Mariano Luque – Kaisa Miettinen – Petri Eskelinen – Francisco Ruiz

THREE DIFFERENT WAYS FOR INCORPORATING PREFERENCE INFORMATION IN INTERACTIVE REFERENCE POINT BASED METHODS

HELSINKI SCHOOL OF ECONOMICS
WORKING PAPERS
W-410

Mariano Luque* – Kaisa Miettinen** – Petri Eskelinen** – Francisco Ruiz*

THREE DIFFERENT WAYS FOR INCORPORATING PREFERENCE INFORMATION IN INTERACTIVE REFERENCE POINT BASED METHODS

* University of Málaga

**Helsinki School of Economics,
Dept. of Business Technology

December
2006

HELSINGIN KAUPPAKORKEAKOULU
HELSINKI SCHOOL OF ECONOMICS
WORKING PAPERS
W-410

HELSINGIN KAUPPAKORKEAKOULU
HELSINKI SCHOOL OF ECONOMICS
PL 1210
FI-00101 HELSINKI
FINLAND

ISSN 1235-5674
(Electronic working paper)
ISBN-10: 952-488-082-2
ISBN-13: 978-952-488-082-4

Helsinki School of Economics -
HSE Print 2006

Three Different Ways for Incorporating Preference Information in Interactive Reference Point Based Methods

Mariano Luque
University of Málaga
Calle Ejido 6, 29071 Málaga, Spain
mluque@uma.es

Kaisa Miettinen, Petri Eskelinen
Helsinki School of Economics
P.O. Box 1210, FI-00101 Helsinki, Finland
kaisa.miettinen@hse.fi, petri.eskelinen@hse.fi

Francisco Ruiz
University of Málaga
Calle Ejido 6, 29071 Málaga, Spain
rua@uma.es

Abstract

In this paper, we introduce new ways of utilizing preference information specified by the decision maker in interactive reference point based methods. A reference point consists of desirable values for each objective function. The idea is to take the desires of the decision maker into account more closely when projecting the reference point onto the set of nondominated solutions. In this way we can support the decision maker in finding the most satisfactory solutions faster. In practice, we adjust the weights in the achievement scalarizing function that projects the reference point. We identify different cases depending on the amount of additional information available and demonstrate the cases with examples. Finally, we summarize results of extensive computational tests that give evidence of the efficiency of the ideas proposed.

Keywords: multiobjective programming, multiple objectives, interactive methods, reference point methods, preferences, weights

1 Introduction

In multiobjective optimization, several objective functions are to be optimized simultaneously. Because the objective functions typically are conflicting, it is impossible to find a solution where all the objectives can reach their individual optima. Instead, we can identify compromise solutions, that is, so-called Pareto optimal or nondominated solutions, where none of the objectives can get a better value without deterioration to at least one of the other objectives. Ultimately, the task of solving multiobjective optimization problems is to find the best nondominated solutions to be called a final solution. This usually necessitates additional information from a decision maker (DM), an expert in the domain of the problem in question. Preference information coming from the DM can be expressed in many ways and in different phases of the solution process. Typically, preference information plays an important role in multiobjective optimization.

Many methods have been developed for solving multiobjective optimization problems during the years. They can be classified in four classes according to the role of the DM in the solution process (see, e.g., [4, 8]). If no preference information is available, the solution is just some neutral compromise solutions. Alternatively, the DM can specify desires and hopes before the solution process in so-called a priori methods. The drawback here is that it may be difficult for the DM to set expectations on a realistic level before getting to know the problem. On the other hand, in so-called a posteriori methods a representation of nondominated solutions is first generated and displayed to the DM who then is supposed to select the best of them as the final solution. The difficulty here is that it may be demanding for the DM to analyze many solutions. In other words, it is not clear how the solutions should be shown to the DM (when there are more than two objective functions) so that the cognitive burden set on the DM would not be too high. Besides, it may be computationally expensive for complicated real-life problems to generate many nondominated solutions. A way to overcome the above-mentioned difficulties is to use interactive methods.

Interactive multiobjective optimization methods are widely used (see, e.g., [8] and references therein). In them, a solution pattern is formed and iteratively repeated, and the DM takes actively part in the solution process by specifying and refining preference information. In this way, the DM can learn about the possibilities and limitations of the problem and about the interdependencies among the objective functions. Furthermore, only such nondominated solutions are generated that are interesting to the DM. Assuming the DM has time enough to take part in an interactive solution process, the final solution can be expected to be more satisfactory than with the other approaches because the DM can genuinely affect and direct the solution process in order to find a desired final solution. (S)he can even change one's mind while learning.

There are many interactive methods and they differ basically from each other in what kind of information is asked from and shown to the DM at each iteration, and in the way the successive solution candidates are calculated. Examples of types of preference information asked from the DM include marginal rates of substitution, surrogate values for trade-offs, classification of objective functions and reference points. For further details, see, for example, [3, 4, 8, 15] and references therein.

Among interactive approaches, methods using reference points (for the idea see, e.g., [8, 17]) have been popular (for some comparative studies, see, e.g., [10, 11]) because of their straightforward nature. A reference point consists of desirable values for each objective function. For DMs, reference points are a natural way of expressing desires in solutions because DMs do not have to learn to use new, artificial concepts. Instead, objective function values are used that as such are meaningful and understandable for DMs. Examples of methods utilizing reference points include reference point method [17], visual interactive approach [6], STOM [14], GUESS [2] and light beam search [5]. In addition, methods based on classification are closely related to reference point methods because a reference point can be formed once a classification has been made [12]. With a classification, the DM indicates what kind of changes are desirable in the current objective function values. Methods based on classification can be found, for example, in [1, 9, 10, 12, 14], among others. As an example of the close relationship between the two types of methods we can mention STOM [14], where reference points are formed based on classification and some additional information. What is common in these methods is that the DM can evaluate the problem to be solved as well as one's preferences in a flexible way.

Furthermore, relationships between reference point techniques and local tradeoffs are analyzed in [7]. There, relations among different types of information requested from the DM (e.g., reference points and local tradeoffs) are studied and such preferences are found which would produce the same solution, starting from the same previous solution. They can be regarded as equivalent pieces of information in the sense that they produce the same solution.

In this paper, we concentrate on interactive reference point based methods where, as already mentioned, the DM is at every iteration asked to specify a reference point consisting of aspiration levels, that is, desirable or acceptable values for each objective function. The next solution candidate is then generated by minimizing an achievement (scalarizing) function. In practice this means that the reference point is projected to the set of nondominated solutions and the idea is that any nondominated solution can be found by altering the reference point. In most of the methods using achievement functions, while the reference point is changed at each iteration, weights determining the projection direction are kept unaltered during the whole process and their purpose is mainly to normalize different objectives. One of the few exceptions is STOM [14], where weights are changed at each iteration using the new reference point given by the DM and the point to be projected is kept unaltered consisting of ideal objective function values. In all, in achievement functions widely used, the weights have no real preference meaning. Rather than that, they are just instrumental.

There is no doubt about the fact that widely-used interactive reference point based methods are comfortable and intuitive for DMs, and many real applications show that they perform well and, eventually, are able to find a good solution. Nevertheless, sometimes it may be difficult for the DM to find certain solutions. For example, if the DM has a greater interest in achieving a certain level for a given objective function than for the others, the only way to do it may be to provide much better values to the corresponding aspiration level. In some cases, it may even be necessary to give an aspiration level better than the ideal value for this objective, in order to push the solution towards the desired value. In these cases, the reference point may not have a clear interpretation for the DM. The use of a greater weight for this particular objective would make the process much easier. In our opinion, in general, the use of some kind of weights reflecting preferences can ease and accelerate the solution process.

The main idea of this paper is related to the fact that the reference point can be projected in many directions to become nondominated and some of the directions may be more desirable to the DM than others (especially when aspiration levels are unachievable and the reference point is far from the set of nondominated solutions). Because DMs do not usually want to use too much time in the solution process, that is, not too many iterations of the interactive method, it is important to help the DM in finding a satisfactory solution fast. With this background, our goal is to reflect the DM's preferences by means of changing the weights. Thus, we incorporate preference information into weights in the achievement function, which should result in a solution that is closer to the most preferred solution of the DM. We assume that when changing from a previous reference point to the next one, the DM has different preference intensities regarding the achievement of different aspiration levels. The main question is how to ask for this preference information without too much increasing the cognitive burden set on the DM. In this paper, we propose several schemes with an increasing amount of information coming from the DM ranging from no extra information to more detailed information.

