number of RVs in the EAs considered in this paper. For RVEA, NSGA-III, iRVEA and iNSGA-III, the RVs (and hence the number of function evaluations) increase exponentially with an increasing number of objectives. As the IOPIS algorithms operate in the low-dimensional PIS, the number of reference vectors, and hence the number of function evaluations, is independent of the number of objectives, and significantly lower than that for the other algorithms considered in the study. It should also be noted that for all of the tests, only an approximate nadir point was provided to the IOPIS EAs, and yet the IOPIS EAs obtain better results than the current state of the art algorithms.

5 Conclusions

A new space PIS, where preferences are incorporated, was proposed as a new paradigm of solving MOPs interactively. This new space makes the creation of interactive EAs very modular, as the algorithm only needs to modify the problem to enable interactivity, rather than the EA itself. As examples, this enabled easy creation of the IOPIS/NSGA-III and IOPIS/RVEA implementations.

The results obtained in the numerical experiments were very promising. The new interactive EAs outperformed standalone NSGA-III and RVEA, as well as their interactive versions. The solutions obtained by the IOPIS EAs were closer to the Pareto optimal front, more preferable based on the reference point and spent less computational resources in the form of function evaluations. Further study of the landscape of the PIS is needed. The effect of choosing different achievement functions, their implications on the interaction mechanism by a DM and the solutions returned by the algorithm also needs to be studied.

Acknowledgements. This research was supported by the Academy of Finland (grant numbers 322221 and 311877). The research is related to the thematic research area DEMO (Decision Analytics utilizing Causal Models and Multiobjective Optimization, jyu.fi/demo) of the University of Jyväskylä.

References

1. Bechikh, S., Kessentini, M., Said, L.B., Ghédira, K.: Chapter four - Preference incorporation in evolutionary multiobjective optimization: A survey of the state-of-the-art. In: Hurson, A.R. (ed.) Advances in Computers, vol. 98, pp. 141–207. Elsevier (2015)
2. Buchanan, J.T.: A naïve approach for solving MCDM problems: The GUESS method. Journal of the Operational Research Society **48**(2), 202–206 (1997)
3. Buchanan, J., Gardiner, L.: A comparison of two reference point methods in multiple objective mathematical programming. European Journal of Operational Research **149**(1), 17—34 (2003)
4. Cheng, R., Jin, Y., Olhofer, M., Sendhoff, B.: A reference vector guided evolutionary algorithm for many-objective optimization. IEEE Transactions on Evolutionary Computation **20**(5), 773–791 (2016)
5. Deb, K., Jain, H.: An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: Solving problems with box constraints. IEEE Transactions on Evolutionary Computation **18**(4), 577–601 (2014)
6. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation **6**(2), 182–197 (2002)
7. Deb, K., Thiele, L., Laumanns, M., Zitzler, E.: Scalable multi-objective optimization test problems. In: Proceedings of the 2002 IEEE Congress on Evolutionary Computation (CEC 2002). pp. 825–830. IEEE (2002)
8. Deb, K., Miettinen, K.: Nadir point estimation using evolutionary approaches: Better accuracy and computational speed through focused search. In: Ehrgott, M., Naujoks, B., Stewart, T.J., Wallenius, J. (eds.) Multiple Criteria Decision Making for Sustainable Energy and Transportation Systems, Proceedings. pp. 339–354. Springer, Berlin, Heidelberg (2010)
9. Deb, K., Saxena, D.: Searching for Pareto-optimal solutions through dimensionality reduction for certain large-dimensional multi-objective optimization problems. In: Proceedings of the World Congress on Computational Intelligence (WCCI-2006). pp. 3352–3360 (2006)
10. Deb, K., Sundar, J.: Reference point based multi-objective optimization using evolutionary algorithms. In: GECCO '06: Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation. pp. 635—-642. ACM, New York (2006)
11. Eskelinen, P., Miettinen, K., Klamroth, K., Hakanen, J.: Pareto Navigator for interactive nonlinear multiobjective optimization. OR Spectrum **32**(1), 211–227 (2010)
12. Hakanen, J., Chugh, T., Sindhya, K., Jin, Y., Miettinen, K.: Connections of reference vectors and different types of preference information in interactive multiobjective evolutionary algorithms. In: Proceeding of the 2016 IEEE Symposium Series on Computational Intelligence (SSCI). pp. 1–8 (2016)
13. Huband, S., Barone, L., While, L., Hingston, P.: A scalable multi-objective test problem toolkit. In: Coello Coello, C.A., Hernández Aguirre, A., Zitzler, E. (eds.) Evolutionary Multi-Criterion Optimization, Third International Conference, Proceedings. pp. 280–295. Springer, Berlin. Heidelberg (2005)
14. Ishibuchi, H., Tsukamoto, N., Nojima, Y.: Evolutionary many-objective optimization: A short review. In: Proceedings of the 2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence). pp. 2419–2426 (2008)

