Review

MCDM, EMO and Hybrid Approaches: Tutorial and Review

Ankur Sinha [1,*] and **Jyrki Wallenius** [2]

1   Centre for Innovation Incubation and Entrepreneurship (CIIE), Indian Institute of Management Ahmedabad, Ahmedabad 380015, India
2   School of Business, Aalto University, P.O. Box 11000, 00076 Aalto, Finland
*   Correspondence: asinha@iima.ac.in

Abstract: Most of the practical applications that require optimization often involve multiple objectives. These objectives, when conflicting in nature, pose both optimization as well as decision-making challenges. An optimization procedure for such a multi-objective problem requires computing (computer-based search) and decision making to identify the most preferred solution. Researchers and practitioners working in various domains have integrated computing and decision-making tasks in several ways, giving rise to a variety of algorithms to handle multi-objective optimization problems. For instance, an *a priori* approach requires formulating (or eliciting) a decision maker's value function and then performing a one-shot optimization of the value function, whereas an *a posteriori* decision-making approach requires a large number of diverse Pareto-optimal solutions to be available before a final decision is made. Alternatively, an *interactive* approach involves interactions with the decision maker to guide the search towards better solutions (or the most preferred solution). In our tutorial and survey paper, we first review the fundamental concepts of multi-objective optimization. Second, we discuss the classic interactive approaches from the field of Multi-Criteria Decision Making (MCDM), followed by the underlying idea and methods in the field of Evolutionary Multi-Objective Optimization (EMO). Third, we consider several promising MCDM and EMO hybrid approaches that aim to capitalize on the strengths of the two domains. We conclude with discussions on important behavioral considerations related to the use of such approaches and future work.

Keywords: evolutionary multi-objective optimization; multi-criteria decision making; interactive optimization

1. Introduction

Multi-Criteria Decision Making (MCDM) as a scientific field is some 60 years old. Its roots are in Goal Programming [1] and Multi-Attribute Utility Theory (MAUT) [2]. A subsequently popular subfield, interactive man–machine multi-objective optimization, developed greatly during the 1970s. The common frameworks used a discrete set of choices and a mathematical programming problem formulation (optimization) to solve multi-objective problems. With the interactive approaches, phases of decision making and computing would alternate. The aim was to converge towards the most preferred solution on the Pareto-optimal frontier.

Independently from MCDM, the Evolutionary Multi-Objective Optimization (EMO) approaches started developing during the 1980s [3]. Many of the EMO scholars had an engineering or a computer science background. EMO algorithms [4,5] have been applied to problems with multiple objectives for the task of finding a well-representative set of Pareto-optimal solutions. These methods [6,7] have been successful in solving a wide variety of problems with two or three objectives. However, these methodologies are criticized for their excessive computational expense, and they often tend to suffer while solving problems with objectives higher than three [8,9]. The major hindrances in handling a higher number of objectives relate to stagnation in search, increased dimensionality of Pareto-optimal front, large computational cost, and difficulty in visualization of the objective space. These

difficulties are inherent to optimization problems having a large number of objectives and are not easy to eliminate; rather, procedures to handle such difficulties need to be explored. EMO methods that are better equipped at handling a larger number of objectives are being continuously explored [10–12]. Some of these approaches aim for solutions that are near Pareto-optimal and provide a discretized and diverse representation of the high-dimensional frontier for many-objective (i.e., more than two or three objectives) problems. However, the level of discretization for an accurate and well-represented many-objective frontier would require a very large number of points. Even if a fine-grained discretization is achieved with a large number of points, the decision-making challenges still remain.

The areas of MCDM and EMO were solving similar problems; therefore, the researchers working in these domains decided to pursue active collaboration through formal channels such as common conferences and seminars. As a result, Branke, Deb, Miettinen, and Słowiński organized the first Dagstuhl seminar [13] in 2004 to allow collaboration between the two communities. This led to researchers combining ideas from MCDM to EMO and vice-versa. Since then, the Dagstuhl seminar has been organized every few years to enhance the collaboration and flow of ideas from one research community to the other. In this article, we evaluate the classic studies in MCDM and EMO and also the hybrid approaches that have been proposed for handling many-objective problems. Some of the review papers that talk about interactive multi-objective optimization are [14,15]. This article takes a tutorial-cum-review approach to discuss the classic ideas published in the areas of MCDM, EMO, and their intersection and is structured as follows. In Section 2, we cover the theoretical concepts on optimization and decision making that arise in the multi-objective literature. This is followed by Section 3, where we discuss how search and decision making can be integrated together in various ways to find the most preferred point for the decision maker (DM). Thereafter, we discuss the classic MCDM (Section 4), EMO (Section 5), and hybrid (Section 6) approaches that have been discussed in the literature over the past few decades. We conclude the article in Section 7 with discussions on behavioral considerations and future work.

2. Multi-Objective Optimization

Multi-objective optimization [4,16–18] involves two or more conflicting objectives that are supposed to be simultaneously optimized subject to a given set of constraints. These problems arise in various fields of science, engineering, economics, and mathematics and have been widely studied in the literature. However, modern applications keep posing challenges with an increasing level of complexity. The complexity depends on a number of factors, such as number of objectives, number of decision variables, type of decision variables (continuous, discrete), number of constraints, and functional form of the functions in the optimization problems (linear, convex, non-convex, non-differentiable, etc.) that may lead to non-separability and multi-modality. While many of the above difficulties are common to single-objective optimization as well, multi-objective optimization poses additional challenges as such problems do not have a single solution which would simultaneously maximize/minimize each of the objectives; instead, there is a set of solutions from which a rational DM should choose. These solutions are called Pareto-optimal solutions. Choosing the most preferred solution from the set of Pareto-optimal solutions requires an additional step of decision making, which is often subjective and not straightforward to model. The challenges posed by multi-objective optimization often include inability to generate a complete ordering of points and requirement of maintaining a pool of non-dominated points. A feasible point in multi-objective optimization is considered to be non-dominated within a set when there does not exist any other feasible point that is better than the former in terms of some objective and is not worse than the former in terms of other objectives. The concept is discussed in detail in Section 2.1. Difficulty in representation and visualization of the solutions in objective space, especially while working with many objectives, makes decision making difficult, and therefore requires preference learning while searching for

the point most preferred by the DM. Below, we describe a general multi-objective problem ($p \geq 2$):

$$\begin{aligned} \text{Maximize} \quad & \mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_p(\mathbf{x})) \\ \text{subject to} \quad & \mathbf{g}(\mathbf{x}) \geq \mathbf{0}, \mathbf{h}(\mathbf{x}) = \mathbf{0} \\ & \mathbf{x}^{(L)} \leq \mathbf{x} \leq \mathbf{x}^{(U)} \end{aligned} \tag{1}$$

In the above formulation, $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ is the n-dimensional decision variable vector which represents the decision space. A search is expected to be performed within the constrained region of the decision space that is determined by the inequality constraints ($\mathbf{g}(\mathbf{x}) \geq \mathbf{0}$), equality constraints ($\mathbf{h}(\mathbf{x}) = \mathbf{0}$) and box constraints ($\mathbf{x}^{(L)} \leq \mathbf{x} \leq \mathbf{x}^{(U)}$). We refer to the set of solutions which are feasible with respect to the constraints and are *non-dominated* with respect to all feasible solutions, as *Pareto-optimal* solutions. Among the Pareto-optimal solutions, the solution that is the most preferred by the DM will be referred to as the *most preferred solution*. We provide formal definitions for these terms in the next sections.

Note that the objective vector $\mathbf{f}(\mathbf{x})$ is the image of the decision vector $\mathbf{x}$ under the objective function $\mathbf{f}$. In a single-objective optimization ($p = 1$) problem, the feasible set is completely ordered according to the objective function $\mathbf{f}(\mathbf{x}) = f_1(\mathbf{x})$, such that for solutions, $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$ in the decision space, either $f_1(\mathbf{x}^{(1)}) \geq f_1(\mathbf{x}^{(2)})$ or $f_1(\mathbf{x}^{(2)}) \geq f_1(\mathbf{x}^{(1)})$. Therefore, for two solutions in the objective space, there are two possibilities with respect to the $\geq$ relation. However, when several objectives ($p \geq 2$) are involved, the feasible set is not necessarily completely ordered but partially ordered. In multi-objective problems, for any two objective vectors, $\mathbf{f}(\mathbf{x}^{(1)})$ and $\mathbf{f}(\mathbf{x}^{(2)})$, the relations $=$, $>$ and $\geq$ can be extended as follows:

- $\mathbf{f}(\mathbf{x}^{(1)}) = \mathbf{f}(\mathbf{x}^{(2)}) \Leftrightarrow f_i(\mathbf{x}^{(1)}) = f_i(\mathbf{x}^{(2)}) : i \in \{1, 2, \ldots, p\}$
- $\mathbf{f}(\mathbf{x}^{(1)}) \geq \mathbf{f}(\mathbf{x}^{(2)}) \Leftrightarrow f_i(\mathbf{x}^{(1)}) \geq f_i(\mathbf{x}^{(2)}) : i \in \{1, 2, \ldots, p\}$
- $\mathbf{f}(\mathbf{x}^{(1)}) > \mathbf{f}(\mathbf{x}^{(2)}) \Leftrightarrow f_i(\mathbf{x}^{(1)}) > f_i(\mathbf{x}^{(2)}) : i \in \{1, 2, \ldots, p\}$

While comparing the multi-objective scenario with the single-objective case, we find that for two solutions in the objective space there are three possibilities with respect to the $\geq$ relation. These possibilities are: $\mathbf{f}(\mathbf{x}^{(1)}) \geq \mathbf{f}(\mathbf{x}^{(2)})$, $\mathbf{f}(\mathbf{x}^{(2)}) \geq \mathbf{f}(\mathbf{x}^{(1)})$ or $\mathbf{f}(\mathbf{x}^{(1)}) \not\geq \mathbf{f}(\mathbf{x}^{(2)}) \wedge \mathbf{f}(\mathbf{x}^{(2)}) \not\geq \mathbf{f}(\mathbf{x}^{(1)})$. If any of the first two possibilities are met, it allows to rank or order the solutions independent of any preference information (or a DM). On the other hand, if the first two possibilities are not met, the solutions cannot be ranked or ordered without incorporating preference information (or involving a DM). Drawing analogy from the above discussion, the relations $<$ and $\leq$ can be extended in a similar way.

2.1. Dominance Concept

Based on the established binary relations for two vectors in the previous section, the following dominance concept [16] can be constituted:

- $\mathbf{x}^{(1)}$ strongly dominates $\mathbf{x}^{(2)} \Leftrightarrow \mathbf{f}(\mathbf{x}^{(1)}) > \mathbf{f}(\mathbf{x}^{(2)})$;
- $\mathbf{x}^{(1)}$ (weakly) dominates $\mathbf{x}^{(2)} \Leftrightarrow \mathbf{f}(\mathbf{x}^{(1)}) \geq \mathbf{f}(\mathbf{x}^{(2)}) \wedge \mathbf{f}(\mathbf{x}^{(1)}) \neq \mathbf{f}(\mathbf{x}^{(2)})$;
- $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$ are non-dominated with respect to each other $\Leftrightarrow \mathbf{f}(\mathbf{x}^{(1)}) \not\geq \mathbf{f}(\mathbf{x}^{(2)}) \wedge \mathbf{f}(\mathbf{x}^{(2)}) \not\geq \mathbf{f}(\mathbf{x}^{(1)})$.

In the case of weak dominance, it is common to drop the word *weak* and refer to it only with *dominance*, which is why we use the word *weak* in brackets. Dominance of $\mathbf{x}^{(1)}$ over $\mathbf{x}^{(2)}$ essentially means that no component of $\mathbf{f}(\mathbf{x}^{(1)})$ is less than the corresponding component of $\mathbf{f}(\mathbf{x}^{(2)})$, and at least one component of $\mathbf{f}(\mathbf{x}^{(1)})$ is greater than the corresponding component of $\mathbf{f}(\mathbf{x}^{(2)})$. The above dominance concept is also explained in Figure 1 for a two-objective maximization case. In Figure 1, two shaded regions are shown in reference to point A. The shaded region in the north-east corner (excluding the lines) is the region which strongly dominates point A, the shaded region in the south-west corner (excluding the lines) is strongly dominated by point A, and the non-shaded region is the non-dominated region.