The rest of this paper is organized as follows. In Section 2, we introduce concepts and notations used as well as some achievement functions utilizing reference points. Then we introduce several ways of taking preference information into account in the weights of achievement functions in Section 3. We illustrate our ideas with examples in Section 4 and summarize results of computational experiments in Section 5. Finally, we conclude in Section 6.

2 Concepts and Notations

We consider *multiobjective optimization problems* of the form

$$\begin{array}{ll} \text{minimize} & \{f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_k(\mathbf{x})\} \\ \text{subject to} & \mathbf{x} \in S \end{array} \tag{1}$$

involving k (≥ 2) conflicting *objective functions* $f_i : S \to \mathbf{R}$ that we want to minimize simultaneously. The *decision variables* $\mathbf{x} = (x_1, x_2, \ldots, x_n)^T$ belong to the nonempty compact *feasible region* $S \subset \mathbf{R}^n$. *Objective vectors* in *objective space* $\mathbf{R}^k$ consist of *objective values* $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_k(\mathbf{x}))^T$ and the image of the feasible region is called a *feasible objective region* $Z = \mathbf{f}(S)$.

In multiobjective optimization, objective vectors are optimal if none of their components can be improved without deteriorating at least one of the others. More precisely, a decision vector $\mathbf{x}' \in S$ is said to be *efficient* if there does not exist another $\mathbf{x} \in S$ such that $f_i(\mathbf{x}) \leq f_i(\mathbf{x}')$ for all $i = 1, \ldots, k$ and $f_j(\mathbf{x}) < f_j(\mathbf{x}')$ for at least one index j. On the other hand, a decision vector $\mathbf{x}' \in S$ is said to be *weakly efficient* for problem (1) if there does not exist another $\mathbf{x} \in S$ such that $f_i(\mathbf{x}) < f_i(\mathbf{x}')$ for all $i = 1, \ldots, k$. The corresponding objective vectors $\mathbf{f}(\mathbf{x})$ are called *(weakly) nondominated objective vectors*. Note that the set of nondominated solutions is a subset of weakly nondominated solutions.

Let us assume that for problem (1) the set of nondominated objective vectors contains more than one vector. Because it is often useful to know the ranges of objective vectors in the nondominated set, we calculate the *ideal objective vector* $\mathbf{z}^\star = (z_1^\star, z_2^\star, \ldots, z_k^\star)^T \in \mathbf{R}^k$ by minimizing each objective function individually in the feasible region, that is, $z_i^\star = \min_{\mathbf{x}\in S} f_i(\mathbf{x}) = \min_{\mathbf{x}\in E} f_i(\mathbf{x})$ for all $i = 1, \ldots, k$, where E is the set of efficient solutions. This gives lower bounds for the objectives. The upper bounds, that is, the *nadir objective vector* $\mathbf{z}^{\text{nad}} = (z_1^{\text{nad}}, z_2^{\text{nad}}, \ldots, z_k^{\text{nad}})^T$, can be defined as $z_i^{\text{nad}} = \max_{\mathbf{x}\in E} f_i(\mathbf{x})$ for all $i = 1, \ldots, k$. In practice, the nadir objective vector is usually difficult to obtain. Its components can be approximated using a pay-off table but in general this kind of an estimate is not necessarily too good (see, e.g., [8] and references therein.)

Furthermore, sometimes a *utopian objective vector* $\mathbf{z}^{\star\star} = (z_1^{\star\star}, z_2^{\star\star}, \ldots, z_k^{\star\star})^T$ is defined as a vector strictly better than the ideal objective vector. Then we set $z_i^{\star\star} = z_i^\star - \varepsilon$ for all $i = 1, \ldots, k$, where $\varepsilon > 0$ is a small real number. This vector can be considered instead of an ideal objective vector in order to avoid the case where ideal and nadir values are equal or very close to each other. In what follows, we assume that the set of nondominated objective vectors is bounded and that we have global estimates of the ranges of nondominated solutions available.

All nondominated solutions can be regarded as equally desirable in the mathematical sense and we need a *decision maker* (DM) to identify the most preferred one among them. A DM is a person who can express preference information related to the conflicting objectives and we assume that less is preferred to more in each objective for her/him. Here we assume that the DM specifies preferences in the form of reference points.

Typically, when solving multiobjective optimization problems, the multiple objective functions and preferences specified by the DM are combined in real-valued *scalarizing functions*. Scalarizing functions can be optimized with appropriate single objective optimization techniques and they generate (weakly) nondominated solutions for the original problem.

The main scheme of interactive techniques based on reference points [18] is the following. At each iteration h, the DM must provide desirable values, that is, aspiration levels q_i^h for every objective f_i $(i = 1, \ldots, k)$, and these levels constitute a reference point $\mathbf{q}^h = (q_1^h, \ldots, q_k^h)^T$ reflecting her/his hopes. Next, a scalarizing function known as an achievement (scalarizing) function is minimized in order to find a solution that best satisfies the hopes expressed. The DM can then give a new reference point and the iterative solution process continues until the DM has found the most preferred solution as the final solution and wants to stop.

An example of an achievement function is given in problem

$$\begin{array}{ll} \text{minimize} & \max_{i=1,\dots,k} \left[\mu_i^h (f_i(\mathbf{x}) - q_i^h) \right] \\ \text{subject to} & \mathbf{x} \in S, \end{array} \tag{2}$$

where μ_i^h is a weight assigned to the objective function f_i. The solution of problem (2) at iteration h is denoted by $\mathbf{x}^h$ and the corresponding objective vector by $\mathbf{f}^h = \mathbf{f}(\mathbf{x}^h)$. The solution is (weakly) efficient for any reference point (see, e.g. [8]).

Usually, in reference point based methods the reference point is changed at each iteration, while the weights are kept unaltered during the whole interactive solution process. The weights can be set for all $i = 1, \dots, k$, for example, as

$$\mu_i = \frac{1}{z_i^{\text{nad}} - z_i^{\star\star}}. \tag{3}$$

These weights normalize the values of each objective function f_i to an approximately similar magnitude with the other objectives. In what follows, we refer to weights specified in (3) as *basic weights*. Another possible normalization is to set $\mu_i = \frac{1}{|z_i^\star|}$ for all $i = 1, \dots, k$. If here some ideal objective value is equal or close to zero, we can, for example, set the corresponding weight to one. Using the latter way instead of (3) avoids the need of calculating nadir objective values, which may be difficult to approximate reliably.

One possible drawback of the achievement function in (2) is that it is generally nondifferentiable even if the functions in the original problem (1) are all differentiable. However, this drawback can be overcome if we introduce a new real-valued variable and new constraints and use an equivalent differentiable formulation

$$\begin{array}{lll} \text{minimize} & \alpha & \\ \text{subject to} & \mu_i^h (f_i(\mathbf{x}) - q_i^h) \leq \alpha & \text{for all } i = 1, \dots, k \\ & \mathbf{x} \in S, \alpha \in \mathbf{R}. & \end{array} \tag{4}$$

There are also other kinds of achievement functions used frequently in the literature. For example, problem (2) can be replaced by

$$\begin{array}{ll} \text{minimize} & \max_{i=1,\dots,k} \left[\mu_i^h (f_i(\mathbf{x}) - q_i^h) \right] + \rho \sum_{i=1}^{k} \mu_i^h (f_i(\mathbf{x}) - q_i^h) \\ \text{subject to} & \mathbf{x} \in S, \end{array} \tag{5}$$

where $\rho > 0$ is a so-called augmentation coefficient. Problem (5) produces nondominated solutions with bounded trade-offs, which often in practice are more useful than weakly nondominated solutions (see, e.g., [8, 18] for more details). It has also been shown that augmentation terms may improve computational efficiency [13].

No matter which achievement function formulation is used, the idea is the same: if the reference point is feasible, or actually to be more exact, $\mathbf{q}^h \in Z + \mathbf{R}_+^k$, then the minimization of the achievement function subject to the feasible region allocates slack between the reference point and nondominated solutions ($\mathbf{q}^h - \mathbf{f}(\mathbf{x}) \in \mathbf{R}_+^k$) producing a nondominated solution. Here $\mathbf{R}_+^k$ stands for the nonnegative orthant of $\mathbf{R}_+^k$, that is, $\mathbf{R}_+^k = \{\mathbf{q} \in \mathbf{R}^k \mid q_i \geq 0 \text{ for } i = 1, \dots, k\}$. In other words, in this case the reference point is nondominated or it is dominated by some nondominated solution. On the other hand, if the reference point is infeasible, that is, $\mathbf{q}^h \notin Z + \mathbf{R}_+^k$, then the minimization must produce a solution that minimizes the distance between $\mathbf{q}^h + \mathbf{R}_+^k$ and Z, see [8, 17]. In what follows, we say that a *reference point is feasible* if $\mathbf{q}^h \in Z + \mathbf{R}_+^k$. Otherwise, we say that it is *infeasible*. We can easily judge the feasibility of the reference point by studying the sign of the optimal achievement function value.

