Petri Eskelinen – Kaisa Miettinen – Kathrin Klamroth – Jussi Hakanen

INTERACTIVE LEARNING-ORIENTED DECISION SUPPORT TOOL FOR NONLINEAR MULTIOBJECTIVE OPTIMIZATION: PARETO NAVIGATOR

HELSINKI SCHOOL OF ECONOMICS
WORKING PAPERS
W-439

Petri Eskelinen* – Kaisa Miettinen* – Kathrin Klamroth** – Jussi Hakanen***

INTERACTIVE LEARNING-ORIENTED DECISION SUPPORT TOOL FOR NONLINEAR MULTIOBJECTIVE OPTIMIZATION: PARETO NAVIGATOR

*Helsinki School of Economics,
Dept. of Business Technology

**University of Erlangen-Nuremberg,
Institute of Applied Mathematics

*** University of Jyväskylä,
Dept. of Mathematical InformationTechnology

December
2007

HELSINGIN KAUPPAKORKEAKOULU
HELSINKI SCHOOL OF ECONOMICS
WORKING PAPERS
W-439

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

ISSN 1235-5674
(Electronic working paper)
ISBN 978-952-488-200-2

Helsinki School of Economics -
HSE Print 2007

Interactive Learning-oriented Decision Support Tool for Nonlinear Multiobjective Optimization: Pareto Navigator

Petri Eskelinen[†,*], Kaisa Miettinen[†], Kathrin Klamroth[§], Jussi Hakanen[⋆]

[†]Helsinki School of Economics
P.O. Box 1210, FI-00101 Helsinki, Finland
e-mail: petri.eskelinen@hse.fi, kaisa.miettinen@hse.fi

[§] Institute of Applied Mathematics, University of Erlangen-Nuremberg
Martensstr. 3, D-91058 Erlangen, Germany
e-mail: klamroth@am.uni-erlangen.de

[⋆] Dept. of Mathematical Information Technology
P.O. Box 35 (Agora), FI-40014 University of Jyväskylä, Finland
e-mail: jussi.hakanen@jyu.fi

Abstract

We describe a new interactive learning-oriented method called Pareto navigator for nonlinear multiobjective optimization. In the method, first a polyhedral approximation of the Pareto optimal set is formed in the objective function space using a relatively small set of Pareto optimal solutions representing the Pareto optimal set. Then the decision maker can navigate around the polyhedral approximation and direct the search for promising regions where the most preferred solution could be located. In this way, the decision maker can learn about the interdependencies between the conflicting objectives and possibly adjust one's preferences. Once an interesting region has been identified, the polyhedral approximation can be made more accurate in that region or the decision maker can ask for the closest counterpart in the actual Pareto optimal set. If desired, (s)he can continue with another interactive method from the solution obtained. Pareto navigator can be seen as a

[*]Corresponding author

nonlinear extension of the linear Pareto race method. Pareto navigator is computationally efficient because most of the computations are performed in the polyhedral approximation and for that reason function evaluations of the actual objective functions are not needed. Thus, the method is well suited especially for problems with computationally costly functions. Furthermore, thanks to the visualization technique used, the method is applicable also for problems with three or more objective functions, and in fact it is best suited for such problems. We illustrate the method and the underlying ideas with an example.

Keywords: multicriteria optimization, MCDM, interactive methods, decision support, Pareto optimality

1 Introduction

Solving multiobjective optimization problems can be understood as finding the most preferred trade-off between conflicting objectives. In the field of multiple criteria decision making (MCDM), the idea is to help a decision maker in finding the best solution among mathematically incomparable compromises, so-called Pareto optimal solutions. During the years, many methods have been developed for this purpose (see, e.g. [1, 4, 12, 19, 20]). However, their real-life applications are still surprisingly few [6]. One possible explanation could be that the tools developed for decision support may not be illustrative and easy-to-use enough for real decision makers.

Multiobjective optimization methods can be classified, for example, according to the role of the decision maker in the solution process [12]. Among the plethora of multiobjective optimization methods available, interactive methods are regarded promising because they allow an active participation of the decision maker in the solution process. In this way, the decision maker can direct the search and concentrate on solutions that are most interesting to her/him. For example, according to [21], two different conceptions regarding interactive methods can be identified. In searching-oriented methods, a converging sequence of solution proposals is presented to the decision maker. On the other hand, in learning-oriented methods, a free exploration of solutions is possible allowing trial and error. As mentioned in [12], the best way would be to combine these approaches. As a matter of fact, in many decision processes, one can identify two phases: a *learning phase* and a *decision phase*. No matter which style of expressing preference information is used (e.g., desirability of trade-offs, reference points, classification of objective functions etc.), it is often valuable for the decision maker first to be able to learn about the possibilities and limitations of the problem in order to adjust one's hopes on a realistic level and then fine-tune the final solution. It is also important to use concepts the decision maker understands well.

An attempt of developing an intuitive and understandable method for linear multiobjective optimization problems was suggested in [10] as a so-called Pareto race. There, the idea is that the decision maker can navigate in the set of Pareto optimal solutions like driving a car, in other words, move around in the Pareto optimal set according to his/her desires in order to identify the most preferred trade-off. Thanks to parametric linear programming, Pareto race can show changes in objective function values in real-time. This is a very appealing learning-oriented approach but, as said, it works only for linear problems.