Therefore, point A strongly dominates point B, and points A, E, and D are non-dominated with respect to each other. Note that point A weakly dominates point C. From hereon, we only talk about dominance by avoiding the word *weak*.
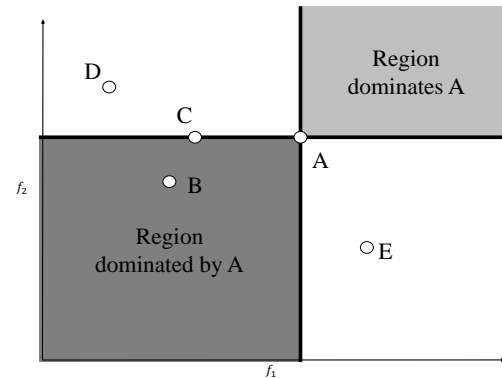

Figure 1. Dominance concept for a maximization problem where A dominates B and C; A, D, and E are non-dominated.

Many of the existing evolutionary multi-objective optimization algorithms use the dominance principle to converge towards the Pareto-optimal set of solutions. The concept allows us to partially order two decision vectors based on the corresponding objective vectors in the absence of any preference information. The algorithms which operate with a sparse set of solutions in the decision space and the corresponding images in the objective space usually give priority to a solution which dominates another solution. The solution which is not dominated with respect to any other solution in the sparse set is referred to as a *non-dominated solution* within that set.

In case of a discrete set of solutions: the subset whose solutions are not dominated by any solution in the discrete set is referred to as the *non-dominated set* within the discrete set. When the set in consideration is the entire search space, the resulting non-dominated set is referred to as a *Pareto-optimal set*, or the frontier formed with these points is referred to as the *Pareto-optimal front*. To formally define a Pareto-optimal front, consider a set $\mathbf{X}$ which constitutes the entire decision space with solutions $\mathbf{x} \in \mathbf{X}$. The subset $\mathbf{X}^* : \mathbf{X}^* \subset \mathbf{X}$, containing solutions $\mathbf{x}^*$ which are not dominated by any $\mathbf{x}$ in the entire decision space, forms the Pareto-optimal front.

The concept of a Pareto-optimal front and a non-dominated set are illustrated in Figure 2. The shaded region in the figure represents $\mathbf{f}(\mathbf{x}) : \mathbf{x} \in \mathbf{X}$. It is the image in the objective space of the entire feasible region in the decision space. The bold curve represents the Pareto-optimal front for a maximization problem. Mathematically, this curve is $\mathbf{f}(\mathbf{x}^*) : \mathbf{x}^* \in \mathbf{X}^*$, which are all the optimal points for the two objective optimization problem. A number of points are also plotted in the figure, which constitute a discrete set. Among this set of points, the points connected by broken lines are the points which are not dominated by any point in the discrete set. Therefore, these points constitute a non-dominated set within the discrete set. The other points which do not belong to the non-dominated set are dominated by at least one of the points in the non-dominated set.

In the field of MCDM, a Pareto-optimal point $\mathbf{f}(\mathbf{x}^*)$ in the objective space is often referred to as a non-dominated point, as it is not dominated by any feasible point in the objective space. The corresponding decision vector $\mathbf{x}^*$ is referred to as an *efficient point*. Similarly, if $\mathbf{f}(\mathbf{z})$ is a dominated point in the objective space, then $\mathbf{z}$ would be referred to as an *inefficient point* in the decision space. In other words, a point is efficient if and only if it is the inverse image of a non-dominated objective vector, and it is inefficient if and only if it is an inverse image of a dominated objective vector.
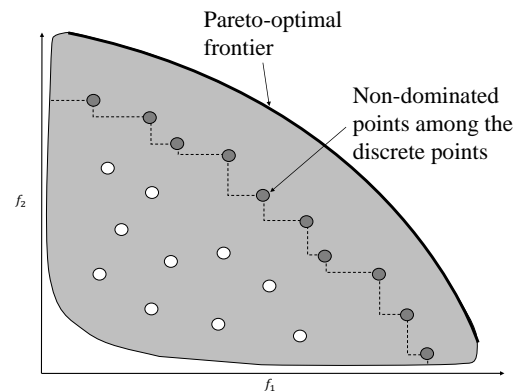
Figure 2. Non-dominated set from a discrete set of points and a Pareto-optimal front that dominates the entire search space.

2.2. Decision Making

Even though there are multiple potentially optimal solutions to a multi-objective problem, there is often just a single solution which is of interest to the DM; which is termed as the *most preferred solution*. Search and decision making are two intricacies [19] involved in handling any multi-objective problem. Search requires an intensive exploration in the decision space to get close to the Pareto-optimal solutions; on the other hand, decision making is required to provide preference information over the available non-dominated solutions in pursuance of the most preferred solution.

In a decision-making context, the solutions can be compared and ordered based on the preference information, though there can be situations where strict preference of one solution over the other is not obtained, and the ordering is partial. For instance, consider two vectors, $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$, in the decision space, having their images, $\mathbf{f}(\mathbf{x}^{(1)})$ and $\mathbf{f}(\mathbf{x}^{(2)})$, in the objective space. A preference structure can be defined using three binary relations $\succ$, $\sim$, and $\|$. The meaning of the binary relations are provided below:

- $\mathbf{x}^{(1)} \succ \mathbf{x}^{(2)} \Leftrightarrow \mathbf{x}^{(1)}$ is preferred over $\mathbf{x}^{(2)}$;
- $\mathbf{x}^{(1)} \sim \mathbf{x}^{(2)} \Leftrightarrow \mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$ are equally preferable;
- $\mathbf{x}^{(1)} \| \mathbf{x}^{(2)} \Leftrightarrow \mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$ are incomparable;

where the preference relation, $\succ$, is asymmetric, the indifference relation, $\sim$, is reflexive and symmetric, and the incomparability relation, $\|$, is irreflexive and symmetric. A weak preference $\succeq$ relation can be established as $\succeq = \succ \cup \sim$ such that

- $\mathbf{x}^{(1)} \succeq \mathbf{x}^{(2)} \Leftrightarrow \mathbf{x}^{(1)}$ is either preferred over $\mathbf{x}^{(2)}$ or they are equally preferred.

As already mentioned, preference can easily be established for pairs where one solution dominates the other. However, for pairs which are non-dominated with respect to each other, a DM's input is required to establish a preference. The following is the inference for preference choice which can be drawn from dominance:

- If $\mathbf{x}^{(1)}$ dominates $\mathbf{x}^{(2)} \Rightarrow \mathbf{x}^{(1)} \succ \mathbf{x}^{(2)}$.

It is common to emulate a DM with a value function, $V(f_1(\mathbf{x}), \ldots, f_p(\mathbf{x}))$, which is scalar in nature and assigns a value or a measure of satisfaction to each of the solutions. For two solutions, $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$:

- If $\mathbf{x}^{(1)} \succ \mathbf{x}^{(2)} \Leftrightarrow V(\mathbf{f}(\mathbf{x}^{(1)})) > V(\mathbf{f}(\mathbf{x}^{(2)}))$;
- If $\mathbf{x}^{(1)} \sim \mathbf{x}^{(2)} \Leftrightarrow V(\mathbf{f}(\mathbf{x}^{(1)})) = V(\mathbf{f}(\mathbf{x}^{(2)}))$;
- If $\mathbf{x}^{(1)} \succeq \mathbf{x}^{(2)} \Leftrightarrow V(\mathbf{f}(\mathbf{x}^{(1)})) \geq V(\mathbf{f}(\mathbf{x}^{(2)}))$.

2.3. Preference Eliciting and Modeling

There are several ways of eliciting preference information from the DM that can be used to create a preference model to be incorporated in the search process. Some of the approaches are listed below:

1. Asking about goals or aspiration levels for the objectives;
2. Pairwise comparisons of solutions in objective space;
3. Asking the DM which objectives and by how much they would be willing to worsen to allow improvements in other objectives;
4. Asking the DM to specify exact marginal rates of substitution between objectives and a reference objective (trade-offs);
5. Directly asking for the search direction;
6. Directly asking the importance of each objective to get an idea of weights or to rank the objectives;
7. Yes–no questions, for instance: Do you like this search direction?

After the preferences are obtained from the DM, there are various ways in which the information is incorporated in the search process. For instance, value functions could be generated based on the preferences expressed by the DM. Methods differ based on the kind of value function, i.e., linear or non-linear, that is chosen to model preference information. While some methods generate a single maximum discriminating value function fitting preference information, others generate multiple value functions fitting the same preference information. Scalarizing functions (for example, see [20]), weighted sum of objectives (similar to a linear value function), and the ϵ-constraint method [21] are other approaches to convert the multi-objective problem into a single-objective problem that aligns with the DM's preferences. Sometimes, the dominance principle is modified to search in a region that better fits the preferences of the DM.

There are other very interesting approaches to modeling preferences in MCDM. Such approaches are outranking relations and rule-based models. Outranking methods were developed by B. Roy in the late 1960s, originating from criticism of utility theory in solving practical problems (see [22,23]). An outranking relation is a binary relation. It is based on the ideas of concordance and discordance. "Loosely speaking", alternative **x** outranks **y**, if there are enough arguments (attributes favoring **x** over **y**) to declare that **x** is at least as good as **y** while there is no essential reason to refute this statement. Decision rules are expressions of the form "if, then" [24]. Procedures for generating decision rules use an inductive learning principle. The authors distinguish three types of rules: certain, possible, and approximate. Certain rules are generated from lower approximations of unions of classes; possible rules are generated from upper approximations of unions of classes and approximate rules are generated from boundary regions. To structure the data prior to the induction of rules, the authors suggest using the Dominance-based Rough Set Approach (DRSA) [25]. As an illustrative example, the authors consider the problem of evaluating high school students based on performance in some of the subjects using "if, then" rules. Multi-criteria classification and sorting are frequently considered problems in rule-based preference modeling, although the ideas can be extended for the problem of identifying the most preferred alternative. Both outranking relations and rule-base preference modeling were originally developed for the problem of choosing among discrete (known) alternatives, and not the mathematical programming or EMO context. Hence, we do not extensively cover them in our survey and tutorial. There are some exceptions, though. For example, the Light Beam search approach (which is based on utilizing outranking relations) was developed for solving multi-objective mathematical programming problems [26]. In a later section, we illustrate how the Light Beam approach is used in an EMO context.

In the later part of the paper, we discuss approaches that elicit and model the preferences of the DM in different ways while searching for the most preferred point.

3. Incorporating Decision Maker's Preferences

Searching and decision making can be combined in various ways to generate procedures which can be classified into three broad categories.

3.1. A Priori Approach

In this approach, DM's preferences are elicited before the start of the algorithm, then the optimization algorithm is executed by incorporating the preference information, and the most preferred solution is identified. Figure 3 shows the process followed to arrive at the most preferred solution. This approach has been common among MCDM practitioners, who realized the complexities involved in decision making for such problems. Their approach to the problem is to ask simple questions from the DM before starting the search process.
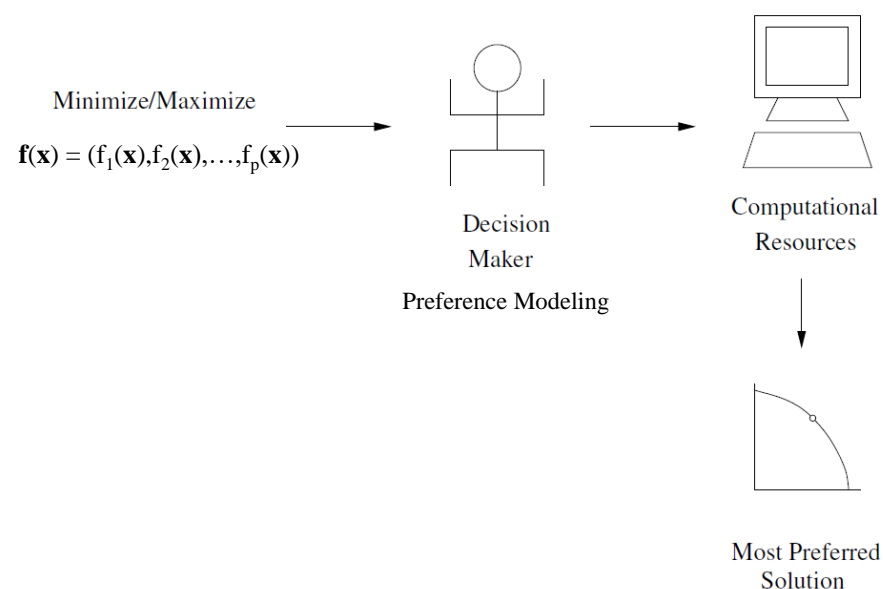
Minimize/Maximize

$\mathbf{f(x)} = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_p(\mathbf{x}))$

Decision
Maker

Preference Modeling

Computational
Resources

Most Preferred
Solution

Figure 3. *A priori* approach.

After eliciting information from the DM, the multi-objective problem is usually converted into a single-objective problem. One of the early approaches, that is, MAUT [2], used the initial information from the DM to construct a utility function which reduced the problem to a single-objective optimization problem. Scalarizing functions (for example, [20]) are also commonly used by the researchers in this field to convert a multi-objective problem into a single-objective problem.