Let us point out that even though we in the following sections refer to formulation (2), the schemes presented do not depend on the form of the achievement function used and any other formulation could be used as well.

3 Reflecting Preference Information in Weights in Achievement Functions

When in reference point based methods the DM provides at iteration h a new reference point $\mathbf{q}^h$, (s)he may expect that there exists that kind of a nondominated objective vector, or a solution very close to it. However, the expectations of the DM may be too optimistic (or pessimistic) and the reference point given may actually be quite far from the set of nondominated objective vectors. Let us illustrate this with an example. In Figure 1, we have a reference point $\mathbf{q}^h$ and if we use the basic weights, we get the solution $\mathbf{f}^h$. However, the reference point could be projected to any nondominated solution between A and B by using different weights. The question is, which solution between A and B is the most satisfactory to the DM?

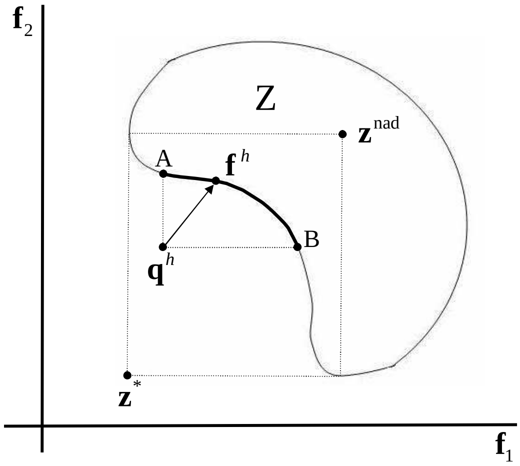

Figure 1: Different nondominated solutions for one reference point.

In this section, we propose several schemes to incorporate the DM's preference information to weights in reference point based interactive procedures. We suggest the new schemes to be used so that both the solutions calculated by minimizing the achievement function with the basic weights and with the new weights proposed are shown to the DM. This is because we do not claim that the new weights could in all possible situations give a more preferred solution than the one produced with basic weights.

Depending on which kind of extra information is available from the DM, we define three cases.

- Case 1: No extra information is available from the DM. Therefore, only information obtained from the previous iterations about the DM's choices is considered.
- Case 2: The DM is asked to give a local preference order regarding the achievement of aspiration levels specified.
- Case 3: The DM is asked to indicate preferences by allocating improvement or relaxation to current aspiration levels.

Before we concentrate in more detail in the three cases, we must say something about how to start the interactive reference point based solution process. It is possible that we ask even the first reference point $\mathbf{q}^0$ from the DM. In this case, it is typically useful to first show the ideal and the nadir objective values to her/him in order to give some understanding of what kind of solutions are feasible. Alternatively, we can calculate a so-called neutral compromise solution [19] as the first solution. This means that we set $q_i^0 = (z_i^{\mathrm{nad}} + z_i^{\star\star})/2$ as aspiration levels for $i = 1, \ldots, k$ in problem (2). The result is a good starting point when no preference information is yet available.

3.1 Case 1: No Extra Information Available from DM

If no extra information is available from the DM, the weights are set using preferences revealed so far in previous iterations. When using interactive methods, it is practical to allow the DM to save interesting solutions found during the solution process in a database (see, for example [5, 12]). In other words, the DM can save solutions that seem good candidates as a final solution although (s)he is not quite satisfied yet.

Here we assume that at least two solutions have been saved in a database and our idea is to calculate mean values $\bar{\mathbf{f}}^h = (\bar{f}_1^h, \ldots, \bar{f}_k^h)^T$ of the objective vectors in the database. We then use these mean values in the weights in order to project the reference point $\mathbf{q}^h$ in the set of nondominated solutions in the direction given by the mean values.

At each iteration the DM specifies a reference point and if (s)he finds interesting solutions, saves them in the database. (S)he can also delete solutions from the database if they become uninteresting. The weights are set for all $i = 1, \ldots, k$ as

$$\mu_i^h = \frac{1}{|q_i^h - \bar{f}_i^h|}. \tag{6}$$

After this, we can minimize the achievement function used, for example, by solving problem (2) or (4). The solution obtained should reflect the hopes of the DM. (If the denominator in (6) is too close to zero, we use only the basic weights.)

In order to calculate the mean values at iteration h, let us denote by A^h the set of indices (i.e., iteration numbers) of the solutions saved in the database by the DM at iteration $h-1$. We calculate the arithmetical mean of these solutions as

$$\bar{f}_i^h = \frac{1}{n_A^h} \sum_{j \in A^h} f_i^j \tag{7}$$

for each $i = 1, \ldots, k$, where n_A^h is the number of solutions in the database.

If the DM is able to specify additional information so that the solutions in the database are classified, for example, in classes 'very good', 'good' and 'fair', then it is possible to use an arithmetical weighted mean of the solutions. This means that we consider different weights for each class and give bigger weights for the better classes. More details of this scheme and other further ideas are given in the Appendix.

3.2 Case 2: Using Local Preference Order of Aspiration Levels

Next we assume that the DM is able to rank the relative importance of achieving each aspiration level every time a new aspiration level has been provided. It should be noted here that the DM is not asked to give any global preference ranking of the objectives, but we are interested in the local importance of achieving each of the aspiration levels. Let us point out that the DM is allowed to assign the same importance to several aspiration levels.

After the DM has specified her/his reference point, (s)he assigns objective functions to classes in an increasing order of importance for achieving corresponding aspiration level. This importance evaluation allows us to allocate the k objective functions into index sets J_r which represent the importance levels $r = 1, \ldots, s$, where $1 \leq s \leq k$. If $r < t$, then achieving the aspiration levels of objective functions in the index set J_r is less important than achieving aspiration levels of the objectives in J_t. One objective function can only belong to one index set but, as mentioned earlier, several objectives can be assigned to the same index set J_r. This means that achieving their aspiration levels is equally important. Here we have different weights depending on whether the reference point is feasible or not.

If the current reference point is infeasible, the weights for objectives f_i with $i \in J_r$ are set as

$$\mu_i^h = \frac{r}{z_i^{\text{nad}} - z_i^{\star\star}} \tag{8}$$

for each $r = 1, \ldots, s$. On the other hand, if the reference point is feasible, we set

$$\mu_i^h = \frac{1}{r(z_i^{\text{nad}} - z_i^{\star\star})}$$

for $i \in J_r$ and $r = 1, \ldots, s$. This scheme allows us to turn ordinal information into cardinal weights, which is expected to produce a solution that is closer to the DM's preferences. Note that the number of importance levels s may be different from one iteration to another.

In the formulas above, differing weights for infeasible and feasible reference points have an intuitive justification. If the reference point is infeasible, the greater the weight is and the greater is the importance assigned to achieve the aspiration level because the unachievement is being penalized more. On the other hand, if the reference point is feasible, the smaller the weight is and the more the corresponding value of the objective function can be improved, because unachievement is now desirable.

The feasibility of the current reference point $\boldsymbol{q}^h$ can be, for instance, determined by examining the sign of the achievement scalarization function (2) at the solution $\boldsymbol{x}^h$ obtained using the basic weights (3) and the reference point $\boldsymbol{q}^h$. If the sign of the achievement scalarization function is strictly negative then the reference point $\boldsymbol{q}^h$ is infeasible, otherwise it is feasible. Because we are proposing that at each iteration the basic solution is also shown to the DM no additional computation is needed to determine feasibility.

3.3 Case 3: Allowing Changes in Aspiration Levels

Finally, in the third case, we assume that more elaborated information can be obtained from the DM. Obviously, in this case, the cognitive burden set on the DM is higher, but on the other hand, the preference information is expected to be more accurate and, consequently, the solution is expected to be better. We ask the DM to specify percentages how (s)he would like the improve the current reference point once (s)he has specified it and (s)he has been told whether it is feasible or not. Naturally, we need two different kinds of questions.