In this paper, we concentrate on (convex) nonlinear problems and introduce a method that enables convenient and real-time navigation in the approximated Pareto optimal set of a nonlinear multiobjective optimization problem. We call this learning-oriented method by the name `Pareto navigator`. One can say that we extend and generalize the navigation ideas used in the Pareto race to nonlinear problems. At the same time, we enable the decision maker to direct the search for the most preferred solution in more diverse ways than in Pareto race.

The starting point of our method is a relatively small discrete representation of the set of Pareto optimal solutions. Using these solutions, we generate a polyhedral approximation of the Pareto optimal set in the objective function space. In this approximation, the decision maker can navigate according to his/her wishes and search for the most preferred trade-off. Because of the structure of the approximation, computation is fast and movements can be shown in real-time. Thus, our approach is particularly useful for problems where function evaluations are costly and time-consuming. Instead of response surface or kriging type of approaches (see, e.g., [9]), we directly approximate the Pareto optimal set and not the objective functions involved. Once the decision maker has identified an interesting region or solution, (s)he can ask for a more accurate approximation of the Pareto optimal set (i.e., more points in the approximation or concentrate the approximation in the desired region [7]) or see (in some sense) the closest Pareto optimal solution in the original problem. Then one can continue navigating or continue with another interactive method after having learned about the feasible trade-offs.

The strengths of our approach include the applicability to computationally costly problems as well as a very intuitive user interface. The decision maker can see global trade-off information between different conflicting objectives in real-time and conveniently control in which direction to move, that is, which objectives to improve or which objectives to allow getting worse. This enables the decision maker to learn about the interdependencies among the objectives in the problem and set one's expectations on a realistic level. Even though we are dealing with an approximation, the decision maker can anytime get to see (in some sense) the closest Pareto solution of the original problem (by projecting the solution identified). We demonstrate our approach and a graphical user-interface of a software implementation with an example.

The rest of this paper is organized as follows. First we introduce some notations and concepts used in Section 2. In Section 3, the algorithm of Pareto navigator is presented. Section 4 is devoted to implementation issues including a user interface and an example that demonstrates how Pareto navigator can be used. In Section 5, we shortly discuss some issues related to the development of the method and future research. We finish with some concluding remarks in Section 6.

2 Some notations and concepts

In this paper, we deal with convex multiobjective optimization problems of the form

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

where $\mathbf{x} \in \mathbb{R}^n$ is called a *decision (variable) vector* and it should belong to the *feasible region* S. We assume that all the *objective functions* $f_i : \mathbb{R}^n \to \mathbb{R}$, for each $i = 1, \dots, k$, and S are convex. The image of a feasible decision vector $\mathbf{x} \in S$ under mapping $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^k$ from the *decision variable space* $\mathbb{R}^n$ to the *objective space* $\mathbb{R}^k$ is called a *feasible objective vector* and denoted by $\mathbf{z} = \mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), \dots, f_k(\mathbf{x}))^T$. The components of objective vectors are called *objective values*.

In this paper, we define optimality using the concept of Pareto optimality.

Definition 2.1
A decision vector $\mathbf{x}^* \in S$ and an objective vector $\mathbf{f}(\mathbf{x}^*)$ are said to be Pareto optimal *if there does not exist another decision vector $\mathbf{x} \in S$ such that $f_i(\mathbf{x}) \leq f_i(\mathbf{x}^*)$ for all $i = 1, \dots, k$ and $f_j(\mathbf{x}) < f_j(\mathbf{x}^*)$ for some j.*

Note that the Pareto optimal solutions are in a mathematical sense incomparable without additional information. Furthermore, problem (1) typically has infinitely many Pareto optimal solutions. The set of all the Pareto optimal objective vectors is called the *Pareto optimal set*. Later on, this will also be referred to as an *actual Pareto optimal set*.

Usually in the MCDM field, the aim of solving a multiobjective optimization problem is to find a single feasible decision vector which is considered as the final solution for problem (1). However, we need some external information to decide which of the Pareto optimal solutions is the *most preferred* one. A *decision maker (DM)* is a person who has knowledge about the problem in question and can express preference information related to Pareto optimal solutions. Naturally, what is to be regarded as most preferred depends on the DM involved.

It is often useful for the DM to know lower and upper bounds for the values of objective functions appearing in the Pareto optimal set. The *ideal objective vector* $\mathbf{z}^\star \in \mathbb{R}^k$ consists of the optimal values $z_i^\star$ for each objective function f_i with respect to the feasible region S. In other words, these are the best values that the individual objective functions can attain in the Pareto optimal set. If the objective functions are conflicting, which usually is the case, the ideal objective vector is infeasible. The *nadir objective vector* $\mathbf{z}^{nad}$ gives upper bounds for the values of the individual objective functions in the Pareto optimal set. However, because the set of Pareto optimal decision vectors is unknown beforehand, we usually need to use an approximated nadir objective vector (in the case of more than two objectives). Often it is approximated using a *payoff table* (see [12] and references therein) but other ways also exist [3].