Since information is elicited towards the beginning, the solution obtained after executing the algorithm may not be close to the most preferred solution. Moreover, the DM's preferences might be different for solutions close to the Pareto-optimal front, and the initial inputs taken from them may not conform to it. Therefore, relying on this approach, it may be difficult to get close to the actual solution which meets the requirements of the DM. The approach is also highly error-prone, as even slight deviations in providing preference information at the beginning may lead to entirely different solutions. Such errors are common because of the inability of the DM to reliably express preferences in case of not knowing the solution space or having no precise understanding of own preferences at the beginning of the preference elicitation process. To avoid the errors due to deviations, researchers in the EMO field used the approach in a slightly modified way. They produced multiple solutions in the region of interest to the DM (often close to the Pareto-optimal front) [27–30], and then elicited the DM's preferences. We discuss this approach next.

3.2. A Posteriori Approach

In this approach, after a set of (approximate) Pareto-optimal solutions are obtained using an optimization algorithm, decision making is performed to find the most preferred solution. Figure 4 shows the process followed to arrive at the final solution which is most preferred to a DM. This approach is based on the assumption that a complete knowledge of all the alternatives helps in taking better decisions. The research in the field of evolutionary multi-objective optimization has been directed along this approach, where the aim is to

produce all the possible alternatives for the DM to make a choice. Until relatively recently, the community has largely ignored decision-making aspects and has been striving towards producing all the possible optimal solutions.
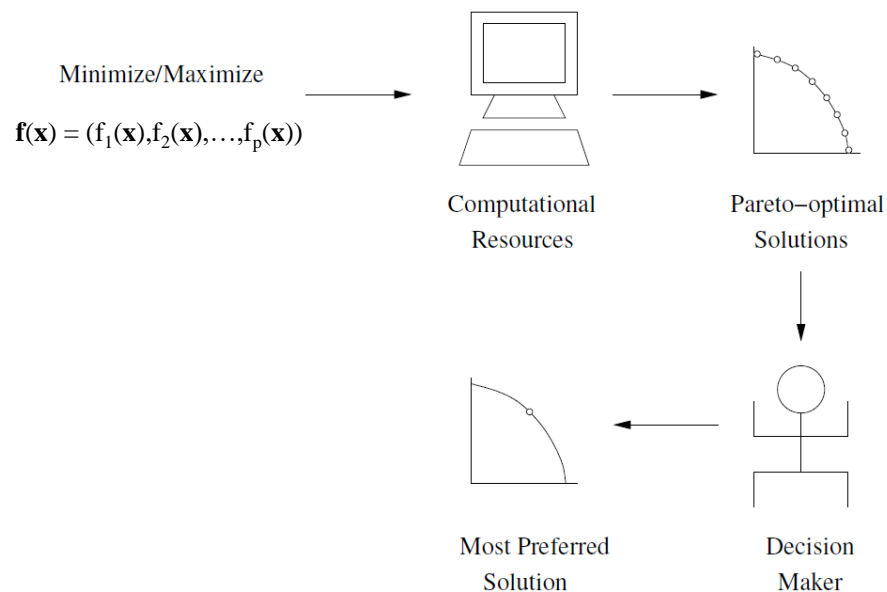

Figure 4. *A posteriori* approach.

There are enormous difficulties in finding the entire Pareto-optimal front for a many-objective problem. Even if it is assumed that an algorithm can approximate the Pareto-optimal front for a high-objective problem with a huge set of points, the herculean task of choosing the best point from the set still remains. For two and three objectives where the solutions in the objective space could be represented geometrically, making decisions might be easy (though even such an instance could be, in reality, a difficult task for a DM). Imagine a multi-objective problem with more than three objectives for which an evolutionary multi-objective algorithm is able to produce the entire front. The front is approximated with a large number of points and high accuracy. Since a graphical representation is not possible for the Pareto-points, how is a DM going to choose the most preferred point? There are of course decision aids available, but the limited accuracy with which the final choice could be made using these aids questions the purpose of producing the entire front with high accuracy. Binary comparisons can be a solution to choose the best point out of a set, but this can only be utilized if the points are very few in number. Therefore, offering the entire set of Pareto-points should not be considered as a complete solution to the problem. However, the difficulties related to decision making have been realized by EMO researchers only after copious research has already gone towards producing the entire Pareto-front for many-objective problems. Most of the EMO algorithms [6,7,10–12,31–33] that aim to produce the entire Pareto-optimal front would lie in this category.

3.3. Interactive Approach

In this approach, the DM interacts with the optimization algorithm and has multiple opportunities to provide preference information to the algorithm. The interaction between the DM and the optimization algorithm continues until a solution acceptable to the DM is obtained. The process is represented in Figure 5. Based on the type of interaction of the DM with the optimization algorithm, this approach is often implemented in two ways.
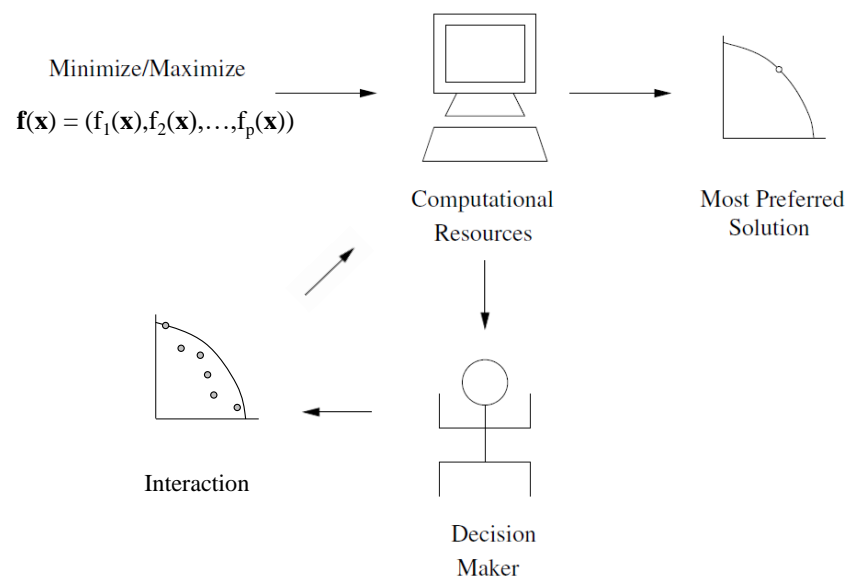
Figure 5. Interactive approach.

The first approach involves elicitation of preference information and execution of the optimization algorithm to obtain one or many Pareto-optimal solutions. If a solution acceptable to the DM is obtained, the process is terminated; otherwise, the process is restarted and continued until a satisfactory solution is found. In this approach, the progression towards the most preferred solution may take place on the Pareto-optimal frontier. MCDM researchers following an interactive approach usually elicit preference information and find a solution conforming to the inputs given by the DM. They iterate this process until a satisfactory solution is obtained. For example, when using a scalarization function, multiple reference points (or starting points) could be provided by the DM. Once a reference point is available, the computer provides a projection of that point on the Pareto-optimal frontier. This process converts the problem into a single-objective optimization problem and produces one of the Pareto-optimal points as the solution. If the point finally produced is not to the liking of the DM, the search is continued with new reference points and projections. This process is continued until a solution acceptable to the DM is obtained. The iterations of a simple algorithm using this approach are shown in Figure 6. The figure shows that a DM is able to find a satisfactory solution in three iterations.
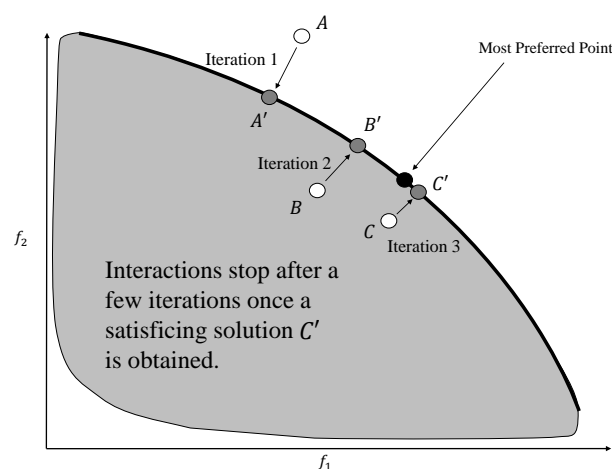

Figure 6. Interaction after a run.

EMO researchers have taken cues from their MCDM counterparts, wherein they used the powerful evolutionary search tool to produce multiple solutions in the region of interest

to the DM or generate a small part of the Pareto-front which the DM finds interesting. This is a similar approach where interactions happen before and after a complete run of the EMO. The algorithm produces multiple solutions in a particular region or multiple regions of the Pareto-optimal front in a single run. Once the solutions are produced, another decision-making task is performed, and the solution to the liking of the DM is chosen. If none of the solutions are acceptable to the DM, the process of elicitation and search is repeated until a satisfactory solution is found. Some examples of evolutionary procedures which have used this approach are [34,35].

The second approach involves elicitation of preference information periodically from a DM while the optimization algorithm is progressing towards the Pareto-optimal frontier. In this approach, preference information is taken at the intermediate steps of the search algorithm, and the algorithm proceeds towards the most preferred point. This is an effective integration of the search and decision-making process, as both work simultaneously towards the exploration of the solution. Such an integration avoids multiple optimization runs and is therefore preferable for problems that are computationally expensive. It also allows the DM to better understand the consequences of their actions, as they can immediately see how the convergence direction changes. Some previous works which have been conducted in a similar vein in the MCDM field are [36,37], and in the EMO field, are [38–44]. The iterations of an algorithm that uses this approach, commonly referred to as a progressively interactive approach, is shown in Figure 7. The DM is presented with a set of points and is expected to choose one of the points to start the search. The choice of the DM gives clues to the search algorithm about the search direction, and the algorithm progresses towards the most preferred solution. The DM may change their preference structure as the search progresses, and the algorithm is able to adapt to such changes.
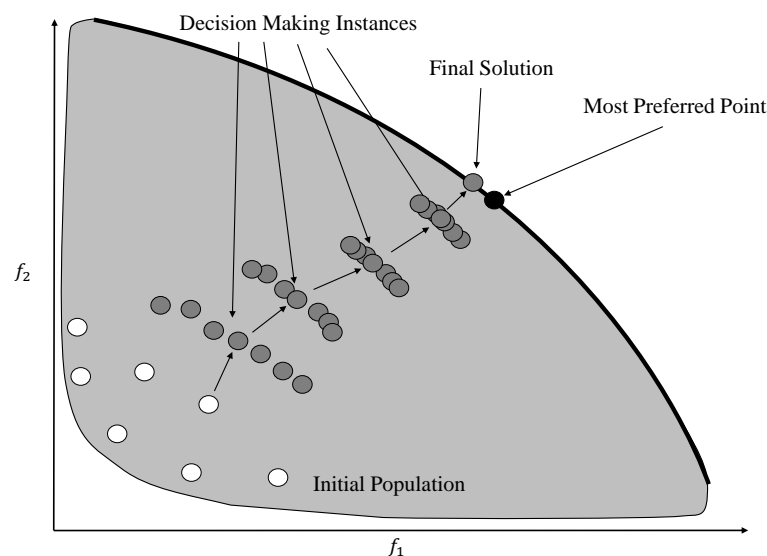

Figure 7. Progressive interaction during the run.

4. MCDM Interactive Techniques

Linear Programming (LP) was rather popular in large Western companies in the 1960s and 1970s, as well as in Gosplan (central government agency) for government level planning in the Soviet Union. To address the need to solve multi-objective LPs, Charnes and Cooper developed Goal Programming [1] in late 1950s and coined the name in the early 1960s. In Goal Programming, the DM is asked to specify aspiration levels in terms of objectives. The algorithm then finds a feasible solution that would minimize the weighted deviations from the aspiration levels. The original version of Goal Programming was for solving multiple-objective LPs. Goal Programming was not an interactive approach, and there was not an

option to update the aspiration levels. In multi-objective linear programs, the concept of an optimum was being replaced by a "compromise" or a "non-dominated solution".

With simultaneous advances in computer technology (teletypes accessing main frame computers), the idea of interactively or progressively solving multi-objective optimization problems was proposed in early 1970s. In the interactive approach:

1. Phases of computing and decision making would alternate: the human would guide the computer (algorithm) towards the most preferred solution;
2. The human and the computer were performing tasks that they were good at;
3. Learning (of one's preferences) was possible;
4. The ideas were based on using linear programming or non-linear programming;
5. Systematic progress towards the most preferred solution would take place;
6. The methods would generally operate with non-dominated solutions, in other words, allow exploration of the Pareto-optimal (non-dominated) frontier.