Let us assume that the DM has specified a reference point $\mathbf{q}^h$. If the reference point is feasible, we ask the DM the following question: *"Your reference point can be outperformed by feasible solutions and you can tighten your aspiration levels. Assuming you have one hundred points available, how would you distribute them among the aspiration levels so that the more points you allocate, the more improvement on the corresponding objective function value you wish to achieve?"*

If the reference point is infeasible, we ask the DM the following question: *"Your reference point is unachievable and you must relax your aspiration levels. Assuming you have one hundred points available to relax the aspiration levels, how would you distribute them so that the more points you allocate, the more the corresponding aspiration level can be relaxed?"*

Let us assume that the DM has given p_i^h points to the aspiration level q_i^h related to the objective function f_i. We set $\Delta q_i^h = p_i^h/100$. Then we set the weight

$$\mu_i^h = \frac{1}{\Delta q_i^h (z_i^{\text{nad}} - z_i^{\star\star})} \tag{9}$$

for all $i = 1, \dots, k$. Then we can minimize the achievement function used and get a solution that reflects the preferences of the DM. Note that in order to avoid computational problems, we assume that each objective function is given at least one of the hundred points, that is $1 \leq p_i^h \leq 100$ for $i = 1, \dots, k$.

Let us point out that here the same formula for weights can indeed be used irrespectively of the question posed. This is because the points allocated in different questions have different meanings standing either for improvement in case of a feasible reference point or relaxation in case of an infeasible reference point.

Once the most appropriate scheme of the three cases introduced has been chosen, as mentioned in the beginning of this section, we propose to calculate the corresponding solution together with the one obtained using basic weights. In this way, the DM is able to choose the most preferred solution. Finally, let us remind that the preference schemes suggested can be combined with any achievement function, because we only modify the weights used.

4 Example

In this section, we illustrate the behaviour of the three schemes introduced in Section 3 with a nonlinear multiobjective optimization problem involving two objective functions of the form

$$\begin{array}{ll} \text{minimize} & f_1(\mathbf{x}) = -4x_1 - x_2 \\ & f_2(\mathbf{x}) = x_1 - 2x_2 \\ \text{subject to} & 2x_1 + x_2 \leq 6 \\ & x_1^2 + x_2^2 \leq 9 \\ & x_1, x_2 \geq 0. \end{array} \tag{10}$$

For this problem, we have the ideal and nadir objective vectors as $\mathbf{z}^\star = (-12, -6)^T$ and $\mathbf{z}^{\text{nad}} = (-3, 3)^T$, respectively. The corresponding feasible objective region $Z = \{(z_1, z_2)^T \in \mathbf{R}^2 \mid \frac{5}{81}z_1^2 + \frac{17}{81}z_2^2 + \frac{4}{81}z_1 z_2 \leq 9, \frac{5}{9}z_1 + \frac{2}{9}z_2 \leq 6, -\frac{2}{9}z_1 + \frac{1}{9}z_2 \leq 0, \frac{1}{9}z_2 + \frac{4}{9}z_2 \geq 0\}$ is illustrated in Figure 2.

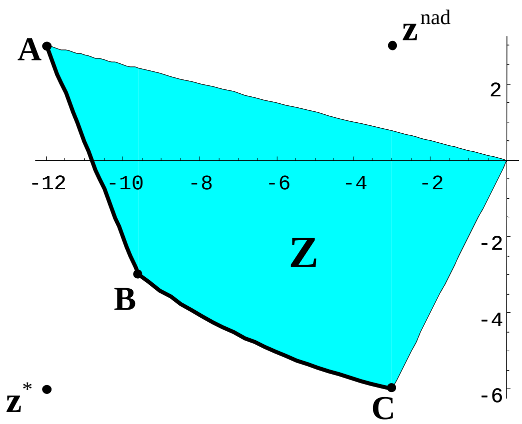

Figure 2: Feasible objective region.

As it can be seen in Figure 2, the nondominated set is the curve ABC. Let us now demonstrate how the three different cases behave in this example.

4.1 Case 1

Let us suppose that the DM has specified four reference points and obtained corresponding solution as follows: $\mathbf{q}^1 = (-11.5, -3)^T$ and $\mathbf{f}^1 = (-10.14, -1.64)^T$, $\mathbf{q}^2 = (-5.4, -5.8)^T$ and $\mathbf{f}^2 =$

$(-5.0, -5.4)^T$, $\mathbf{q}^3 = (-6.75, -5.5)^T$ and $\mathbf{f}^3 = (-6.19, -4.94)^T$ and finally $\mathbf{q}^4 = (-10.0, -5.5)^T$ and $\mathbf{f}^4 = (-8.35, -3.85)^T$. At the moment (s)he finds the first and the fourth solution the most interesting and saves them in the database.

Their arithmetic mean as defined in (7) is $\bar{\mathbf{f}} = (-9.25, -2.75)^T$. If the DM wants to improve the value of the first objective function from $\mathbf{f}^4$ and provides a new reference point $\mathbf{q}^5 = (-9.75, -5.75)^T$, after solving problem (4), we get the solution $\mathbf{f}^5 = (-9.32, -3.21)^T$. This is the projection of the reference point in the nondominated set. Let us mention that if we had used basic weights (3), we would have obtained solution $\mathbf{f}_b^5 = (-8.03, -4.03)^T$, which does not correspond to the DM's wish of improving the first objective value so well. The solution process is depicted in Figure 3.

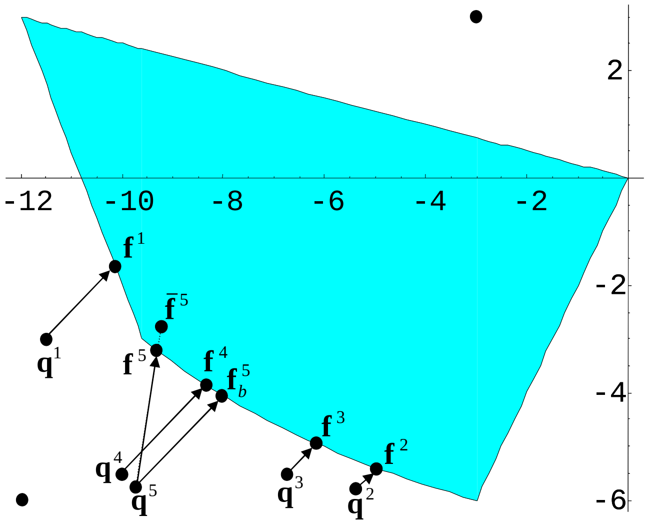

Figure 3: Case 1 with the example problem.

4.2 Case 2

Now we assume that the DM has provided an infeasible reference point $\mathbf{q}^1 = (-8.5, -5.75)^T$ and the preference order ranking 2, 1 for achieving the aspiration levels, that is, it is more important to achieve the aspiration level of the first objective function. Then the weights defined by (8) are

$$\boldsymbol{\mu}^1 = \left(\frac{2}{9}, \frac{1}{9}\right)^T.$$

When problem (4) is solved with this information, we obtain $\mathbf{f}^1 = (-7.73, -4.20)^T$. On the other hand, if we had used the basic weights, we would have obtained solution $\mathbf{f}_b^1 = (-7.22, -4.47)^T$, which has a higher, that is, worse value for f_1.

We can also demonstrate what happens if the DM specifies a feasible reference point. Let us assume that the DM sets $\mathbf{q}^2 = (-4.0, -4.0)^T$. In this case, the weights are $\boldsymbol{\mu}^2 = \left(\frac{1}{18}, \frac{1}{9}\right)^T$ and the solution obtained is $\mathbf{f}^2 = (-6.02, -5.01)^T$. If we had used basic weights, we would have obtained solution $\mathbf{f}_b^2 = (-5.29, -5.29)^T$. The solutions are depicted in Figure 4.

4.3 Case 3

Finally, let us again assume that the DM has specified a reference point $\mathbf{q}^1 = (-8.5, -5.75)^T$. Because it is infeasible, we ask the DM: *"Your reference point is unachievable and you must relax your aspiration levels. Assuming you have one hundred points available to relax the aspiration levels, how would you distribute them so that the more points you allocate, the more the corresponding aspiration level can be relaxed?"*

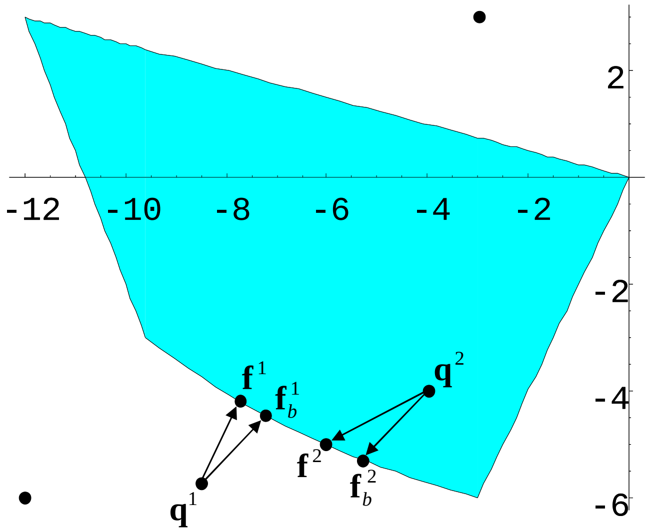

Figure 4: Case 2 with the example problem.