As motivated in the introduction, we consider here interactive approaches and, in particular, learning-oriented methods. The purpose of interactive methods is to aid the DM in finding a Pareto optimal solution which corresponds to the preferences of the particular

DM in the best possible way. In interactive methods, a solution pattern is formed and iteratively repeated allowing the DM to adjust one's preferences and concentrate on solutions (s)he finds interesting. This means that the DM is directing the search according to her/his desires.

In interactive multiobjective optimization, there are several possibilities for the DM to express preference information [12]. For example, the DM can indicate desired changes in the objective function values of the current Pareto optimal solution by specifying a *classification*. Widely used classes are the following: Objective functions whose values should be improved, are satisfactory or are allowed to impair. Note that if some objective function value is improved, then some other one must be allowed to impair in order to get another Pareto optimal solution. Another possibility to express preferences is to use reference points. A *reference point* $\bar{\mathbf{z}} \in \mathbb{R}^k$ consists of aspiration levels $\bar{z}_i$, $i = 1, \ldots, k$, that represent desired values for the objective functions. Note that the reference point does not need to be feasible. A reference point can also be extracted from the classification information [15, 16]. In that case, the components of the reference point $\bar{\mathbf{z}}$ are the ideal value $z_i^\star$, the current value and the nadir value z_i^{nad} for the classes described above, respectively.

3 Pareto navigator

We propose a new interactive learning-oriented method, Pareto navigator, for multiobjective optimization. We use a polyhedral approximation of the Pareto optimal set in order to enable the DM to explore the Pareto optimal set. With Pareto navigator, the DM can conveniently study trade-offs between the conflicting objectives in real-time and locate regions that are interesting to her/him. Therefore, we have a good learning tool to study the behavior of the problem. Because the exploration takes place in the approximated instead of the actual Pareto optimal set, it can be done with low cost even for computationally expensive problems. Thus, our approach is particularly useful in complex real-world problems where function evaluations, for instance, come from some simulation tool and may necessitate solving partial differential equations.

The starting point of our approach is a relatively small set of Pareto optimal solutions that is used to represent the actual Pareto optimal set. The polyhedral approximation generated by these solutions in the objective space will be referred to as an *approximated Pareto optimal set*. Solutions in this set will be called *approximated Pareto optimal solutions*. Using the approximated Pareto optimal set enables studying changes in objective function values in real-time when moving from one solution to another. This is realized using parametric programming. Once the DM has located an interesting solution or area from the approximation, (s)he is shown the *corresponding solution* in the actual Pareto optimal set. By a corresponding solution we here mean an actual Pareto optimal solution that is in some sense closest to the approximated solution. If desired, one can continue, for instance, with some other multiobjective optimization method which allows more detailed comparison of trade-offs between the Pareto optimal solutions (see, e.g., [12]). If the approximated Pareto optimal solution and the corresponding actual Pareto optimal solution are far from each other, the DM can also ask for a more accurate approximation of the Pareto optimal set to be generated (e.g., by increasing the number of solutions that the polyhedral approximation is based on). It is also possible to improve the approximation locally, as suggested in [7], and study the polyhedral approximation related to a subset of the original Pareto optimal set. Our simplified setting allows us to decrease cognitive burden set on to the DM while (s)he is making an overall evaluation about what kind of solutions may be available and which of them are interesting.

Usually, in interactive multiobjective optimization methods, the interdependencies between objectives are analyzed in a rather local sense, that is, mostly pointwise (e.g., trade-off rates at some solution). Instead, in Pareto navigator, the idea is to concentrate on capturing a global understanding of the possibilities and limitations in the problem considered. In other words, we provide means to study the overall behavior of Pareto optimal solutions. This is an essential part of the learning phase, discussed in the introduction. If a separate decision phase is needed after using Pareto navigator, one can switch to some other interactive method which typically are designed to support the decision phase.

The Pareto navigator method consists of two phases: *initialization* and *navigation*. The initialization phase is purely technical where we first produce a (relatively small) representative set of Pareto optimal solutions of problem (1). Using these solutions we generate a polyhedral approximation for the Pareto optimal set in the objective space. After the initialization phase is completed we are ready to start the navigation phase where the DM is in charge. In Figure 1, the algorithm of the Pareto navigator method is presented from the DM's perspective using steps $1-6$. Next, we describe more detailed actions related to these steps.