15. Li, K., Chen, R., Min, G., Yao, X.: Integration of preferences in decomposition multiobjective optimization. IEEE Transactions on Cybernetics **48**(12), 3359–3370 (2018)
16. Luque, M., Ruiz, F., Miettinen, K.: Global formulation for interactive multiobjective optimization. OR Spectrum **33**(1), 27–48 (2011)
17. Meignan, D., Knust, S., Frayret, J.M., Pesant, G., Gaud, N.: A review and taxonomy of interactive optimization methods in operations research. ACM Transactions on Interactive Intelligent Systems **5**(3), 1–43 (2015)
18. Miettinen, K.: Nonlinear Multiobjective Optimization. Kluwer Academic Publishers, Boston (1999)
19. Miettinen, K., Hakanen, J., Podkopaev, D.: Interactive nonlinear multiobjective optimization methods. In: Greco, S., Ehrgott, M., Figueira, J.R. (eds.) Multiple Criteria Decision Analysis: State of the Art Surveys, pp. 927–976. Springer, New York (2016)
20. Miettinen, K., Mäkelä, M.M.: On scalarizing functions in multiobjective optimization. OR Spectrum **24**(2), 193–213 (2002)
21. Miettinen, K., Mäkelä, M.M.: Synchronous approach in interactive multiobjective optimization. European Journal of Operational Research **170**(3), 909–922 (2006)
22. Nakayama, H., Sawaragi, Y.: Satisficing trade-off method for multiobjective programming. In: Grauer, M., Wierzbicki, A.P. (eds.) Interactive Decision Analysis. pp. 113–122. Springer, Berlin, Heidelberg (1984)
23. Ruiz, A.B., Luque, M., Miettinen, K., Saborido, R.: An interactive evolutionary multiobjective optimization method: Interactive WASF-GA. In: Gaspar-Cunha, A., Henggeler Antunes, C., Coello, C.C. (eds.) Evolutionary Multi-Criterion Optimization, 8th International Conference, Proceedings. pp. 249–263. Springer, Cham (2015)
24. Ruiz, F., Luque, M., Miettinen, K.: Improving the computational efficiency in a global formulation (GLIDE) for interactive multiobjective optimization. Annals of Operations Research **197**(1), 47–70 (2012)
25. Sawaragi, Y., Nakayama, H., Tanino, T.: Theory of Multiobjective Optimization. Elsevier (1985)
26. Thiele, L., Miettinen, K., Korhonen, P.J., Molina, J.: A preference-based evolutionary algorithm for multi-objective optimization. Evolutionary Computation **17**(3), 411–436 (2009)
27. Vesikar, Y., Deb, K., Blank, J.: Reference point based NSGA-III for preferred solutions. In: Proceedings of the 2018 IEEE Symposium Series on Computational Intelligence (SSCI). pp. 1587–1594 (2018)
28. Wierzbicki, A.P.: The use of reference objectives in multiobjective optimization. In: Fandel, G., Gal, T. (eds.) Multiple criteria decision making theory and application, pp. 468–486. Springer (1980)
29. Wierzbicki, A.P.: A mathematical basis for satisficing decision making. Mathematical Modelling **3**(5), 391–405 (1982)
30. Xin, B., Chen, L., Chen, J., Ishibuchi, H., Hirota, K., Liu, B.: Interactive multiobjective optimization: A review of the state-of-the-art. IEEE Access **6**, 41256–41279 (2018)
31. Zhang, Q., Li, H.: MOEA/D: A multiobjective evolutionary algorithm based on decomposition. IEEE Transactions on Evolutionary Computation **11**(6), 712–731 (2007)
32. Zitzler, E., Deb, K., Thiele, L.: Comparison of multiobjective evolutionary algorithms: Empirical results. Evolutionary Computation **8**(2), 173–195 (2000)