We review the following classic interactive multi-objective optimization methods, which all represented the state of the art at the time:

1. STEP method due to Benayoun et al. (1971) [45];
2. GDF method due to Geoffrion, Dyer, and Feinberg (1972) [36];
3. ZW method due to Zionts and Wallenius (1976) [37];
4. Reference point method due to Wierzbicki (1980) [20];
5. Reference direction method due to Korhonen and Laakso (1986) [46];
6. Pareto Race due to Korhonen and Wallenius (1988) [47];

4.1. STEP Method (Benayoun et al., 1971) [45]

The ancestor of the STEP method [45] was the Progressive Orientation Procedure (POP) by Benayoun and Tergny [48]. In the POP method, a subset of efficient extreme points is computed and presented to the DM for her evaluation. The DM can either choose the most preferred solution, or choose an attractive subset, and so forth. The STEP method was one of the first truly interactive approaches for solving multi-objective LPs. In this man–model symbiosis, phases of computation alternate with phases of decision. The process allows the DM to learn to recognize good solutions and the relative importance of the objectives.

In the STEP method, each objective is optimized one at a time to obtain the ideal point of the problem. For a maximization problem, the components of the ideal point describe the upper bounds of the individual objectives for the points corresponding to the Pareto-optimal front. Similarly, the nadir point (not used in STEP method) is defined as the lower bounds of the individual objectives for the points corresponding to the Pareto-optimal front. Denote the ideal point as $\mathbf{M} = (M_1, M_2, \ldots, M_p)$. At each iteration, the following LP problem is solved to obtain the feasible compromise solution $\mathbf{x}^{(k)}$ (k is the iteration counter), which is nearest in the *minimax* sense to $\mathbf{M}$:

$$
\begin{aligned}
\text{Minimize} \quad & q \\
\text{subject to} \quad & q \geq (M_i - f_i(\mathbf{x}))\lambda_i \ \forall \ i \in 1, \ldots, p \\
& \mathbf{x} \in B^k \\
& q \geq 0
\end{aligned}
\tag{2}
$$

where B^k is the feasible region at iteration k, $f_i(\mathbf{x})$ is the function for the ith objective at decision $\mathbf{x}$, and λ_i is the set of normalized weights (not specified by the DM). At the decision phase, the objective function vector associated with the compromise solution $\mathbf{x}^{(k)}$ is presented to the DM. Next, the DM must choose the objectives f_{i^*} (if any), where $i^* \subset \{1, \ldots, p\}$, which they would be willing to worsen to allow an improvement in the unsatisfactory ones. Then, the DM must specify the maximal amount of relaxation in the above objectives. At the next iteration, the feasible region is modified as $B^{k+1} = \{\mathbf{x} : f_j(\mathbf{x}) \geq f_i(\mathbf{x}^k) - \Delta f_j, f_i(\mathbf{x}) \geq f_i(\mathbf{x}^k) \ \forall \ i \notin i^*, j \in i^*\}$. The weights of the objectives to be relaxed are set to 0, and the next calculation phase is performed. The process is terminated as soon as the DM has found a satisfactory solution. The solutions at termination are not necessarily

always non-dominated, but with modifications, they can all be made non-dominated. Note that the *minimax* operation corresponds to minimizing the Chebycheff norm.

4.2. GDF Algorithm (Geoffrion, Dyer, and Feinberg, 1972) [36]

In Geoffrion, Dyer, and Feinberg's algorithm [36], the problem is formulated as follows:

$$
\begin{array}{ll}
\text{Maximize} & U(f_1(\mathbf{x}), \ldots, f_p(\mathbf{x})) \\
\text{subject to} & \mathbf{x} \in \mathbf{X}
\end{array}
\tag{3}
$$

where $\mathbf{X}$ is the feasible set (convex and compact), f_i are objective functions of the decision vector $\mathbf{x}$, and U is the overall utility (or value) function defined over the values of the objectives, assumed to be concave (under maximization) and differentiable. Everything else, except for U, is assumed to be explicitly known. U, however, is only assumed to be implicitly known. (If U were explicitly known, the problem would be an ordinary non-linear program.)

The GDF algorithm uses a modification of the Frank–Wolfe [49] algorithm from 1956. Note that the Frank-Wolfe algorithm is a steepest ascent algorithm. Two problems alternate: the direction-finding problem and the step-size problem. Let us ignore for the moment that U is not explicitly known, then the algorithm progresses as follows:

1. Choose an initial solution $\mathbf{x}^{(1)} \in \mathbf{X}$. Set $k = 1$ (iteration counter).
2. Determine an optimal solution $\mathbf{y}^{(k)}$ of the direction-finding problem

$$
\begin{array}{ll}
\text{Maximize} & \nabla_{\mathbf{x}} U(f_1(\mathbf{x}), \ldots, f_p(\mathbf{x})) \mathbf{y} \\
\text{subject to} & \mathbf{y} \in \mathbf{X}.
\end{array}
\tag{4}
$$

3. Set $\mathbf{d}^{(k)} = \mathbf{y}^{(k)} - \mathbf{x}^{(k)}$. This step determines the "best" search direction based on a linear (first-order Taylor expansion) approximation of U.
4. Next, solve the step-size problem for an optimal t:

$$
\begin{array}{ll}
\text{Maximize} & U(f_1(\mathbf{x}^{(k)} + t\mathbf{d}^{(k)}), \ldots, f_p(\mathbf{x}^{(k)} + t\mathbf{d}^{(k)})) \\
\text{subject to} & 0 \leq t \leq 1.
\end{array}
\tag{5}
$$

5. Set $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + t^k \mathbf{d}^{(k)}, k = k + 1$, and return to the direction-finding problem. Theoretical termination criterion is satisfied if $\mathbf{x}^{(k)}$ and $\mathbf{x}^{(k+1)}$ are equal.

Now, assume that we do not know U. The gradient of U can be replaced with the sum of the product of weights w_i^k times the gradient of f in terms of $\mathbf{x}$.

$$
\begin{array}{ll}
\text{Maximize} & \Sigma_{i=1}^p w_i^k \nabla_{\mathbf{x}} f_i(\mathbf{x}^{(k)}) y_i \\
\text{subject to} & \mathbf{y} \in \mathbf{X}
\end{array}
\tag{6}
$$

where we define $w_i^k = \dfrac{\partial U / \partial f_i^{(k)}}{\partial U / \partial f_j^{(k)}}, i = 1, \ldots, p$ with f_j being arbitrarily chosen as the reference criterion. The weights reflect the DM's tradeoff between f_j and f_i (at the current point), and must be elicited from the DM. We determine what change Δf_j in the reference criterion exactly compensates for a change Δf_i: $w_i^k = -\dfrac{\Delta f_j}{\Delta f_i}$. This is the Marginal Rate of Substitution (MRS) between the objectives.

The step-size problem must be solved directly by the DM. In early work, the computer would tabulate the values of the objectives at selected intervals and let the DM choose from this numerical display their most preferred solution.

4.3. ZW Method (Zionts and Wallenius, 1976) [37]

The Zionts–Wallenius method [37] is a simple-to-use multi-objective "simplex method", which companies could easily adopt for relatively large-scale problems. The authors initially made the assumption that the DM's underlying (implicit) value function would be

linear (in terms of the objectives). LP theory suggests that the optimal solution would be a non-dominated extreme point solution. Hence, it would be sufficient to operate with efficient extreme point solutions. The authors first developed a *naïve* approach, which starts with an efficient extreme point and asks the DM about neighboring extreme points: Do you prefer any of the neighboring points to the current point? If yes, the DM is moved to one of the preferred neighbors and the method continues. If not, the optimal solution (or most preferred solution) is assumed to be found. The problem with the *naïve* approach is that the convergence was awfully slow for even moderately large problems. Therefore, a more elaborate approach had to be thought through to make the algorithm more efficient.

In the elaborate approach, the process starts by assuming some arbitrary (positive) weights for the objectives. If no other information exists, one may start with equal weights. The method uses the current set of weights to generate a non-dominated solution, and then asks the DM to tell whether any of the "efficient" neighboring solutions are preferred to the current solution (or a unit movement in that direction = trade-offs). If not, the most preferred solution is found, otherwise, the process continues. Note that the trade-offs can be obtained from the simplex table corresponding to the objective function rows and the non-basic variable columns.

The following so-called "λ-problem" tells how the weights are updated based on the DM's yes/no answers:

$$
\begin{aligned}
\text{Maximize} \quad & \epsilon \\
\text{subject to} \quad & \Sigma_{i=1}^{p} \lambda_i x_i^{(r)} - \epsilon \geq \Sigma_{i=1}^{p} \lambda_i x_i^{(s)} \ \forall \ \mathbf{x}^{(r)} \in \mathbf{X}_r, \mathbf{x}^{(s)} \in \mathbf{X}_s \\
& \Sigma_{i=1}^{p} \lambda_i = 1 \\
& \lambda_i > 0, i \in 1, \ldots, p
\end{aligned}
\tag{7}
$$

The sets $\mathbf{X}_r$ and $\mathbf{X}_s$ contain points where every element in $\mathbf{X}_r$ is preferred to every element in $\mathbf{X}_s$, i.e., $\mathbf{x}^{(r)} \succ \mathbf{x}^{(s)} \ \forall \ r, s$. The updated weights are used to generate an improved non-dominated extreme point solution and the process is repeated. The process terminates when none of the neighboring extreme point solutions are preferred to the current solution, which is assumed to be the optimal solution. Note that, in this approach, it is not necessary to ask the DM about all neighboring extreme point solutions, but only the *efficient* ones. The algorithm was tested for moderately sized LP problems with 3–4 objectives.

4.4. Reference Point Method (Wierzbicki, 1980) [20]

The Reference Point method [20] asks the DM to provide aspiration levels for the objectives. The aspiration point is then projected to the non-dominated frontier. Note that it does not matter whether the aspiration point provided by the DM is feasible or not. In the projection, Wierzbicki used the so-called Achievement Scalarizing Function (ASF), which was minimized as:

$$
\begin{aligned}
\text{Min} \quad & \text{Max}_{i=1}^{p} \left(\frac{g_i - f_i(\mathbf{X})}{w_i} \right) + \rho \Sigma_{i=1}^{p} \frac{g_i - f_i(\mathbf{X})}{w_i} \\
\text{subject to} \quad & \mathbf{x} \in \mathbf{X}
\end{aligned}
\tag{8}
$$

where $w_i > 0$ is a set of weights, ρ is a small number, and g_i is the vector of aspiration levels. Note that when $\rho = 0$, the indifference contours being optimized are orthogonal (90 degree angle); when $\rho > 0$, the indifference contours being optimized form an angle between 90 and 180 degrees. Once the non-dominated projection of the aspiration levels is found, the method asks the DM to update the aspiration levels. The method stops when the DM is satisfied with the solution. In contrast with the GDF and ZW methods, no assumptions of U are made.

4.5. Reference Direction Approach (Korhonen and Laakso, 1986) [46]

Instead of projecting a single reference point using Wierzbicki's ASF, Korhonen and Laakso [46] suggested projecting multiple directions to the efficient frontier. The projection was determined by solving the following parametric program:

$$
\begin{aligned}
\text{Min} \quad & \epsilon \\
\text{subject to} \quad & f_i(\mathbf{x}) + \epsilon w_i \geq q_i + td_i, \ i \in 1, \ldots, p \\
& \mathbf{x} \in \mathbf{X}
\end{aligned}
\tag{9}
$$

where $w_i > 0$ is a set of weights, q_i is any vector in the criterion space, and $d_i = g_i - q_i$ is a reference direction, with g_i being an aspiration level or a reference goal in the spirit of Wierzbicki's reference point approach. When the parameter t in the above problem is varied from zero to infinity, an efficient curve emanating from point $\mathbf{q}$ is obtained.

The interface of the reference direction method is similar to the GDF method. When the DM has identified the most preferred solution along the projection, then they are asked to revise their aspiration point, and the process is repeated. An extension and application of the reference direction approach on multi-objective quadratic linear programming can be found in [50].

4.6. Pareto Race (Korhonen and Wallenius, 1988) [47]

Pareto Race [47] is a visual, dynamic search procedure for exploring the non-dominated frontier of a multi-objective LP problem. It is based on the idea of projecting reference directions on the efficient frontier. However, no aspiration levels are elicited from the DM. Instead, if the DM wants to improve the value of a certain objective, they press the number key (one or more times, depending on the relative desired improvement in that objective) of the corresponding objective.