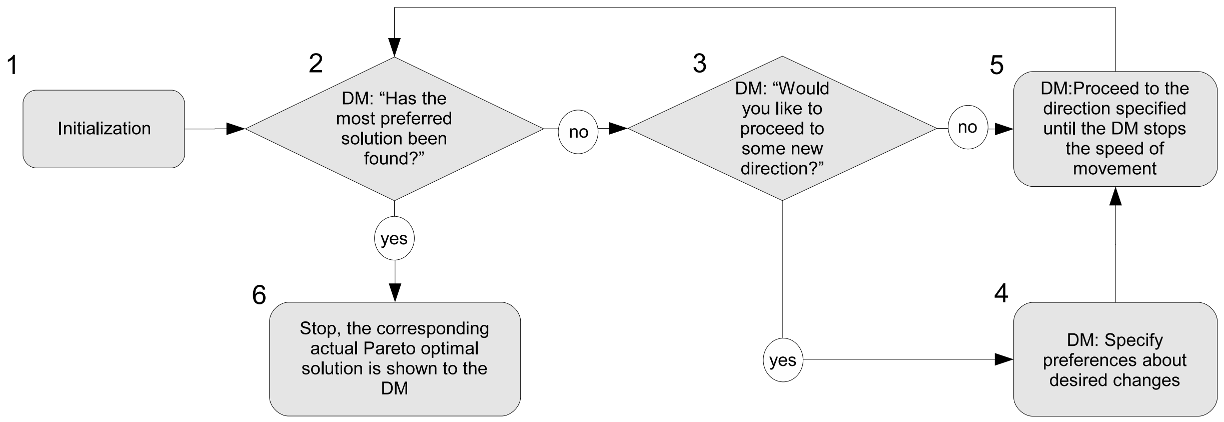
1
Initialization
2
DM: "Has the most preferred solution been found?"
no
yes
3
DM: "Would you like to proceed to some new direction?"
no
yes
5
DM:Proceed to the direction specified until the DM stops the speed of movement
4
DM: Specify preferences about desired changes
6
Stop, the corresponding actual Pareto optimal solution is shown to the DM

Figure 1: Flowchart of Pareto navigator from the DM's perspective.

Initialization phase

Before the DM can start to use Pareto navigator, an initialization procedure is carried out. In Figure 1, this phase is denoted more generally as a step 1 but technically it contains the following two separate substeps.

1a. A discrete representation of the Pareto optimal set and a corresponding polyhedral approximation are produced in the objective space.

This step needs no interaction with the DM. There exist many methods that can be used to generate the discrete representation (see, e.g., [11, 18]). Here we apply an adaptive approximation of the Pareto optimal set [8]. The advantage of this method is the upper bound for the approximation error it produces. It would also be possible to use some evolutionary approaches developed for approximating the Pareto optimal set (see, e.g., [2]). However, it must be pointed out that with evolutionary approaches the resulting solutions may only be nondominated with respect to the final population but not actually Pareto optimal (the actual Pareto optimal set may dominate the polyhedral approximation considered). Computationally, the most demanding task is the production of a discrete representation of the Pareto optimal set. However, this representative set can be generated well before the DM is involved in the method.

Let us assume that we have a discrete set of Pareto optimal solutions related to problem (1). As an example, we describe one possible way to build a polyhedral approximation for the actual Pareto optimal set using these solutions. Because we are dealing with convex problems in this paper, it is convenient to produce the polyhedral approximation using the convex hull of the Pareto optimal solutions available. A convex hull can be expressed in a form $\mathbf{Az} \leq \mathbf{b}$ and this inequality characterizes a polyhedral set. In other words, the inequality holds for objective vectors $\mathbf{z} \in \mathbb{R}^k$ which belong to the convex hull. Later on, when step 5 is described, we explain how a convex hull of the form $\mathbf{Az} \leq \mathbf{b}$ can be used to obtain approximated Pareto optimal solutions.

In order to generate visualizations, we also need information about the ranges of the objective values in the Pareto optimal set. We can approximate ideal and nadir objective values as discussed in Section 2, or alternatively, it is possible to ask the DM for the best and the worst objective values to be considered.

1b. In what follows, the DM is involved. First, the DM is asked to select a starting point for the navigation phase.

For example, (s)he can specify a reference point which is then projected in the approximated Pareto optimal set. The projection can be made, for example, by solving a parametric programming problem (3) introduced in step 5. Alternatively, (s)he can select the most preferred actual Pareto optimal solution from the set used in step 1a. In this, a visualization like the one given in Figure 2 can be helpful for the DM.

Navigation phase

The overall idea of the navigation phase is to allow the DM to move around in the approximated Pareto optimal set in those directions where (s)he feels the most promising solutions could be located. After the desired direction has been found, the method starts, in real-time, to produce approximated Pareto optimal solutions to the direction determined. The approximated solutions are produced with a parametric programming procedure using a relatively small step size such that when the solutions are shown to the DM, (s)he experiences the motion as continuous. The speed of movement (i.e. how rapid the change in objective function values is) is determined by the DM. At any time the DM wishes, (s)he can adjust the speed, change the direction or even go backwards (i.e., where (s)he came from). In this way, the DM rapidly evaluates each approximated Pareto optimal solution produced along the determined direction. In practice, the objective function values can be visualized, for example, using a bar chart. In other words, the DM sees the lengths of the bars changing as the corresponding objective function value changes. Once the DM has found the most promising solution in the current direction, (s)he can stop the movement. This solution is called a *current solution.* In what follows, we describe steps $2 - 6$ forming the navigation phase of Pareto navigator (see Figure 1) in more detail.

2. The objective values of the current solution are visualized to the DM using a bar chart and the DM is asked the question "Has the most preferred solution been found". If the DM is satisfied with this solution, we proceed to step 6. Otherwise, we continue to step 3.

Note that in principle the DM may feel that the solution selected in the initialization phase is good enough as the final solution and we can stop without the navigation phase. However, we assume here that the DM is interested in learning about the problem and studying further solution possibilities.

3. If the DM is not satisfied with the current solution (i.e., (s)he answered no to the question in step 2), (s)he is further asked "Would you like to proceed to some new direction". A negative answer means that the DM wants to continue in the current direction. Then we proceed to step 5. In the opposite case the DM can change one's preferences and then a new search direction is determined.