Let us suppose that the DM distributes the points so that the first aspiration level gets 25 points and the second gets 75 points. In other words, it is more important for the DM to achieve the aspiration level of the first objective function. Now we can get the weights (9) as

$$\boldsymbol{\mu}^1 = \left(\frac{4}{9}, \frac{4}{27}\right)^T.$$

After solving problem (4) with these weights and reference point $\mathbf{q}^1$, we get the solution $\mathbf{f}^1 = (-7.94, -4.08)^T$. On the other hand, with basic weights, we would have obtained the same solution as mentioned in Case 2, that is, $\mathbf{f}_b^1 = (-7.22, -4.47)^T$, which again has a higher value for f_1. In other words, our weights could produce a better solution.

If the DM gives a feasible reference point $\mathbf{q}^2 = (-4.0, -4.0)^T$ as in Case 2 and distributes the points for improving aspiration levels as 25 and 75 for the first and the second objective respectively, we get weights $\boldsymbol{\mu}^2 = \left(\frac{4}{9}, \frac{4}{27}\right)^T$ and the solution is $\mathbf{f}^2 = (-4.52, -5.56)^T$. With basic weights, the solution would have been $\mathbf{f}_b^2 = (-5.29, -5.29)^T$ as in Case 2.

5 Computational Tests

We have carried out several computational tests in order to compare the performances of our three weighting schemes to the solutions generated by using basic weights. With four multiobjective optimization problems we have tested all the three cases with real decision makers and, in addition, Cases 2 and 3 with three different types of value functions. In other words, in the latter type of tests, we have replaced the responses of the DM by value functions. In each test, we compared the solution of the particular case to the solution obtained with basic weights. We used two settings of tests, Test I and Test II. In the first setting, Test I, we used several single reference points and the solution minimizing the achievement scalarizing function was found using each weighting scheme. In this test setting, we assumed that our weighting schemes would produce better solutions than basic weights. In the second test setting, Test II, an interactive solution process was carried out with each weighting scheme and we assumed that the number of solutions that had to be generated before finding the most preferred solution should be smaller with our weighting schemes.

In the tests we used the four following problems:

Problem *caballeroreyruiz2:*

$$\begin{array}{ll}\text{minimize} & f_1(\boldsymbol{x}) = 50x_1^4 + 10x_2^4 \\ & f_2(\boldsymbol{x}) = 30(x_1-5)^4 + 100(x_2-3)^4 \\ & f_3(\boldsymbol{x}) = 70(x_1-2)^4 + 20(x_2-4)^4 \\ \text{subject to} & g_1(\boldsymbol{x}) = (x_1-2)^2 + (x_2-2)^2 \\ & 1 \le x_1 \le 3 \\ & 1 \le x_2 \le 3\end{array}$$

Problem *chankonghaimes:*

$$\begin{array}{ll}\text{minimize} & f_1(\boldsymbol{x}) = (x_1-1)^2 + (x_2-1)^2 \\ & f_2(\boldsymbol{x}) = (x_1-2)^2 + (x_2-3)^2 \\ & f_3(\boldsymbol{x}) = (x_1-4)^2 + (x_2-2)^2 \\ \text{subject to} & g_1(\boldsymbol{x}) = x_1 + 2x_2 - 10 \le 0 \\ & 0 \le x_1 \le 10 \\ & 0 \le x_2 \le 4\end{array}$$

Problem *peakfunctions:*

$$\begin{array}{ll}\text{minimize} & f_1(\boldsymbol{x}) = \phi(x_1, x_2) \\ & f_2(\boldsymbol{x}) = \phi(x_1-1.2, x_2-1.5) \\ & f_3(\boldsymbol{x}) = \phi(x_1+0.3, x_2-4.0) \\ & f_4(\boldsymbol{x}) = \phi(x_1-1.0, x_2+0.5) \\ & f_5(\boldsymbol{x}) = \phi(x_1-0.5, x_2-1.7) \\ \text{subject to} & -4.9 \le x_1 \le 3.2 \\ & -3.5 \le x_2 \le 6.0\end{array}$$

where

$$\phi(x_1, x_2) = -3(1-x_1)^2 e^{-x_1^2-(x_2+1)^2} + 10(\frac{1}{4}x_1 - x_1^3 - x_2^5)e^{-x_1^2-x_2^2} - \frac{1}{3}e^{-(x_1+1)^2-x_2^2}$$

Problem *peakfunctions_ mod:*

$$\begin{array}{ll}\text{minimize} & f_1(\boldsymbol{x}) = \phi(x_1, x_2) \\ & f_2(\boldsymbol{x}) = \phi(x_1-1.2, x_2-1.5) \\ \text{subject to} & -4.9 \le x_1 \le 3.2 \\ & -3.5 \le x_2 \le 6.0\end{array}$$

where $\phi(x_1, x_2)$ is like in the *peakfunctions* problem.

Problems *caballeroreyruiz2*, *chankonghaimes*, and *peakfunctions* are described in [13], and *peakfunctions_ mod* is a reduced version of the *peakfunctions* problem.

5.1 Tests with human DMs

In Test I, each DM specified several different reference points for every problem and graded the solution obtained (with achievement functions using weighting schemes in Cases 1–3 and basic weights) using a scale 1–5 reflecting how well her/his expectations were satisfied (5 indicated that (s)he was very satisfied with the solution obtained). In Test II, DMs solved each test problem four times (one for each weighting scheme) but this time using an interactive reference point method incorporating weighting schemes of Cases 1–3 or basic weights and in each test the aim was to find the most preferred solution. They graded the final solutions obtained using the scale 1–5. In addition, we recorded the number of iterations used, that is, how many reference points were needed before the final solution was found.

		Test I Grades		Test II Grades		Test II No. of iterations	
		mean	differ.	mean	differ.	mean	differ.
Basic		**2.75**		**4.13**		**6.79**	
	caballeroreyruiz2	2.65		4.00		6.86	
	chankonghaimes	2.65		4.33		7.14	
	peakfunctions	2.75		3.61		6.86	
	peakfunctions_mod	2.95		4.56		6.29	
Case 1		**3.07**	**0.32**	**4.25**	**0.13**	**4.96**	**-1.83**
	caballeroreyruiz2	3.23	0.58	4.00	0.00	4.86	-2.00
	chankonghaimes	3.28	0.63	4.50	0.17	5.29	-1.85
	peakfunctions	2.78	0.08	3.50	-0.11	4.86	-2.18
	peakfunctions_mod	3.00	0.05	5.00	0.44	4.86	-1.43
Case 2		**3.76**	**1.01**	**4.67**	**0.54**	**5.56**	**-1.23**
	caballeroreyruiz2	3.83	1.18	5.00	1.00	5.75	-1.11
	chankonghaimes	3.85	1.20	4.67	0.34	5.50	-1.64
	peakfunctions	3.60	0.85	4.00	0.39	6.13	-0.73
	peakfunctions_mod	3.75	0.80	5.00	0.44	4.88	-1.41
Case 3		**3.78**	**1.03**	**4.67**	**0.54**	**5.44**	**-1.35**
	caballeroreyruiz2	3.78	1.13	5.00	1.00	5.50	-1.36
	chankonghaimes	3.65	1.00	4.67	0.34	5.00	-2.14
	peakfunctions	3.78	1.03	4.00	0.39	6.38	-0.48
	peakfunctions_mod	3.90	0.95	5.00	0.44	4.88	-1.41

Table 1: Tests with human DMs, average values

In Table 1, we summarize average values for the grades given in Tests I and II and numbers of iterations for Test II in columns 'mean'. For the convenience of the reader, we list the differences between the average values related to basic weights and each case in the columns 'differ'. Each value marked in bold is an overall average value of mean values computed for a particular problem. For instance, overall average grade of mean grades for Case 2 in column 'Test II, Grades' is 4.67. In column 'Test II, No. of iterations, differ.' negative values indicate that when solving problem with Case 1, Case 2, or Case 3 the DMs used in average less iterations than with the basic scheme. Let us point out that the separate negative value -0.11 at row 'Case 1, peakfunctions' and column 'Test II, Grades, differ.' means that the DMs have in average experienced that in the case of peakfunction problem Case 1 produced less satisfying results when compared to the basic scheme.