There is an analogy to driving an automobile (on the efficient frontier). The user sees the objective function values on a display in numeric form and as bar graphs as they travel along the non-dominated frontier. Keyboard controls include accelerator, gears, breaks, and a steering mechanism. Technically, two parameters are used to control the motion: the reference direction (direction) and step size (speed). Figure 8 shows the interactive dashboard used in the Pareto Race approach.
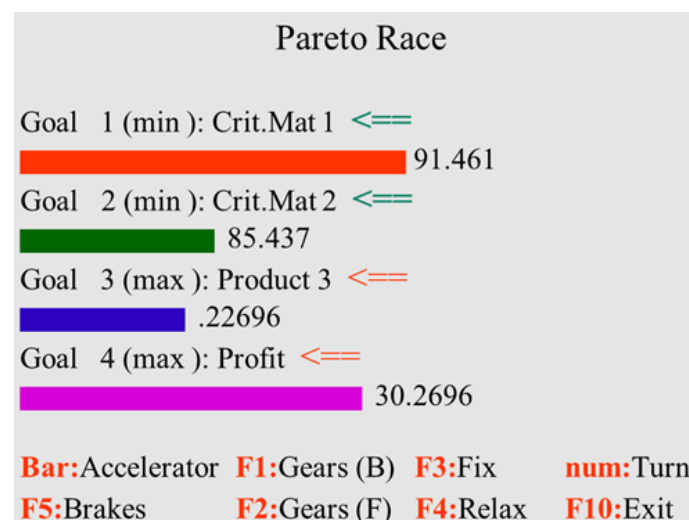

Figure 8. Pareto Race interface.

5. EMO Introduction and History

An evolutionary algorithm is a general population-based optimization algorithm which uses a mechanism inspired by biological evolution, i.e., selection, crossover, mutation, and replacement. The common underlying idea behind an evolutionary technique is

that, for a given population of individuals, the environmental pressure causes natural selection, which leads to a rise in fitness of the population. A comprehensive discussion of the principles of an evolutionary algorithm can be found in [51–55]. In contrast to classical algorithms, which iterate from one solution point to the other until termination, an evolutionary algorithm works with a population of solution points. Each iteration of an evolutionary algorithm results in an update of the previous population by eliminating inferior solution points and including the superior ones. In the terminology of evolutionary algorithms, an iteration is commonly referred to as a generation and a solution point as an individual. A pseudo-code for a general genetic algorithm, which is a type of evolutionary algorithm, is provided below:

Step 1: Create a random initial population (i.e., a set of solution points in the decision space).

Step 2: Evaluate the individuals (i.e., the solution points) in the population with respect to objective(s) and constraints, if present, and assign fitness (i.e., quality measure).

Step 3: Repeat the generations (i.e., iterations of the evolutionary algorithm) until termination.

Substep 1: Select the fitter individuals (referred to as parents) from the population for reproduction (i.e., producing new solution points through genetic operators of crossover and mutation).

Substep 2: Produce new individuals (referred to as offspring) through crossover and mutation operators.

Substep 3: Evaluate the new individuals and assign fitness.

Substep 4: Replace the low-fitness individuals in the population with high-fitness individuals that may have been generated through crossover and mutation.

Step 4: Report the highest fitness individual as the output.

Along with the pseudo-code presented above, a flowchart for a general evolutionary algorithm is presented in Figure 9. In evolutionary algorithms, to begin with, a pool of individuals is generated by randomly creating points in the search space, which is called the population. Each individual in the population is evaluated on objective(s) and constraints (if any) and is assigned a fitness. For instance, while solving a single-objective maximization problem, a solution point with a higher function value is better than a solution point with lower function value when both solutions are feasible. Therefore, in such cases, the individual with higher function value is assigned a higher fitness. While comparing two infeasible solutions, the solution with a smaller constraint violation is often assigned a higher fitness as compared with the solution with larger constraint violation. In the presence of multiple constraints, the constraint violation for a particular point is defined as the sum of violation of those constraints that are infeasible with respect to that point. While comparing a feasible solution against an infeasible solution, a feasible solution is often assigned a higher fitness as compared with the infeasible solution. There can, of course, be other ways to assign fitness. For an unconstrained maximization problem, the function value itself can be treated as the fitness value and, for an unconstrained minimization problem negative of the function value, may serve the purpose of fitness. In all such cases, the algorithm searches for a higher fitness solution.

In a multi-objective context, the requirement is to produce a set of solutions that approximate the Pareto-optimal front. Fitness assignment based on constraint violation can be performed in the multi-objective case in a similar manner as the single-objective case. Moreover, a feasible solution point which dominates another feasible solution point can be assigned a higher fitness. However, fitness assignment for two solutions that are non-dominated with respect to each other is tricky. In such cases, algorithms often consider a measure of diversity or crowdedness [4] in the objective space to assign fitness and prefer one solution over the other. The measure for crowdedness prefers solutions that are isolated over solutions that are in crowded regions to enhance diversity in the population and to

obtain a "well-spread" set of solutions approximating the Pareto-optimal front. A multi-objective evolutionary procedure, therefore, assigns fitness to each of the solution points based on their superiority over other solution points in terms of constraints, dominance, and diversity in the objective space. Different algorithms use different quality functions to assign fitness to an individual in a population. Once an initial population is generated and the fitness is assigned, a few of the better candidates from the population are chosen as parents. Crossover and mutation are performed to generate new solutions. Crossover is an operator applied to two or more selected individuals and results in one or more new individuals. Mutation is applied to a single individual and results in one new individual. Executing crossover and mutation lead to offspring that compete, based on their fitness, with the individuals in the population, for a place in the next generation. An iteration of this process often leads to a rise in the average fitness of the population and, over iterations, helps the algorithm converge towards the optimum in a single-objective case and towards the Pareto-optimal front in a multi-objective case.
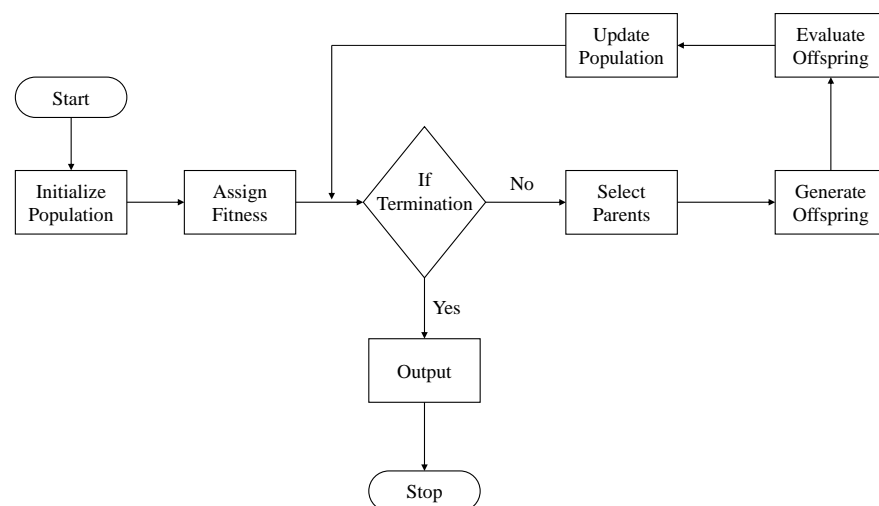

Figure 9. A flowchart for a general evolutionary algorithm.

Using the described evolutionary framework, a number of algorithms have been developed which successfully solve a variety of optimization problems. Their strength is particularly observable in handling two- to three-objective optimization problems and generating the entire Pareto-front. The aim of an EMO algorithm is to produce solutions which are (ideally) Pareto-optimal and uniformly distributed over the entire Pareto-front, so that a complete representation is provided. In the domain of EMO algorithms, these aims are commonly referred to as convergence and diversity. Figure 10 shows the working of a typical EMO algorithm that starts with a random initial population and aims to converge to the efficient frontier with a diverse set of solutions. The researchers in the EMO community have so far regarded an *a posteriori* approach to be an ideal approach where a representative set of Pareto-optimal solutions are found and then a DM is invited to select the most preferred point. The assertion is that only a DM who is well-informed is in a position to take a right decision. A common belief is that decision making should be based on complete knowledge of the available alternatives; current research in the field of EMO algorithms has taken inspiration from this belief. Though the belief is true to a certain extent, there are inherent difficulties associated with producing the entire set of alternatives and performing decision making thereafter, which many a time renders the approach ineffective.
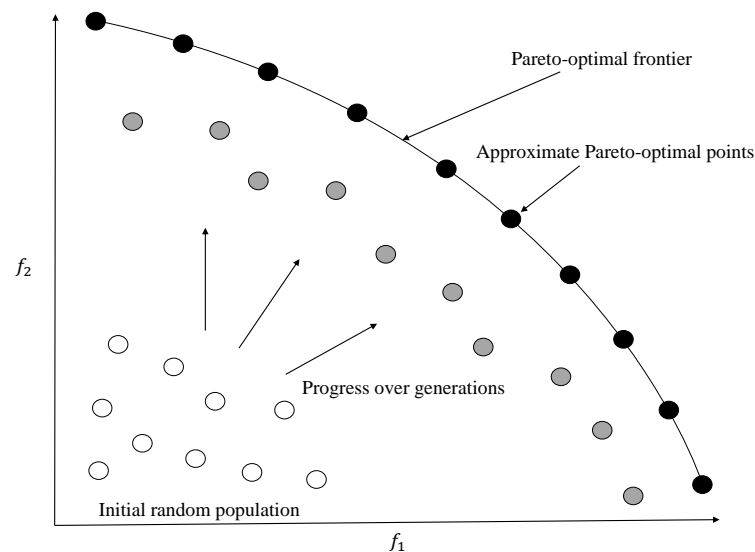
Figure 10. The working of a general evolutionary multi-objective optimization (EMO) algorithm.

The EMO approaches can be divided into three broad categories based on the idea that they use to achieve convergence and diversity. The categories are:

1. Pareto-based Methods [6,7,31,32,56]
2. Indicator-based Methods [11,57–59]
3. Decomposition-based Methods [10,12,33,60,61]

While Pareto-based approaches have been popular for solving two- or three-objective test problems, their efficiency deteriorates on problems with a higher number of objectives. Many of these methods are based on the approach of non-dominated sorting of the population as the primary driver. In problems with a large number of objectives, most of the solutions generated by these approaches are non-dominated in the comparison set leading to deterioration in progress towards the Pareto-frontier. Indicator-based approaches attempt to optimize a particular indicator that accounts for both convergence and diversity but did not become popular because of high computational costs involved in computing the indicator metric (for example, Hypervolume or Inverted Generational Distance) in many-objective problems. Despite these issues, both Pareto-based and indicator-based methods still hold promise, as non-dominated sorting in Pareto-based approaches is one of the fundamental ideas for partial ordering that cannot be ignored; similarly, faster computation of indicator metrics would make the indicator-based approaches competitive.

An alternative to Pareto-based approaches and indicator-based approaches are decomposition-based approaches, which have been effective in handling a larger number of objectives by decomposing the original problem into a set of subproblems, either multiple single-objective problems or multiple simplified multi-objective problems. These multiple problems are solved simultaneously in a collaborative manner and lead to better convergence and diversity, as the convergence is guaranteed by ensuring that each subproblem is properly optimized, and diversity is guaranteed by implicitly distributing the subproblems in an even manner. Interestingly, the decomposition-based methods utilize MCDM approaches while decomposing the multi-objective problems into subproblems. For instance, a distributed set of reference directions from the ideal point (or sometimes from the nadir point) towards the Pareto-front would lead to a well-distributed set of Pareto-optimal solutions if the front is uniform in shape. The methods that rely on decomposition solve these subproblems in a parallel manner and differ mostly on the basis of how the subproblems are created, how information between subproblems is shared during the generations, and how the subproblems may adapt during intermediate generations. However, note that if one considers a 10-objective problem, with a discretization of 10 along

each objective, one would need 10^{10} points to approximate the frontier. Moreover, even if the points are produced by a computationally efficient algorithm, the decision-making challenge still remains. If the Pareto-front is not found with sufficient discretization, the DM may expect the method to explore additional solutions. For the purpose of evaluating the EMO approaches, a large body of literature exists on test problem toolkits [62–65] and performance assessment metrics [66–70] that allow the developers to compare the performance of various algorithms.

6. Hybrid Methods

In this section, we focus on hybrid approaches that incorporate decision making within EMO. As already highlighted, the aim of the EMO algorithms is to find a diverse set of solutions close to the Pareto-optimal front, and the DM is then expected to choose the most preferred point from the objective space. However, approximating the entire Pareto-optimal front with a set of points is not always easy and may not serve the purpose, especially in the context of problems with a large number of objectives. To alleviate these problems associated with *a posteriori* EMO approaches, some EMO researchers taking cues from their MCDM counterparts have attempted an *a priori* approach, where a small set of Pareto-optimal points in the region of interest to the DM is targeted. As soon as the region of interest becomes smaller, certain problems associated with the high dimensionality of the problem in the objective space gets alleviated. Greenwood et al. [30] used an evolutionary approach to optimize a linear value function obtained from the DM through ranking of a few alternatives. In this method, the preference information is employed before optimization, and therefore this qualifies as an *a priori* method. Other studies in this direction are the cone-dominance-based EMO [71], biased-niching-based EMO [27], the light beam approach based EMO [35], and reference-point-based EMO approaches [28,29].