Note that the DM must always specify preferences related to desired changes in objective values after the initialization phase. After this, the initial search direction can be determined correspondingly.

4. Now, the DM wants to change the search direction and (s)he is asked to specify preferences on how the current solution should be improved. The preference information is requested from the DM, for instance, in the form of a classification or as a reference point.

With the preference information the DM indicates what kind of changes would make the current solution more preferred. This information is used to determine a new search direction from the current solution. Basically, any form of preference information can be used as long as it is possible to extract a unique search direction from it. For example, if a reference point $\bar{\mathbf{z}}$ is used, we can set the search direction as $\mathbf{d} = \bar{\mathbf{z}} - \mathbf{z}^c$, where $\mathbf{z}^c$ is the current solution. Remember that a reference point can also be extracted from classification information, as described in Section 2. Determining the search direction plays a significant role when reflecting the preferences of the DM. Thus, special attention should be paid in selecting a convenient and intuitive form for the DM to specify preferences.

5. From the current solution $\mathbf{z}^c$ we move to the search direction determined by using the preference information obtained from the DM. After the search direction has been determined, the method starts to produce approximated Pareto optimal solutions. The DM can stop the movement at any solution desired. Then we continue to step 2.

As far as producing approximated Pareto optimal solutions is concerned, we solve a parametric programming problem to generate them in real-time. To be more specific, we consider the formulation

$$\begin{array}{ll} \text{minimize} & \max\limits_{i=1,\dots,k} w_i(z_i - \bar{z}_i(\alpha)) \\ \text{subject to} & \mathbf{Az} \leq \mathbf{b}. \end{array} \tag{2}$$

Problem (2) uses an achievement (scalarizing) function employed in the reference point method [22, 23] where w_i, $i = 1, \dots, k$, are the scaling coefficients and the reference point $\bar{\mathbf{z}}(\alpha) = (\mathbf{z}^c + \alpha\mathbf{d}) \in \mathbb{R}^k$ can be moved parametrically to the specified search direction $\mathbf{d} \in \mathbb{R}^k$ by altering parameter $\alpha \in \mathbb{R}$ (negative values of α allow us to move backwards). Here we use scaling coefficients $w_i = 1/(z_i^{nad} - z_i^\star)$. The linear constraints $\mathbf{Az} \leq \mathbf{b}$ of problem (2) are forming a convex hull for a set of Pareto optimal solutions, as described in step 1a, and actually we are now projecting a reference point $\bar{\mathbf{z}}(\alpha)$ to the nondominated facets of the convex hull. This gives an approximated Pareto optimal solution. The parametric problem (2) above can be considered in an equivalent linear form by adding a new variable $\zeta \in \mathbb{R}$. This leads to the following formulation

$$\begin{array}{ll} \text{minimize} & \mathbf{c}^T\mathbf{z}' \\ \text{subject to} & \mathbf{A}'\mathbf{z}' \leq \mathbf{b}' \\ & \mathbf{z}' \in \mathbb{R}^{k+1} \end{array} \tag{3}$$

where

$$\boldsymbol{c} = \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \mathbf{z}' = \begin{pmatrix} \zeta \\ z_1 \\ \vdots \\ z_k \end{pmatrix}, \mathbf{A}' = \begin{pmatrix} -\frac{1}{w_1} & 1 & 0 & \dots & 0 \\ -\frac{1}{w_2} & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -\frac{1}{w_k} & 0 & 0 & \dots & 1 \\ 0 & a_{11} & a_{12} & \dots & a_{1k} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & a_{q1} & a_{q2} & \dots & a_{qk} \end{pmatrix}, \mathbf{b}' = \begin{pmatrix} \bar{z}_1(\alpha) \\ \vdots \\ \bar{z}_k(\alpha) \\ b_1 \\ \vdots \\ b_q \end{pmatrix},$$

where q is the number of linear constraints in problem (2). The problem (3) is a parametric linear programming problem which can be solved with a suitable optimizer. The solution of problem (3) is an approximated Pareto optimal solution in the polyhedral approximation (i.e., due the projection it belongs to one of the nondominated facets of the convex hull). We can generate new solutions fast because we are solving a linear parametric programming problem.

6. Once the DM has found a satisfactory approximated Pareto optimal solution, we stop the search. The approximated solution is projected to the actual Pareto optimal set and the resulting solution is shown to the DM.

The projection can be realized, for example, by setting the selected solution as a reference point and using some achievement scalarizing function to project it (see, e.g. [12, 23]). Note that the DM can at any moment ask for the actual Pareto optimal solution corresponding to the current solution. If the problem is convex, the approximated solution is always feasible. This means that the objective values in the corresponding actual Pareto solution are always as good as in the solution selected by the DM. If the DM is willing to continue navigation after this, we continue to step 2. If desired, it is possible to add the computed actual Pareto optimal solution to the approximation to make it more accurate. This, naturally, necessitates regenerating the polyhedral approximation, that is, we go back to step 1.