5.2 Automated tests without DMs

In addition to tests carried out with human DMs, we have designed a couple of automated tests to study the behavior of weighting schemes proposed. In these tests, we simply replaced the DM with a value function (see for instance [16]). Let us point out that designing a good general automated testing procedure is not a trivial task and, therefore, we decided to use a relatively simple approach where artificial parameters are avoided as far as possible. The automated tests were carried out for Cases 2 and 3 as well as for basic weights for each test problem and each value function. Case 1 was left out from the automated tests because it is more difficult to design intuitive tests for it. The automated tests are based on Test I and Test II where the DM is replaced with one of the following three (linear, quadratic and exponential) value functions:

Case	problem	quad	exp	lin	Avg
Case2					**69%**
	caballeroreyruiz2	69%	81%	64%	71%
	chankonghaimes	75%	81%	60%	72%
	peakfunctions	55%	75%	67%	66%
	peakfunctions_mod	74%	79%	46%	66%
Case3					**73%**
	caballeroreyruiz2	79%	70%	74%	74%
	chankonghaimes	83%	71%	59%	71%
	peakfunctions	67%	87%	78%	77%
	peakfunctions_mod	72%	81%	53%	69%

Table 2: Test I using randomized points and value functions

$$
\begin{aligned}
U_{\text{quad}}(\boldsymbol{f}(\boldsymbol{x})) &= 100\left(1-\sum_{i=1}^{k}\omega_i\left(\frac{f_i-z_i^\star}{z_i^{\text{nad}}-z_i^\star}\right)^2\right)\\
U_{\text{exp}}(\boldsymbol{f}(\boldsymbol{x})) &= 100\left(k-\sum_{i=1}^{k}\exp\left(\omega_i\frac{f_i-z_i^\star}{z_i^{\text{nad}}-z_i^\star}\right)\right)\\
U_{\text{lin}}(\boldsymbol{f}(\boldsymbol{x})) &= 100\left(1-\sum_{i=1}^{k}\omega_i\frac{f_i-z_i^\star}{z_i^{\text{nad}}-z_i^\star}\right)
\end{aligned}
$$

where weights $\omega_i > 0$, $\sum_{i=1}^{k}\omega_i = 1$, were used to simulate the DM's preferences. A large weight ω_i, for some $i = 1, \ldots, k$, reflects that the objective f_i is important for the DM.

In Test I, 100 random reference points $\boldsymbol{q} = (q_1, \ldots, q_k)^T$ were generated with $q_i \in [z_i^\star, z_i^{nad}]$ for all $i = 1, \ldots, k$. Using each reference point $\boldsymbol{q}$ a solution was obtained by solving problem (2). For Case 2 and 3 the weights were set using the gradient vector of the value function at the reference point considered. In Case 2, the weights $\boldsymbol{\mu}_i$, $i = 1, \ldots, k$, were set using formula (8) where the ranking r was determined directly by the value of $\frac{\partial U(\boldsymbol{q})}{\partial f_i}$, where a large value means a more important objective. For Case 3, points p_i^h in formula (9) were obtained using an integer part of $100\frac{\partial U(\boldsymbol{q})}{\partial f_i}/\sum_{j=1}^{k}\frac{\partial U(\boldsymbol{q})}{\partial f_j}$ for every $i = 1, \ldots, k$.

The results of Test I are summarized in Table 2 where values reported indicate how many times the use of Case 2 or Case 3 produced a better solution than the basic scheme. Columns 'quad', 'exp', and 'lin' show average results for individual value functions in the case of each problem, and column 'Avg' shows an average of all results. Values marked in bold are overall averages for each particular weighting scheme. We can see that 69% of 100 randomly generated reference points in Case 2 produced a solution with a better value function value when compared to the solution obtained with the basic scheme using the same reference point. For Case 3, the corresponding average was 73%.

Algorithm 1 describes the testing procedure that was used in the automated Test II. The algorithm proceeds in such a way that on line 3 we first initialize the reference point $\boldsymbol{q} \in [\boldsymbol{z}^\star, \boldsymbol{z}^{\text{nad}}]$, set parameters $\alpha \in (0, 1)$, and a limit for maximum iterations $h_{\max}$. In our tests, these parameters was set as $\alpha = 0.5$ and $h_{\max} = 30$. The function $U : \boldsymbol{R}^k \to \boldsymbol{R}$ is a value function used. It is assumed that the value function was maximized. On line 10, the function SetWeight($\nabla U(\boldsymbol{q}^h)$) means that the weights are set using the gradient of the value function at the reference point $\boldsymbol{q}^h$ considered (this was shortly explained in the description of automated Test I). The function MinACH($\boldsymbol{q}^h, \boldsymbol{\mu}^h$) on line 11 means that the achievement scalarizing function problem (2) is solved using the reference point $\boldsymbol{q}^h$ and weights $\boldsymbol{\mu}^h$.

Figure 5 demonstrates how Algorithm 1 works. At the first iteration ($h = 1$), a Pareto optimal objective vector $\boldsymbol{f}^h$ is obtained using the initial reference point $\boldsymbol{q}^h$ by minimizing the achievement scalarizing function (2). After the first iteration, every new reference point $\boldsymbol{q}^h$ is determined from the current objective vector $\boldsymbol{f}^{h-1}$ in direction $\nabla U(\boldsymbol{f}^{h-1})$. A real number β^h

Algorithm 1 Automated testing procedure for Test II

1: $h \leftarrow 1$
2: $u_2 \leftarrow -\infty$
3: Initialize($\boldsymbol{q}^h, \alpha, h_{\max}$)
4: **repeat**
5:   **if** $h > 1$ **then**
6:     $\beta^h \leftarrow \min\limits_{i=1,\ldots,k} \{\beta_i : \beta_i = \frac{(z_i^\star - f_i)}{\frac{\partial U}{\partial f_i}(\boldsymbol{f}^h)} \geq 0\}$
7:     $\boldsymbol{q}^h \leftarrow \boldsymbol{f}^h + \alpha\beta^h \nabla U(\boldsymbol{f}^h)$
8:     $u_2 \leftarrow u_1$
9:   **end if**
10:   $\boldsymbol{\mu}^h \leftarrow$ SetWeight($\nabla U(\boldsymbol{q}^h)$)
11:   $\boldsymbol{x}^h \leftarrow$ MinACH($\boldsymbol{q}^h, \boldsymbol{\mu}^h$)
12:   $\boldsymbol{f}^h \leftarrow \boldsymbol{f}(\boldsymbol{x}^h)$
13:   $u_1 \leftarrow U(\boldsymbol{f}^h)$
14:   $h \leftarrow h + 1$
15: **until** $u_1 \leq u_2$ or $h \geq h_{\max}$

is used to scale the vector $\nabla U(\boldsymbol{f}^{h-1})$ in such a way that the vector $\boldsymbol{b}^h = \boldsymbol{f}^{h-1} + \beta \nabla U(\boldsymbol{f}^{h-1})$ is located on the hyperplane which contains the ideal objective vector $\boldsymbol{z}^\star$ and has a normal vector $\boldsymbol{e}^i \in \mathbb{R}^k$, where $e_i^i = 1$, for some $i = 1, \ldots, k$, and $e_j^i = 0$ for all $j = 1, \ldots, k$ and $j \neq i$.

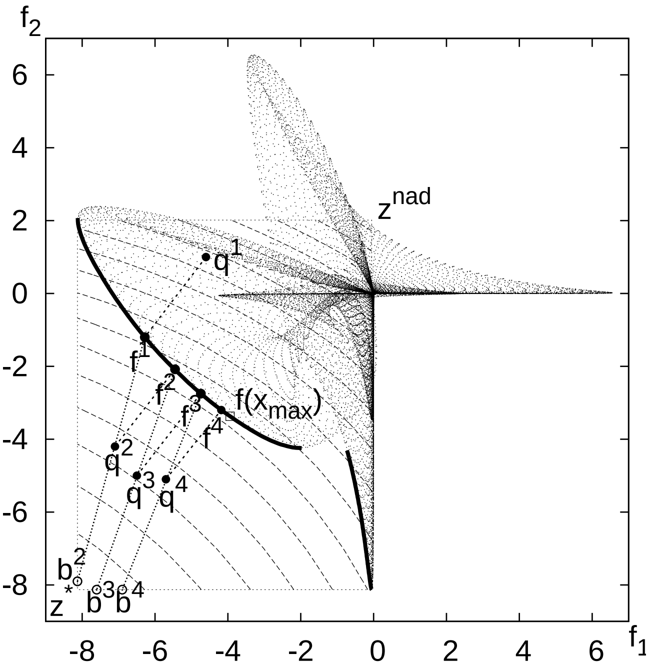

Figure 5: An iteration of automated Test II

In Algorithm, 1 the formula on 6 is derived from a standard line–plane intersection formula. The parameter α can be used to select how far each new generated reference point $\boldsymbol{q}^h$ is from the objective vector $\boldsymbol{f}^{h-1}$. The reference point $\boldsymbol{q}^h$ generated is at each iteration used to obtain the next Pareto optimal objective vector $\boldsymbol{f}^h$.