In [71], the authors modify the dominance principle based on interactions with the DM. For every pair of objectives, the DM specifies maximally acceptable trade-offs, i.e., what is the improvement of one unit in one objective (say f_1) worth in terms of degradation of another objective (say f_2). If the degradation is worth at most a_{12} in f_2 when f_1 improves by unity, and at most a_{21} in f_1 when f_2 improves by unity, then the dominance scheme $x \succ y$ is modified as follows with a strict inequality in at least one case:

$$(f_1(x) + a_{12}f_2(x) \leq f_1(y) + a_{12}f_2(y)) \wedge (a_{21}f_1(x) + f_2(x) \leq a_{21}f_1(y) + f_2(y))$$

Incorporating the above principle in an EMO is straightforward, as one can simply replace objectives f_1 and f_2 with Ω_1 and Ω_2, respectively, where Ω_1 and Ω_2 are defined below, and solve the problem with the standard dominance principle.

$$\Omega_1(\mathbf{x}) = f_1(\mathbf{x}) + a_{12}f_2(\mathbf{x})$$
$$\Omega_2(\mathbf{x}) = a_{21}f_1(\mathbf{x}) + f_2(\mathbf{x})$$

The approach can be incorporated in any EMO and does not lead to any increase in complexity.

Figures 11 and 12 indicate the working of the light beam approach based EMO [35] and the reference-direction-based EMO [28,29] approaches, respectively. In their study, Jaskiewicz and Branke [40] showed that it is difficult for an EMO algorithm alone to find a good spread of solutions in five- to ten-objective problems, and when solutions around the most preferred point are targeted, the hybrid approaches are able to find satisfactory solutions.
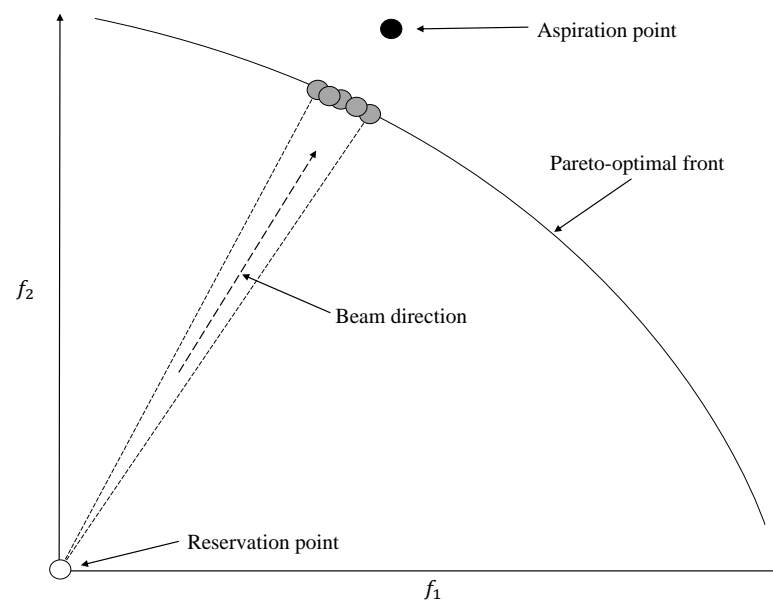
Figure 11. A light beam approach integrated within an EMO that finds a crowded set of points close to the Pareto-frontier based on the aspirations of the DM.
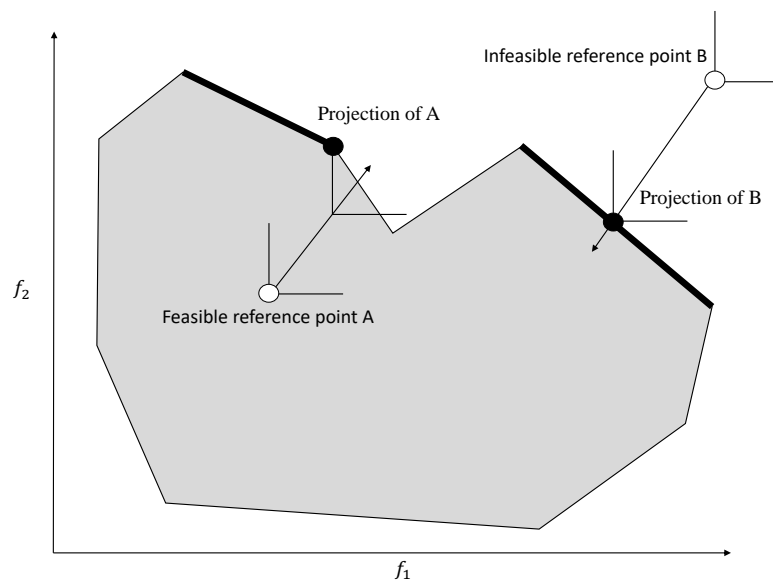

Figure 12. Projection of a feasible and infeasible reference point on the Pareto-optimal frontier within an EMO.

Preference-based EMO algorithms can differ from each other based on the following aspects:

1. Stage at which preference is incorporated;
2. Manner in which preference information is elicited;
3. Type of preference modeling performed;
4. Integration of preference model with EMO search;
5. Choice of the EMO, i.e., Pareto-based, indicator-based or decomposition-based.

Apart from a priori and a posteriori approaches, a more seamless and effective way to incorporate DMs' preferences in the EMO would be to collect and incorporate preferences at the intermediate generations of the EMO algorithm to guide the search towards the most preferred point. Such an approach is commonly referred to as a progressively interactive EMO approach. We discuss, in detail, some of the progressively interactive techniques studied in the literature.

6.1. Phelps and Köksalan (2003) [38]

Phelps and Köksalan [38] presented one of the first hybrid approaches, where they optimized a linearly weighted utility function during the iterations of an evolutionary algorithm. The decision maker makes a number of binary comparisons that leads to the weights of the utility function. For a given parameter t and ideal point $f_k^* = \text{Max}_{x \in X}\{f_k(x)\}$, the authors solve the following optimization problem to obtain the weights w_k, $k = 1, \ldots, p$.

$$
\begin{aligned}
\text{Max} \quad & \varepsilon \\
\text{s.t.} \quad & \sum_{k=1}^{p} w_k = 1 \\
& \sum_{k=1}^{p} w_k \left(f_k^* - f_k\left(\mathbf{x}^{(i)}\right) \right)^t \leq \sum_{k=1}^{p} w_k \left(f_k^* - f_k\left(\mathbf{x}^{(j)}\right) \right)^t - \varepsilon \,\, \forall \,\, \mathbf{x}^{(i)} > \mathbf{x}^{(j)} \\
& w_k \geq \varepsilon \,\, \forall \,\, k = 1, \ldots, p
\end{aligned}
$$

The above problem is an LP that leads to $\mathbf{w}^*$ used in calculating the fitness of each point using the following utility function:

$$
U(\mathbf{x}) = -\sum_{k=1}^{p} w_k^* (f_k^* - f_k(\mathbf{x}))^t
$$

The preference from the DM is taken initially or during the execution of the algorithm to modify the fitness function. The authors considered linear utility functions in their study.

To incorporate the properties of an implicit quasi-concave utility function into an EMO, Fowler et al. [39] developed an interactive EMO approach using convex preference cones. They used feasibility, dominance, and preference cones to order the population members and used that information for fitness calculation. They tested their algorithm on multi-dimensional (up to four dimensions) knapsack problems using a similar interactive genetic algorithm framework to that of Phelps and Köksalan [38]. Jaszkiewicz [40] constructed an achievement scalarizing function using random weights. The random weights are preferred if the scalarizing function generated conforms to the preference information provided by the DM. The EMO search is then guided by the scalarizing function generated with random weights.

6.2. Branke, Greco, Słowiński, and Zielniewicz (2009) [41]

Branke et al. [41] implemented the GRIP [72] methodology, in which the DM-provided pairwise information is used to find all possible compatible additive value functions (not necessarily linear). A preference-based dominance relationship and a preference-based diversity preserving operator is used in an EMO to find new solutions for the next few generations. In their approach, they make pairwise comparisons after every few generations in order to develop the preference structure. It is also possible for the DM to specify the intensities of preference. The authors use robust ordinal regression on information obtained through interaction with the DM to determine the set of all compatible value functions. Thereafter, the EMO procedure performs a parallel search for all non-dominated solutions that are preferred with respect to the compatible value functions. The authors demonstrated their procedure on a two-objective test problem. The study was later extended to solve up to five-objective test problems in [73]. The study takes a robustness approach to avoid arbitrary selection of a value function, which makes it different from most of the other studies that determine the single most discriminating value function. The use of preference information in a robust manner in EMO is a significant contribution of this study. Other recent studies that use a set of instances of the preference model compatible with the DM's preference information are [74,75]. These studies generate multiple instances of the preference model using Monte Carlo simulation and utilize the instances as search directions in a decomposition-based EMO approach.

6.3. Deb, Sinha, Korhonen, and Wallenius (2010) [42]

Based on earlier interactive MCDM approaches [76,77], this paper proposes a preference-based EMO to guide a DM to the most preferred solution by creating non-linear value functions in the intermediate generations of the algorithm. The approach accepts preference information in the form of complete or partial ranking, i.e., the DM may prefer one solution over the other or the DM may be indifferent between two solutions. The authors do not consider the situation where the DM is unable to compare two solutions. Through an extensive computational study on two- to five-objective problems, the authors evaluated the performance of their approach when the DM interacts less/more frequently with the EMO approach, as well as the impact on the quality of the solution produced when the DM provides erroneous preference information. The approach utilized the approximated value function in an innovative manner by partitioning the objective space into two areas using the value function. They also utilized the value function for performing local search and termination of the method.

In this paper, the authors fit a polynomial value function with the following structure for two objectives.

$$V(f_1, f_2) = (f_1 + k_1 f_2 + l_1)(f_2 + k_2 f_1 + l_2)$$

where f_1, f_2 are the objective functions

and k_1, k_2, l_1, l_2 are the value function parameters

For a higher number of objectives, they use a higher-order polynomial function of the following kind:

$$V(\mathbf{f}) = \prod_{i=1}^{p} \left(\sum_{j=1}^{p} \left[k_{ij} f_j + k_{i(p+1)} \right] \right)$$

where $\sum_{j=1}^{p} k_{ij} = 1$ for all i, and $k_{ij} \geq 0$ for $j \leq p$ and for all i. The value function is fitted by solving the following optimization problem with respect to the value function parameters when preference information is available:

Maximize ϵ

subject to V is non-negative at every point $\mathbf{x}^{(i)}$

           V is strictly increasing at every point $\mathbf{x}^{(i)}$

           $V(\mathbf{x}^{(i)}) - V(\mathbf{x}^{(j)}) \geq \epsilon,$ for all (i, j) pairs satisfying $\mathbf{x}^{(i)} \succ \mathbf{x}^{(j)}$

           $\left| V(\mathbf{x}^{(i)}) - V(\mathbf{x}^{(j)}) \right| \leq \delta_V,$ for all (i, j) pairs satisfying $\mathbf{x}^{(i)} \sim \mathbf{x}^{(j)}$

A look into the above optimization problem reveals that it attempts to find a value function for which the minimum difference in the value function values between the ordered pairs of points is maximum. At the same time, it also ensures that the difference in the value function values for a pair of indifferent points is smaller than a threshold that is proposed to be $\delta_V = 0.1\epsilon$. Figures 13 and 14 show how the preference structure is captured using the value function when points have a complete or a partial order, respectively. An extension of this study suggested a generalized polynomial value function [78]; however, any attempt to fit a very complex value function to user preference is not always advisable. Unless there are errors or conflicts in preference information, the preference structure in a region can often be captured using relatively simple value functions.
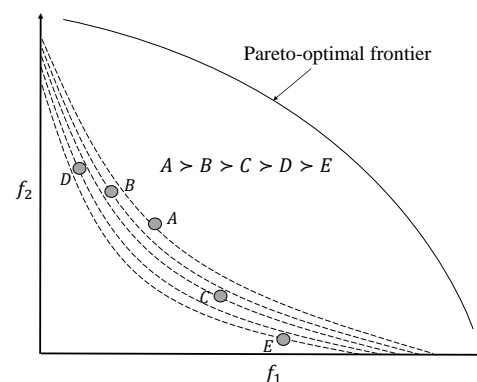
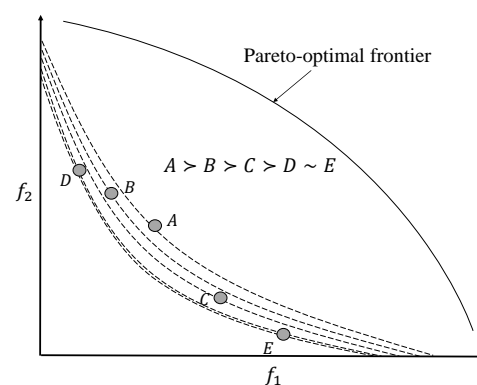**Figure 13.** Value function fitting when the points are ordered.