After the learning phase has been completed with the Pareto navigator method (after step 6), the DM can stop (if the most preferred solution has been found) or proceed to the decision phase and continue, for example, with some interactive method (see, e.g. [1, 4, 12, 20]), as mentioned earlier. If the DM has specified preferences in the form of reference points or classifications, it may be natural and intuitive to continue with interactive reference point [5, 22] or classification based methods [13, 14, 16, 17], respectively. Alternatively, the DM can continue the learning phase and, for example, ask for a more accurate approximation of the Pareto optimal set to be generated in the neighborhood of the selected solution (see, e.g., [7]). Then we go to step 1 to generate a new set of Pareto optimal solutions in the specified region and initiate Pareto navigator again.

4 Using Pareto navigator

In this section, we demonstrate how the Pareto navigator method can be used from the DM's point of view. To be more specific, by using an example problem, we describe step-by-step what kind of interaction can take place during the navigation phase outlined in Figure 1. However, it must be emphasized that visualizations used and the way how the DM indicates his/her preferences are only suggesting one possible approach, and in this respect the user interface can be customized also in other ways to meet the needs of the DM. We consider the following simple problem:

$$\begin{array}{ll}
\text{minimize} & \left\{ \begin{array}{c} -x_1 - x_2 + 5 \\ \frac{1}{5}(x_1^2 - 10x_1 + x_2^2 - 4x_2 + 11) \\ (5 - x_1)(x_2 - 11) \end{array} \right\} \\
\text{subject to} & 3x_1 + x_2 - 12 \le 0 \\
& 2x_1 + x_2 - 9 \le 0 \\
& x_1 + 2x_2 - 12 \le 0 \\
& 0 \le x_1 \le 4, \quad 0 \le x_2 \le 6.
\end{array}$$

Actually this example problem is not convex due the third objective function but we can say that it is mildly nonconvex. However, the example demonstrates that this does not affect the applicability of Pareto navigator.

1a. To begin with, we apply the adaptive approximation method in the initialization phase and as a result we get a list of actual Pareto optimal solutions (see Table 1) to be used to produce a polyhedral approximation. Here we have selected to use 7 solutions. After using the adaptive approximation method, we also get approximations for the ideal and the nadir objective vectors. In what follows, we describe actions of a DM and explain how Pareto navigator can be used to search for a promising solution in the objective space.

	f_1	f_2	f_3
1	-2.00	0.00	-18.00
2	-1.00	4.60	-25.00
3	0.00	-3.10	-14.25
4	1.38	0.62	-35.33
5	1.73	1.72	-38.64
6	2.48	1.45	-42.41
7	5.00	2.20	-55.00

Table 1: Initial set of Pareto optimal solutions.

1b. The DM specifies the starting point for Pareto navigator by selecting the most interesting actual Pareto optimal solution from the initial set listed in Table 1. The same set of solutions is also shown with a value path visualization in Figure 2 (remember that the starting point could alternatively be specified with a reference point.) Let us now assume that the solution 4 is the most appealing to the DM.

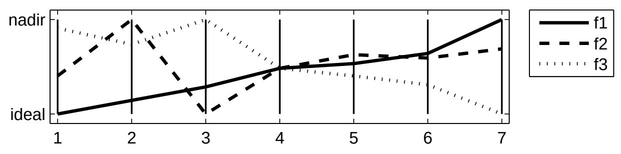

Figure 2: Value path visualization of the initial set of Pareto optimal solutions.

2. In what follows, we refer to the solution selected as $A = (1.38, 0.62, -35.33)^T$ (see Figure 3) and assume that the DM wants to examine its surroundings.

3. The initial solution A has now been selected but we do not yet have a search direction. After the initialization step, it is obligatory for the DM to specify preferences in order to set the search direction.

4. When studying A, the DM is interested in finding solutions where values of objectives f_1 and f_2 are improving while the value of objective f_3 can be relaxed. By using this classification information, an initial search direction from the selected solution A can be generated. To be more precise, the classification made produces a reference point $\bar{\mathbf{z}}^1 = (z_1^\star, z_2^\star, z_3^{nad})^T$ which is used together with the current solution A to produce the search direction.

5. Pareto navigator starts to generate solutions in the direction determined and updates in real-time the bar chart visualization depicting approximated solutions that are generated. The DM is able to determine a desired speed of movement. Based on the continuously changing lengths of bars in the bar chart visualization, the DM can rapidly see what kind of solutions are available in the current direction (which corresponds to the given classification). When an interesting area has been reached, the DM stops the movement. Note that the DM is also allowed to move backwards in the current direction if the DM feels that (s)he already passed an area which seemed more interesting than the current one.

2. Let us now assume that the DM has arrived from the starting solution A to the solution $B = (0.35, -0.51, -26.26)^T$, where (s)he has stopped the movement (see Figure 3). However, solution B is not satisfactory to the DM.

3. While moving from A to B the DM has learned how the solutions are behaving with respect to the classification given at point A. The DM thinks that continuing in this direction is no more interesting and wants to change the direction by giving a new classification indicating how objective values should be changed from those obtained at B.

4. The DM classifies the objectives at B in such a way that the value of objective f_1 should be improved even more and therefore allows objective f_2 to degrade. On the other hand, the DM feels that the value of objective f_3 should maintain its current level. A new search direction is generated based on this classification similarly as illustrated in the context of the first classification.

5. The method starts to produce approximated solutions to the new search direction. The DM analyzes in real-time the bar chart visualization until an interesting solution is achieved.