Algorithm 1 is stopped if the most recent solution has a lower value function value than the previous solution, or if the maximum number of iterations $h_{\max}$ is reached. Let us point out that if the first stopping condition in Algorithm 1 is fulfilled, the solution obtained at the previous iteration is considered as the final solution. In Figure 5, objective vector $\boldsymbol{f}(\boldsymbol{x}_{\max})$ (marked with a rectangle) indicates the maximal value of the value function considered.

Case	problem	quad	exp	lin	Avg
Case2					**60%**
	caballeroreyruiz2	80%	100%	40%	73%
	chankonghaimes	100%	60%	40%	67%
	peakfunctions	40%	80%	80%	67%
	peakfunctions_mod	0%	20%	80%	33%
Case3					**65%**
	caballeroreyruiz2	100%	100%	40%	80%
	chankonghaimes	100%	60%	40%	67%
	peakfunctions	40%	80%	60%	60%
	peakfunctions_mod	0%	100%	60%	53%

Table 3: Test II using iterative procedure and value functions

In the automated tests related to Test II, we generated five reference points for each problem. These random reference points were used as an initial reference points for Algorithm 1, and a separate test runs were carried out for each value function. Table 3 contains the summary of these test runs, and it summarizes the results in the situation where we compared solutions at the last such iteration where both the schemes (basic and either Case 2 or Case 3) produced a solution (the idea is to compare the schemes when they had used a comparable amount of computation).

The test result summary in Table 3 can be interpret like Table 2 for the automated Test I. In Case 2, for instance, 60% of all initial reference points, used in Algorithm 1, produced a final solution with a better value function value than the basic scheme. The corresponding number for Case 3 was 65%. One can say that the results of Test II are worse than those obtained in Test I. The differences in the results of the automated Test I (Table 2) and Test II (Table 3) can be partly explained with limitations of Algorithm 1, and on the other hand, with the small number of initial reference points used. If we, for instance, in Table 3 compare problem 'peakfunctions' to 'peakfunctions_mod' we notice that for the some reason the latter one seems to be more difficult for Case 2 and Case 3. However, the problem 'peakfunctions_mod' is a reduced version of 'peakfunctions' and intuitively it should be easier.

A summary of the results of automated tests is given in Table 4. In the first row, we compare the solutions obtained using weighting scheme of Case 2 to basic weights and in the second row we compare Case 3 and basic weights. The percentage values in the first two columns indicate how often Case 2 or Case 3 produced a solution with a higher value function value than when using basic weights. These values are taken directly from Tables 2 and 3. The last two columns summarize the average improvements in the optimal value function values in Case 2 or 3 when compared to basic weights. In the third column we, report the average improvement for optimal value function values for Test I and in the fourth column the same for Test II (when corresponding iterations were compared). For example, according to Table 4, in Test I when using weighting scheme Case 2 the average increase in the optimal value of the value function was 224% for those solutions which had a better value function value when compared to solutions with basic weights. We can notice that the percentage values in the last two columns of Table 4 are rather high and this is mainly due the fact that differences in the exponential value function values were quite large (even though values were normalized with the maximal values).

Finally, let us point out that the automated test framework presented in Algorithm 1 for Test II is in a general case working only a local sense and, furthermore, it is not guaranteed to converge to a point where the value function value is close to the maximal obtainable value function value. In other words, the test procedure might get stuck, for instance, in such a case where the gradient vector of the value function at some solution is parallel (or very close) to

	Percentage of better solutions		Mean percentage improvement	
	Test I	Test II	Test I	Test II
Case 2	69%	60%	224%	625%
Case 3	73%	65%	345%	287%

Table 4: Average results of automated tests

the projection direction. This might happen especially when a linear value function is used. In addition, the testing procedure presented may also get stuck when the Pareto optimal surface in the objective space is disconnected or highly nonlinear. If the testing procedure gets stuck, it does not necessarily simulate the behavior of any DM anymore. One more problem with this testing procedure is related to a proper stopping condition, that is, how the stopping condition should be set in such a way that the comparison of different weighting schemes is possible in a meaningful way.

6 Discussion

As we mentioned earlier, the new weighting schemes suggested do not always produced better solutions than basic weights. Their success depends, for example, on the problem in question and the consistency of answers given by the DM. In any case, our tests reported indicate that the new weights produce better solutions in the majority of tests. This confirms that the new weighting schemes are useful.

As far as Case 1 is concerned, we can see in Table 1 that the number of iterations used was smaller than the rest of cases (including basic weights) but the grades are worse in both Tests I and II. A probable reason for this is that the DMs felt they could not find better solutions.

We must point out that Case 1 does not necessarily perform well if the DM has saved very different kinds of objective vectors in the database. In particular, if the mean objective vector of the solutions saved in the database is not located in the set $\mathbf{q}^h + \mathbf{R}^k_+$, it may be wise not to use the weight scheme proposed. This is because the projected reference point will always be in the above-mentioned set, which may be undesirable in this case. Thus, our ideas can be considered most fruitful once the DM has passed the first learning phase of the solution process. Otherwise, the weighting used will hinder the DM from getting very different solutions from the saved ones because the weights force the solution to be close to the saved solutions. In other words, when the DM has learned to know the basic possibilities and limitations of the problem, we can expect that the reference points do not necessarily change radically from iteration to iteration but they rather converge toward the final solution and we can rely more on the solutions saved in the database. This justifies our assumption that the DM updates the set of solutions during the solution process and deletes solutions that become uninteresting during the search.

In any case, we have suggested our weighting scheme to be used together with basic weights so that the DM always gets to see both the solutions and has the possibility to select the most preferred one to continue with. The joint use of the two weighting schemes can help the DM to identify solutions that are not as interesting as (s)he may have thought.

Based on our experiments we can say that the weighting scheme in Case 2 seems to be the most intuitive and least demanding approach for the DM. The results support our claim that it outperforms the basic weight scheme in a majority of cases. However, the conversion used from an ordinal scale into a cardinal scale is rather rough. This means that with more objectives the ratio between different weights is not equal. It would naturally be possible to formulate weights with an equal ratio for all objectives, for example, for each $i \in J_r$ as

$$\mu_i^h = \frac{p^{r-1}}{z_i^{\text{nad}} - z_i^{\star\star}}$$

with $p > 1$ being the constant proportion. However, we want to keep our weighting schemes as simple as possible and, thus, do not use this formulation.

According to our experiments, the questions posed in Case 3 allow the DM to give more accurate information about her/his preferences and thus, it overcomes the drawbacks previously described for Case 2. But, on the other hand, this information needs more cognitive effort to be provided and, thus, the DM can make more inconsistencies than in Case 2. This idea is supported by the fact that the results obtained for Case 3 in the automatic tests are better than those of Case 2, but this is not the case with tests involving human DMs, specially if we take into account the number of iterations taken. This means that DMs may find it more difficult to iterate in Case 3. After the tests, we asked the human DMs to judge the relation involving effort needed versus results obtained for each Case and for basic weights. The fact that the DMs gave a higher rating to Case 2 than for others, supports the above-mentioned idea.

Let us point out that when we state in Case 3 that the reference point is not feasible, we mean that all of its components cannot be achieved (or outperformed) simultaneously. But still one or more components could be achievable. When asking the DM to relax the reference point, we do not mean that all the components should actually be relaxed to obtain a solution. Rather than that, the question is related to the relaxation amounts given by the DM as a measure of importance in achieving each aspiration level.

7 Conclusions

We have suggested several new ways of taking preference information coming from the DM more closely into account in interactive reference point based methods developed for multiobjective optimization. Our goal is to be able to produce solutions that are more satisfactory to the DM than the ones produced with standard approaches. In this way, the DM can find the final solution with less iterations.

We have proposed three different weighting schemes depending on how much information we have available from the DM. The treatment differs depending on whether the reference point specified by the DM is feasible or not. We have also tested our schemes numerically and the results support the usability of our ideas. Nevertheless, as it has also been reported, the basic weighting scheme may produce better results in some cases. This is why we recommend to use our weighting schemes together with the basic one in each case, and to let the DM choose the best solution.

Acknowledgements

This research was partly supported by Tekes, the Finnish Funding Agency for Technology and Innovation and by the Andalusian Regional Ministry of Innovation, Science and Enterprises (PAI group SEJ-445). Research of F. Ruiz has also been supported by the Spanish Ministry of Education and Science (mobility program, PR2005-0212).