Figure 14. Value function fitting when the points are partially ordered.

6.4. Sinha, Deb, Korhonen, and Wallenius (2014) [43]

In this study, Sinha et al. [43] generate the most preferred solution on the Pareto-optimal frontier in a fixed budget of decision-making calls. Most of the earlier hybrid approaches did not assume that the DM will be available for providing preferences only for a fixed number of times. In fact, in most of the procedures, there is no control on the number of DM calls, or the DM calls are not utilized effectively. The assumption in most of the interactive approaches is that the DM would be available for as many interactions as desired by the method until a satisfactory solution is found; however, this is not a wise assumption to make. The approach discussed in this section attempts to address this concern by solving the problem in a fixed number of interactions with the DM. The study also deviated from constructing value functions and, instead, suggested constructing polyhedral cones heuristically to guide the EMO. The authors tested their approach on two- to five-objective test problems and studied the impact of increasing or decreasing the budget of DM calls on the performance of the algorithm in getting close to the most preferred point.

The algorithm requires the ideal point at start. Once the ideal point is known, the initial random population is created, and the point in the initial population closest to the ideal point is chosen. Let the distance be denoted as D_I. This distance D_I is divided into certain equal parts (say d_I) based on the budget of DM calls available. Thereafter, the EMO run starts, and preference from the DM is elicited only after a progress of d_I has been made. During the progress, the algorithm stores all non-dominated solutions produced in an archive set, and preference from the DM is taken in terms of the most preferred point from the archive set.

The method heuristically constructs a polyhedral cone using the most preferred solution suggested by the DM from the archive set and the end points along each of the objectives. For a p-objective problem, a polyhedral cone is formed using $p + 1$ points. Figures 15 and 16 show the construction of cones in two and three dimensions. Once the

polyhedral cone is determined, it provides an idea for a search direction. The normal unit vectors (V_i) of all the p hyperplanes can be summed up to obtain a heuristic search direction ($\vec{W} = \sum_{i=1}^{p} V_i$), which is used in the algorithm for the purpose of local search.

As an extension, a mathematically driven preference-cone-based approach was later proposed in [79], where the user's preferences were assumed to follow an unknown quasi-concave and increasing value function. In addition to considering the preference cones as a tool for eliminating non-preferred solutions, the authors also presented how the cones could be leveraged in approximating the steepest ascent direction to guide an evolutionary algorithm. A merit function is proposed that the authors use for fitness calculations in the algorithm. In addition to test problems, a mixed-integer facility location problem was solved in the later study.
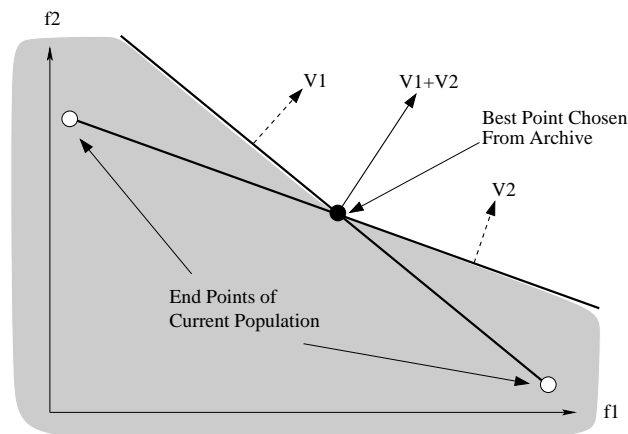

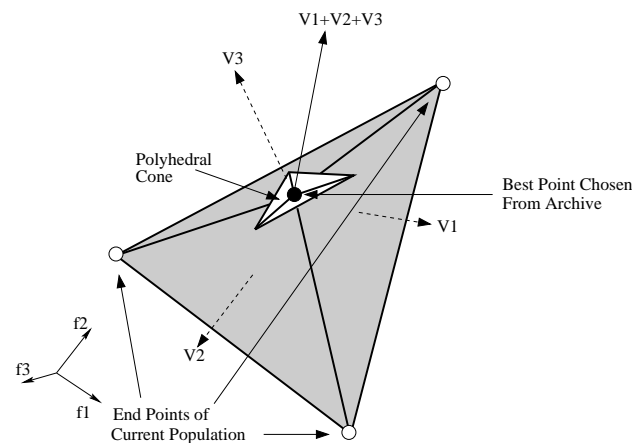**Figure 15.** Polyhedral cone in 2 dimensions.

Figure 16. Polyhedral cone in 3 dimensions.

7. Interaction Styles, Behavioral Considerations, and Future Work

Given that preferences can be elicited in a number of ways, developers of a specific method normally think that the interaction style embedded in their approach is the best. However, we need more research to answer what kind of cognitive load is caused by the interaction style (preference elicitation) and which interaction style leads closer to the true most preferred solution. An interesting approach is to use neuro-physiological measurement instruments to measure the DM's cognitive and emotional load. Scholars in the 1970s pioneered the idea of interactively solving multi-objective problems, which was remarkable considering the state of the art of computer technology in the early to mid 1970s. No personal computers, nor computer graphics capabilities, were available before the early 1980s. Scholars had to access the main frame computer via teletypes (time sharing). Their contribution was not the development of the concept of efficient or Pareto-optimal solution,

rather, it was Pareto who introduced the ideas of non-dominance. However, the scholars during the period made it possible to explore or move around the non-dominated frontier in an effective way. Problems studied were largely limited to LP framework (convex and compact feasible sets).

With computation becoming faster, newer applications arising in practice, and numerical or computational techniques for more general classes of problems being developed, researchers started looking beyond LPs. However, the decision-making difficulties did not receive the attention they deserved. For instance, there has been only limited effort in trying to solve multi-objective problems in a fixed number of interactions with the DM [43,80]. The decision-making calls are often assumed to be an unlimited resource, with the expectation that the DM is available for a large number of interactions.

Termination criterion for methods involving human–machine interaction is a challenge. Optimization methods may terminate based on gradient-based criterion, Karush–Kuhn–Tucker-based criterion, or improvement-based criterion. However, with the DM interacting with the method, it is difficult to terminate the process, as one does not know in advance the proximity of the current best solution for the DM to the true most preferred solution. Effort is also required on visualization techniques to reduce the burden on the DM during the decision-making process. Many of the visualization techniques focus on commonly used descriptive approaches, such as scatter plots, bar charts, value plots, etc. However, very few of the techniques offer an immersive experience to the DM, where the DM can easily navigate in the search space, understand trade-offs, possible improvements, and then make a decision. A comprehensive review of visualization-based approaches can be found in [81,82].

Other challenges in the decision-making context which have led to difficulties in preference modeling are as follows:

1. DM providing erroneous preferences;
2. DM providing conflicting or inconsistent preferences;
3. DM preference structure changing in different regions of the objective space;
4. DM preference structure changing as a function of learning;
5. DM becoming biased (anchored) based on the initial set of options presented;
6. DM unable to compare two options in terms of either dominance or indifference.

Significant effort is still required towards developing decision-making and search techniques that are robust to the above mentioned issues.

References

1. Charnes, A.; Cooper, W. Management models and industrial applications of linear programming. *Manag. Sci.* **1957**, *4*, 38–91. [CrossRef]
2. Keeney, R.L.; Raiffa, H. *Decisions with Multiple Objectives: Preferences and Value Tradeoffs*; Wiley: New York, NY, USA, 1976.
3. Schaffer, J.D. Multiple objective optimization with vector evaluated genetic algorithms. In *Proceedings of the First International Conference of Genetic Algorithms and Their Application*; Psychology Press: New York, NY, USA, 1985, pp. 93–100.
4. Deb, K. *Multi-Objective Optimization Using Evolutionary Algorithms*; Wiley: Chichester, UK, 2001.
5. Coello, C.A.C.; VanVeldhuizen, D.A.; Lamont, G. *Evolutionary Algorithms for Solving Multi-Objective Problems*; Kluwer: Boston, MA, USA, 2002.
6. Zitzler, E.; Laumanns, M.; Thiele, L. SPEA2: Improving the Strength Pareto Evolutionary Algorithm for Multiobjective Optimization. In Proceedings of the Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems, Athens, Greece, 19–21 September 2001; pp. 95–100.
7. Deb, K.; Agrawal, S.; Pratap, A.; Meyarivan, T. A fast and Elitist multi-objective Genetic Algorithm: NSGA-II. *IEEE Trans. Evol. Comput.* **2002**, *6*, 182–197. [CrossRef]

8.   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), Vancouver, BC, Canada, 16–21 July 2006; pp. 3352–3360.

9.   Knowles, J.; Corne, D. Quantifying the Effects of Objective Space Dimension in Evolutionary Multiobjective Optimization. In Proceedings of the Fourth International Conference on Evolutionary Multi-Criterion Optimization (EMO-2007), Matsushima, Japan, 5–8 March 2007; pp. 757–771.

10. Zhang, Q.; Li, H. MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Trans. Evol. Comput.* **2007**, *11*, 712–731. [CrossRef]

11. Bader, J.; Zitzler, E. HypE: An algorithm for fast hypervolume-based many-objective optimization. *Evol. Comput.* **2011**, *19*, 45–76. [CrossRef] [PubMed]

12. 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 Trans. Evol. Comput.* **2013**, *18*, 577–601. [CrossRef]

13. Branke, J.; Deb, K.; Miettinen, K.; Słowiński, R. *Multiobjective Optimization: Interactive and Evolutionary Approaches*; Springer: Heidelberg, Germany, 2008; Volume 5252.

14. Wang, H.; Olhofer, M.; Jin, Y. A mini-review on preference modeling and articulation in multi-objective optimization: Current status and challenges. *Complex Intell. Syst.* **2017**, *3*, 233–245. [CrossRef]

15. 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* **2018**, *6*, 41256–41279. [CrossRef]

16. Steuer, R.E. *Multiple Criteria Optimization: Theory, Computation and Application*; Wiley: New York, NY, USA, 1986.

17. Miettinen, K. *Nonlinear Multiobjective Optimization*; Springer Science & Business Media: Berlin, Germany, 2012; Volume 12.

18. Cohon, J.L. Multicriteria programming: Brief review and application. In *Design Optimization*; Gero, J.S., Ed.; Academic Press: New York, NY, USA, 1985; pp. 163–191.

19. Horn, J. Multicriterion decision making. In *Handbook of Evolutionary Computation*; Institute of Physics Publishing: Bristol, UK; Oxford University Press: New York, NY, USA, 1997; pp. F1.9:1–15.

20. Wierzbicki, A.P. The use of reference objectives in multiobjective optimization. In *Multiple Criteria Decision Making Theory and Applications*; Fandel, G.; Gal, T., Eds.; Springer: Berlin, Germany, 1980; pp. 468–486.

21. Haimes, Y.Y.; Lasdon, L.S.; Wismer, D.A. On a bicriterion formulation of the problems of integrated system identification and system optimization. *IEEE Trans. Syst. Man Cybern.* **1971**, *1*, 296–297.

22. Roy, B. The outranking approach and the foundations of ELECTRE methods. In *Readings in Multiple Criteria Decision Aid*; Springer: Heidelberg, Germany, 1990; pp. 155–183.

23. Roy, B.; Vanderpooten, D. The European school of MCDA: Emergence, basic features and current works. *J. Multi-Criteria Decis. Anal.* **1996**, *5*, 22–38. [CrossRef]

24. Greco, S.; Matarazzo, B.; Słowiński, R. Decision rule preference model. In *Wiley Encyclopedia of Operations Research and Management Science*; John Wiley & Sons, Inc.: New York, NY, USA, 2011; pp. 1–16.

25. Greco, S.; Matarazzo, B.; Słowiński, R. Rough sets theory for multicriteria decision analysis. *Eur. J. Oper. Res.* **2001**, *129*, 1–47. [CrossRef]

26. Jaszkiewicz, A.; Słowiński, R. The light beam search—Outranking based interactive procedure for multiple-objective mathematical programming. In *Advances in Multicriteria Analysis*; Springer: Heidelberg, Germany, 1995; pp. 129–146.