2. The DM stops at solution $C = (-0.64, 1.82, -25.95)^T$ (see Figure 3) and decides to explore the surrounding solutions further.

3. (S)he feels that the current direction no longer produces solutions that are interesting enough.

4. The DM is satisfied with the value of f_1 but is ready to sacrifice in its value to slightly improve objectives f_2 and f_3. This classification gives us again a new search direction as previously described.

5. The DM continues to analyze approximated solutions in the direction determined until (s)he sees a promising solution.

2. The solution $D = (-0.32, 2.33, -27.85)^T$ (see Figure 3) seems very satisfactory to the DM.

6. The navigation phase is stopped and the final approximated Pareto optimal solution D is now projected to the actual Pareto optimal set, as described in Section 3. The corresponding actual Pareto optimal solution $(-0.33, 2.32, -27.91)^T$ is shown to the DM. The DM is satisfied with this solution, which is quite similar to the approximated solution D. The whole solution process can be stopped.

As we can see, the DM was able to learn about the solutions available and conveniently direct the search for the most preferred solution. The whole solution process described above is summarized and depicted in Figure 3 where the vertical value path visualization on the right side is reflecting what kind of objective vectors the DM has analyzed during the solution process. On the left side of Figure 3, the solutions A, B, C, and D are snapshots of the vertical value path. These are the solutions where the DM has temporarily stopped the movement and the continuously changing visualization in the bar chart, to determine a new search direction (i.e., classification). It is not trivial to illustrate real-time movements here. Thus, it must be emphasized that during the solution process the DM analyzes approximated Pareto optimal solutions using only one bar chart visualization where the lengths of the bars are changing in real-time.

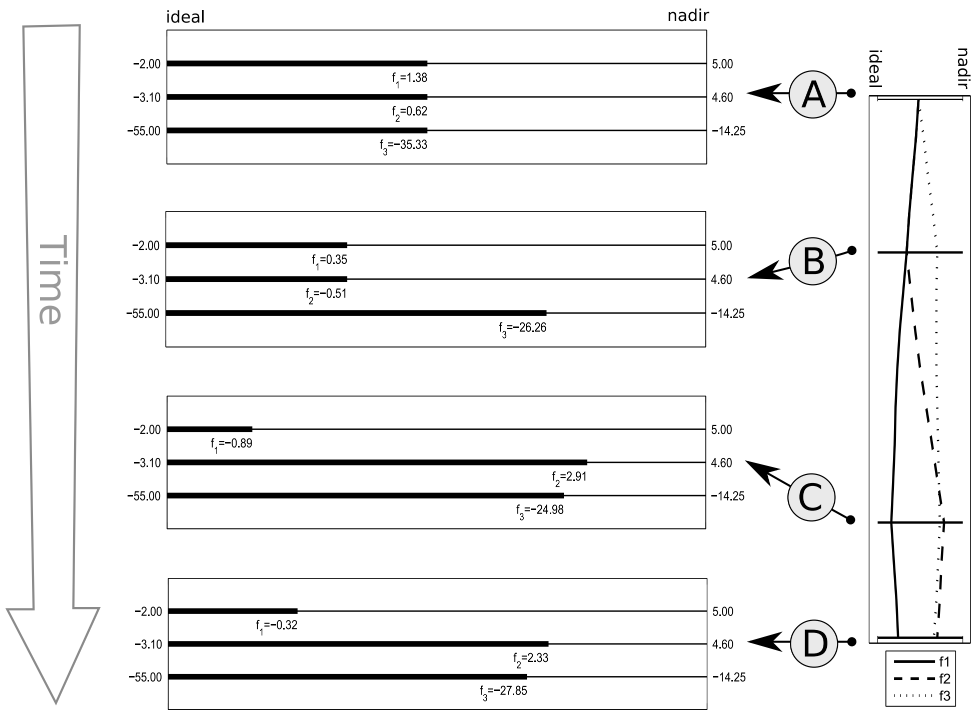

Figure 3: The progress of the method.

Because our simple example contains only three objectives, it is possible to illustrate what happened in the objective space during the solution process described. In Figure 4, the triangulated area represents the polyhedral approximation (i.e., nondominated facets of the convex hull) which was constructed by using the initial set of Pareto optimal solutions (vertex points in Figure 4). The approximation contains only nondominated facets so for the DM the movement on this polyhedral approximation feels like moving in the actual Pareto optimal set. Solutions A, B, C, and D are the same approximated Pareto optimal solutions that were obtained during the solution process. Figure 5 illustrates the actual Pareto optimal set. The black points in Figure 5 corresponds to the vertex points in Figure 4. It must be emphasized that in a general case it may be computationally very expensive to produce a visualization of the actual Pareto optimal set (as presented in Figure 5). Naturally, for problems with more than three objective functions, we cannot generate visualizations like Figures 4 and 5.