Appendix

In Section 3, we mentioned in Case 1 some variants for the weighting scheme proposed. Because we assume in Case 1 that no extra preference information is available from the DM, such information must be inferred from the DM's reactions during the previous iterations of the solution process. Obviously, the DM's preferences can be misinterpreted when very little information is available. Therefore, it may be useful to have some other schemes available. In what follows, we suggest other possible approaches for Case 1.

Case 1.A

Let us suppose that the DM is able to classify the solution in the database into three classes, like 'very good', 'good' and 'fair'. Let us denote the indices of the solutions assigned to these classes by I_1^h, I_2^h and I_3^h, respectively. Furthermore, the numbers of solutions in each class are denoted by n_1^h, n_2^h and n_3^h, respectively. Now an arithmetical weighted mean can be calculated for each $i = 1, \ldots, k$ as

$$\bar{f}_i^h = \frac{1}{C} \sum_{l=1}^{3} \sum_{j \in I_l^h} (4 - l) f_i^j,$$

where C is a normalizing constant, defined by

$$C = 3n_1^h + 2n_2^h + n_3^h.$$

If the DM provides different preference grades of the solutions saved in the database, by using Case 1.A, we can reflect the DM's preferences better in the calculated mean. Therefore, the importance grade is calculated taking into account the dispersion of the objective vectors of the database.

Case 1.B

Depending on the nature of the problem considered, two reference points located relatively close to each other can produce significantly different solutions. This may be useful at initial stages of the solution process, when the DM is exploring the nondominated set. But it may be undesirable in later iterations when the DM wants to concentrate on some part of the nondominated set. In this case, if aspiration levels of two consecutive iterations for a given objective function are very close to each other, we can interpret that the DM is satisfied with that value of the corresponding objective function. In order to take this possibility into account, we can augment the scheme introduced in Case 1 by introducing a new constraint to problem (2) at the current iteration h.

Let us suppose that the aspiration level q_i^h is close enough to the aspiration level q_i^{h-1} of the previous iteration. We can measure closeness as

$$100 \left(\frac{|q_i^h - q_i^{h-1}|}{z_i^{\text{nad}} - z_i^{\star\star}} \right) \leq P_{min}, \tag{11}$$

where P_{min} is a minimum allowed percentage difference in aspiration levels (for example, $P_{min} = 5\%$ or $P_{min} = 1\%$). Let S^h be the set of indices of objective functions for which their aspiration levels satisfy (11). Then new constraints can be added to problem (1) for such objectives. To be more precise, the problem to be solved is

$$\begin{array}{ll} \text{minimize} & \max\limits_{i=1,\ldots,k} \left[\mu_i^h (f_i(\mathbf{x}) - q_i^h) \right] \\ \text{subject to} & f_i(\mathbf{x}) \leq f_i^{h-1} + \delta_i^h \quad \text{for all } i \in S^h \\ & \mathbf{x} \in S, \end{array} \tag{12}$$

where δ_i^h is a given tolerance, for example, $\delta_i^h = |q_i^h - q_i^{h-1}|$ for all $i \in S^h$. Note that the new constraints are valid for one iteration only.

Because the new constraints can be very restrictive in some cases, we propose to calculate both the solution $\mathbf{f}_c^h$ of (12) (where the weights are specified as in Case 1) and the solution $\mathbf{f}^h$ obtained without adding the new constraints, that is, by solving problem (2) with the weights defined in Case 1. Then both the solutions, together with the basic one, are shown to the DM who can choose the one (s)he prefers.

Let us point out that problem (12) always has feasible solutions because $\mathbf{f}^{h-1}$ is a feasible objective vector. A graphical representation of Case 1.B is given in Figure 6, where q_2^h is very close to q_2^{h-1}.

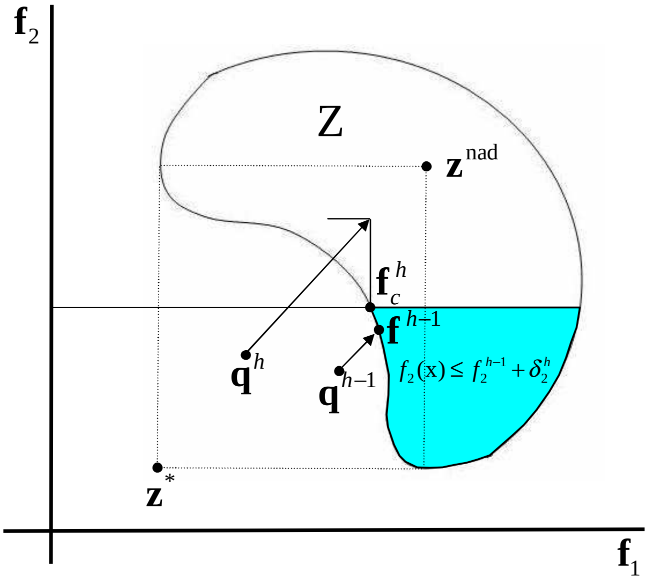

Figure 6: Example where aspiration levels of consecutive iterations are close to each other.

References

[1] R. Benayoun, J. de Montgolfier, J. Tergny, O. Laritchev, Linear Programming with Multiple Objective Functions: Step Method (STEM), Mathematical Programming 1 (3) (1971) 366–375.

[2] J.T. Buchanan, A Naïve Approach for Solving MCDM Problems: The GUESS Method, Journal of the Operational Research Society, 48 (1997) 202–206.

[3] V. Chankong, Y.Y. Haimes, Multiobjective Decision Making Theory and Methodology, Elsevier Science Publishing Co., Inc., New York, 1983.

[4] C.-L. Hwang, A.S.M. Masud, Multiple Objective Decision Making – Methods and Applications, Springer-Verlag, Berlin, Heidelberg, 1979.

[5] A. Jaszkiewicz, R. Slowiński, The 'Light Beam Search' Approach – An Overview of Methodology and Applications, European Journal of Operational Research 113 (1999) 300–314.

[6] P. Korhonen, J. Laakso, A Visual Interactive Method for Solving the Multiple Criteria Problem, European Journal of Operational Research 24 (1986) 277–287.

[7] M. Luque, R. Caballero, J. Molina, F. Ruiz, Equivalent Information for Multiobjective Interactive Procedures, Management Science (to appear).

[8] K. Miettinen, Nonlinear Multiobjective Optimization, Kluwer Academic Publishers, Boston, 1999.

[9] K. Miettinen, M.M. Mäkelä, Interactive Bundle-Based Method for Nondifferentiable Multiobjective Optimization: NIMBUS, Optimization 34 (3) (1995) 231–246.

[10] K. Miettinen, M.M. Mäkelä, Comparative Evaluation of Some Interactive Reference Point-based Methods for Multi-Objective Optimisation, Journal of the Operational Research Society 50 (1999) 949–959.

[11] K. Miettinen, M.M. Mäkelä, On Scalarizing Functions in Multiobjective Optimization, OR Spectrum 24 (2002) 193–213.

[12] K. Miettinen, M.M. Mäkelä, Synchronous Approach in Interactive Multiobjective Optimization, European Journal of Operational Research 170 (2006) 909–922.

[13] K. Miettinen, M.M. Mäkelä, K. Kaario, Experiments with Classification-Based Scalarizing Functions in Interactive Multiobjective Optimization, European Journal of Operational Research (to appear).

[14] H. Nakayama, Y. Sawaragi, Satisficing Trade-Off Method for Multiobjective Programming, in: M. Grauer, A.P. Wierzbicki (Eds.), Interactive Decision Analysis, Springer Verlag, Berlin, 1984, pp. 113–122.

[15] Y. Sawaragi, H. Nakayama, T. Tanino, Theory of Multiobjective Optimization, Academic Press, Inc., Orlando, Florida, 1985.

[16] M. Sun, Some Issues in Measuring and Reporting Solution Quality of Interactive Multiple Objective Programming Procedures, European Journal of Operational Research 162 (2) (2005) 468–483.

[17] A.P. Wierzbicki, A Mathematical Basis for Satisficing Decision Making, Mathematical Modelling 3 (1982) 391–405.

[18] A.P. Wierzbicki, On the Completeness and Constructiveness of Parametric Characterizations to Vector Optimization Problems, OR Spectrum 8, (1986) 73–87.

[19] A.P. Wierzbicki, Reference Point Approaches, in: T. Gal, T.J. Stewart, T. Hanne (Eds.), Multicriteria Decision Making: Advances in MCDM Models, Algorithms, Theory, and Applications, Kluwer Academic Publishers, Boston, 1999, pp. 9-1–9-39.