27. Branke, J.; Deb, K. Integrating user preferences into evolutionary multi-objective optimization. In *Knowledge Incorporation in Evolutionary Computation*; Jin, Y., Ed.; Springer: Hiedelberg, Germany, 2004; pp. 461–477.

28. Deb, K.; Sundar, J.; Uday, N.; Chaudhuri, S. Reference Point Based Multi-Objective Optimization Using Evolutionary Algorithms. *Int. J. Comput. Intell. Res.* **2006**, *2*, 273–286.

29. Thiele, L.; Miettinen, K.; Korhonen, P.; Molina, J. A preference-based interactive evolutionary algorithm for multi-objective optimization. *Evol. Comput. J.* **2009**, *17*, 411–436. [CrossRef] [PubMed]

30. Greenwood, G.W.; Hu, X.; D'Ambrosio, J.G. Fitness functions for multiple objective optimization problems: Combining preferences with pareto rankings. *Found. Genet. Algorithms* **1996**, 437–455.

31. Murata, T.; Ishibuchi, H. MOGA: Multi-objective genetic algorithms. In Proceedings of the Second IEEE International Conference on Evolutionary Computation, Perth, Western Australia, 29 November–1 December 1995; pp. 289–294.

32. Kukkonen, S.; Lampinen, J. GDE3: The third Evolution Step of Generalized Differential Evolution. In Proceedings of the 2005 Congress on Evolutionary Computation (CEC 2005), Scotland, UK, 2–5 September 2005; pp. 443–450.

33. Wang, R.; Purshouse, R.C.; Fleming, P.J. Preference-inspired co-evolutionary algorithms using weight vectors. *Eur. J. Oper. Res.* **2015**, *243*, 423–441. [CrossRef]

34. Deb, K.; Kumar, A. Interactive evolutionary multi-objective optimization and decision-making using reference direction method. In Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2007), London, UK, 7–11 July 2007; The Association of Computing Machinery (ACM): New York, NY, USA, 2007; pp. 781–788.

35. Deb, K.; Kumar, A. Light Beam Search Based Multi-objective Optimization using Evolutionary Algorithms. In Proceedings of the Congress on Evolutionary Computation (CEC-07), Singapore, Singapore, 25–28 September 2007; pp. 2125–2132.

36. Geoffrion, A.M.; Dyer, J.S.; Feinberg, A. An interactive approach for multi-criterion optimization with an application to the operation of an academic department. *Manag. Sci.* **1972**, *19*, 357–368. [CrossRef]

37. Zionts, S.; Wallenius, J. An interactive programming method for solving the multiple criteria problem. *Manag. Sci.* **1976**, *22*, 656–663. [CrossRef]
38. Phelps, S.; Koksalan, M. An interactive evolutionary metaheuristic for multiobjective combinatorial optimization. *Manag. Sci.* **2003**, *49*, 1726–1738. [CrossRef]
39. Fowler, J.W.; Gel, E.S.; Köksalan, M.; Korhonen, P.; Marquis, J.L.; Wallenius, J. Interactive evolutionary multi-objective optimization for quasi-concave preference functions. *Eur. J. Oper. Res.* **2010**, *206*, 417–425. [CrossRef]
40. Jaszkiewicz, A. Interactive multiobjective optimization with the pareto memetic algorithm. *Found. Comput. Decis. Sci.* **2007**, *32*, 15–32.
41. Branke, J.; Greco, S.; Słowiński, R.; Zielniewicz, P. Interactive evolutionary multiobjective optimization using robust ordinal regression. In Proceedings of the Fifth International Conference on Evolutionary Multi-Criterion Optimization (EMO-09), Nantes, France, 7–10 April 2009; Springer: Berlin, Germany, 2009; pp. 554–568.
42. Deb, K.; Sinha, A.; Korhonen, P.; Wallenius, J. An Interactive Evolutionary Multi-Objective Optimization Method Based on Progressively Approximated Value Functions. *IEEE Trans. Evol. Comput.* **2010**, *14*, 723–739. [CrossRef]
43. Sinha, A.; Korhonen, P.; Wallenius, J.; Deb, K. An interactive evolutionary multi-objective optimization algorithm with a limited number of decision maker calls. *Eur. J. Oper. Res.* **2014**, *233*, 674–688. [CrossRef]
44. Sinha, A.; Saxena, D.K.; Deb, K.; Tiwari, A. Using objective reduction and interactive procedure to handle many-objective optimization problems. *Appl. Soft Comput.* **2013**, *13*, 415–427. [CrossRef]
45. Benayoun, R.; de Montgolfier, J.; Tergny, J.; Laritchev, P. Linear programming with multiple objective functions: Step method (STEM). *Math. Program.* **1971**, *1*, 366–375. [CrossRef]
46. Korhonen, P.; Laakso, J. A visual interactive method for solving the multiple criteria problem. *Eur. J. Oper. Res.* **1986**, *24*, 277–287. [CrossRef]
47. Korhonen, P.; Wallenius, J. A Pareto race. *Nav. Res. Logist.* **1988**, *35*, 615–623. [CrossRef]
48. Benayoun, R.; Tergny, J. Mathematical Programming with multi-objective functions: A solution by P.O.P. (Progressive Orientation Procedure). *Revue METRA* **1970**, *9*, 279–299.
49. Frank, M.; Wolfe, P. An algorithm for quadratic programming. *Nav. Res. Logist. Q.* **1956**, *3*, 95–110. [CrossRef]
50. Korhonen, P.; Yu, G.Y. A reference direction approach to multiple objective quadratic-linear programming. *Eur. J. Oper. Reseaech* **1997**, *102*, 601–610. [CrossRef]
51. Goldberg, D.E. *Genetic Algorithms for Search, Optimization, and Machine Learning*; Addison-Wesley: Reading, MA, USA, 1989.
52. Koza, J.R. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*; MIT Press: Cambridge, MA, USA, 1992.
53. Fogel, D.B. *Evolutionary Computation*; IEEE Press: Piscataway, NY, USA, 1995.
54. Bäck, T. *Evolutionary Algorithms in Theory and Practice*; Oxford University Press: New York, NY, USA, 1996.
55. Mitchell, M. *Introduction to Genetic Algorithms*; MIT Press: Ann Arbor, MI, USA, 1996.
56. Srinivas, N.; Deb, K. Multi-Objective function optimization using non-dominated sorting genetic algorithms. *Evol. Comput. J.* **1994**, *2*, 221–248. [CrossRef]
57. Zitzler, E.; Künzli, S. Indicator-Based Selection in Multiobjective Search. In *International Conference on Parallel Problem Solving from Nature (PPSN VIII)*; Springer: Berlin/Heidelberg, Germany, 2004; Volume 3242, pp. 832–842.
58. Beume, N.; Naujoks, B.; Emmerich, M. SMS-EMOA: Multiobjective selection based on dominated hypervolume. *Eur. J. Oper. Res.* **2007**, *181*, 1653–1669. [CrossRef]
59. Sun, Y.; Yen, G.G.; Yi, Z. IGD indicator-based evolutionary algorithm for many-objective optimization problems. *IEEE Trans. Evol. Comput.* **2018**, *23*, 173–187. [CrossRef]
60. Murata, T.; Ishibuchi, H.; Gen, M. Cellular genetic local search for multi-objective optimization. In Proceedings of the 2nd Annual Conference on Genetic and Evolutionary Computation, Las Vegas, NV, USA, 10–12 July 2000; pp. 307–314.
61. Wu, M.; Li, K.; Kwong, S.; Zhou, Y.; Zhang, Q. Matching-based selection with incomplete lists for decomposition multiobjective optimization. *IEEE Trans. Evol. Comput.* **2017**, *21*, 554–568. [CrossRef]
62. Zitzler, E.; Deb, K.; Thiele, L. Comparison of multiobjective evolutionary algorithms: Empirical Results. *Evol. Comput. J.* **2000**, *8*, 125–148. [CrossRef] [PubMed]
63. Deb, K.; Thiele, L.; Laumanns, M.; Zitzler, E. Scalable multi-objective optimization test problems. In Proceedings of the Congress on Evolutionary Computation (CEC-2002), Honolulu, HI, USA, 12–17 May 2002; pp. 825–830.
64. Huband, S.; Hingston, P.; Barone, L.; While, L. A review of multiobjective test problems and a scalable test problem toolkit. *IEEE Trans. Evol. Comput.* **2006**, *10*, 477–506. [CrossRef]
65. Deb, K.; Sinha, A.; Kukkonen, S. Multi-objective Test Problems, Linkages, and Evolutionary Methodologies. In Proceedings of the 8th Annual Genetic and Evolutionary Computation Conference (GECCO 2006), Seattle, WA, USA, 8–12 July 2006; ACM Press: New York, NY, USA, 2006; pp. 1141–1148.
66. Zitzler, E.; Thiele, L. Multiobjective optimization using evolutionary algorithms – A comparative case study. In Proceedings of the Parallel Problem Solving from Nature V (PPSN-V), Amsterdam, The Netherlands, 27–30 September 1998; pp. 292–301.
67. Fonseca, V.G.D.; Fonseca, C.M.; Hall, A.O. Inferential performance assessment of stochastic optimisers and the attainment function. In *International Conference on Evolutionary Multi-Criterion Optimization*; Springer: Berlin/Heidelberg, Germany, 2001; pp. 213–225.

68. Zitzler, E.; Thiele, L.; Laumanns, M.; Fonseca, C.M.; Fonseca, V.G.d. Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Trans. Evol. Comput.* **2003**, *7*, 117–132. [CrossRef]

69. Jiang, S.; Ong, Y.; Zhang, J.; Feng, L. Consistencies and contradictions of performance metrics in multiobjective optimization. *IEEE Trans. Cybern.* **2014**, *44*, 2391–2404. [CrossRef]

70. Audet, C.; Bigeon, J.; Cartier, D.; Le Digabel, S.; Salomon, L. Performance indicators in multiobjective optimization. *Eur. J. Oper. Res.* **2021**, *292*, 397–422. [CrossRef]

71. Branke, J.; Kauβler, T.; Schmeck, H. Guidance in evolutionary multi-objective optimization. *Adv. Eng. Softw.* **2001**, *32*, 499–507. [CrossRef]

72. Figueira, J.; Greco, S.; Słowiński, R. Building a set of additive value functions representing a reference preorder and intensities of preference: GRIP method. *Eur. J. Oper. Res.* **2009**, *195*, 460–486. [CrossRef]

73. Branke, J.; Greco, S.; Słowiński, R.; Zielniewicz, P. Learning value functions in interactive evolutionary multiobjective optimization. *IEEE Trans. Evol. Comput.* **2015**, *19*, 88–102. [CrossRef]

74. Tomczyk, M.K.; Kadziński, M. Decomposition-based interactive evolutionary algorithm for multiple objective optimization. *IEEE Trans. Evol. Comput.* **2019**, *24*, 320–334. [CrossRef]

75. Tomczyk, M.K.; Kadziński, M. Decomposition-based co-evolutionary algorithm for interactive multiple objective optimization. *Inf. Sci.* **2021**, *549*, 178–199. [CrossRef]

76. Korhonen, P.; Moskowitz, H.; Wallenius, J. A progressive algorithm for modeling and solving multiple-criteria decision problems. *Oper. Res.* **1986**, *34*, 726–731. [CrossRef]

77. Korhonen, P.; Moskowitz, H.; Salminen, P.; Wallenius, J. Further developments and tests of a progressive algorithm for multiple criteria decision making. *Oper. Res.* **1993**, *41*, 1033–1045. [CrossRef]

78. Sinha, A.; Deb, K.; Korhonen, P.; Wallenius, J. Progressively Interactive Evolutionary Multi-Objective Optimization Method Using Generalized Polynomial Value Functions. In Proceedings of the 2010 IEEE Congress on Evolutionary Computation (CEC-2010), Barcelona, Spain, 18–23 July 2010; IEEE Press: Piscataway, NJ, USA, 2010; pp. 1–8.

79. Sinha, A.; Malo, P.; Kallio, M. Convex preference cone-based approach for many objective optimization problems. *Comput. Oper. Res.* **2018**, *95*, 1–11. [CrossRef]

80. Miettinen, K.; Eskelinen, P.; Ruiz, F.; Luque, M. NAUTILUS method: An interactive technique in multiobjective optimization based on the nadir point. *Eur. J. Oper. Res.* **2010**, *206*, 426–434. [CrossRef]

81. Miettinen, K. Survey of methods to visualize alternatives in multiple criteria decision making problems. *OR Spectr.* **2014**, *36*, 3–37. [CrossRef]

82. Korhonen, P.; Wallenius, J. Visualization in the multiple objective decision-making framework. In *Multiobjective Optimization*; Springer: Heidelberg, Germany, 2008; pp. 195–212.