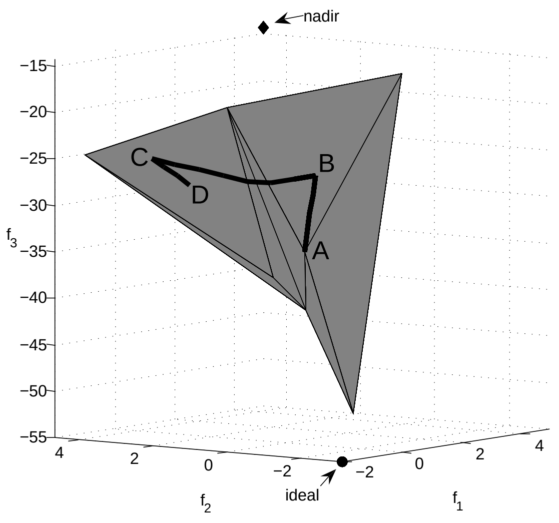

Figure 4: The progress in the objective space

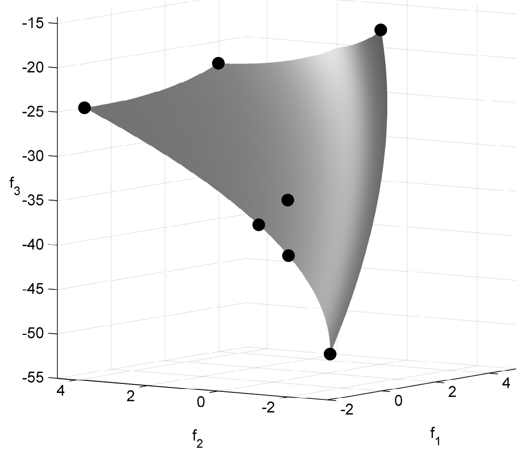

Figure 5: The actual Pareto optimal set

5 Discussion

In what follows, we wish to discuss some topics related to the implementation and future research. First of all, we must emphasize that in this kind of an interactive method, the user interface plays a very important role. When designing the user interface, it is especially important to consider how the DM is able to indicate preferences (for setting a search direction) to utilize the full strength of the Pareto navigator method. The interaction should be as easy, intuitive and as fast as possible.

As far as setting the search direction is concerned, we can say that Pareto navigator is a more versatile method than what the implementation of Pareto race was. This is because the DM could only say which objective function was to be improved in Pareto race, whereas Pareto navigator allows the DM to express desires for all the objective functions about how their values should change. Future research could include a comparison of different methods to aid the DM to determine directions while navigating in the approximation.

When describing the idea of classification in general, we mentioned that some objective function should be allowed to get worse if some other is desired to be improved. However, in Pareto navigator the DM can violate this, if so desired. In other words, the DM is allowed to ask for improvement in all the objectives. Naturally, it is not possible to improve all objective values simultaneously but the search direction can be set so. In this case, when producing new solutions, the method selects some objective functions to get worse in order to allow the others to improve. However, if the DM wants to feel being more in control, it is recommended that the classification indicates which are the objective functions that can impair in value so that the others can improve. This kind of behavior must be also taken into account when designing a user interface.

Our method may have difficulties when dealing with, for example, design problems where the visualization of solutions needs variable values. In such cases, in addition to the objective values, the DM analyzes the quality of solutions via visualization. However, while moving in the approximated Pareto optimal set, a connection to the variable space is temporarily lost. This may be regarded as a shortcoming of Pareto navigator, but this is the price to be paid for enabling a computationally inexpensive navigation phase. Let us point out that the actual Pareto optimal solution together with the corresponding variable values can always be obtained by using projection (in step 6 of the algorithm).

As mentioned before, once the DM has asked for the actual Pareto optimal solution corresponding to the current approximated one, one does not have to stop the whole solution process. Instead, one can continue the navigation from that point. It is also possible to update the approximation used by including the actual Pareto optimal solutions generated so far in the set that is the basis of the polyhedral approximation and repeating the initialization phase. In this way, the approximation can be made more accurate while the DM is using the method. Naturally, this option is convenient only if we assume that the computation of actual Pareto optimal solution does not take too much time.

Even though our method has here been described for convex problems, it can basically be used for some other problems, as well. However, in convex cases we always know that the approximated solutions are dominated by the corresponding actual Pareto optimal solutions. In nonconvex problems, this is not necessarily the case. For example, as shown in Section 4, the applicability of Pareto navigator is not restricted to convex problems only. Moreover, the general idea of navigating around a polyhedral approximation of the Pareto optimal set can be extended to nonconvex problems. The implementation of the navigation idea in the case of highly nonconvex problems (that cannot be approximated by a convex polyhedron) is subject to future research.

In this paper, we have used the concept of Pareto optimality to determine what kind of solutions of problem (1) are interesting to the DM. However, despite the name, the Pareto navigator method can also be used in the case of more general dominance structures than Pareto dominance (see, e.g., [19]).

6 Conclusions

We have described a new interactive learning-oriented method for convex nonlinear multiobjective optimization, called Pareto navigator. The method enables convenient navigation in the approximated Pareto optimal set. With this method, the DM can see changes in the trade-offs between conflicting objectives in real time. This intuitive approach gives general understanding of the possibilities and limitations of the problem considered and, thus, supports learning. Even though we operate in the approximated Pareto optimal set, the DM can at any moment see the corresponding actual Pareto optimal solution. Because most of the computations are carried out in the approximated Pareto optimal set (and this can be realized with the help of parametric linear programming), the method is particularly suitable for computationally challenging real-world problems.

We have demonstrated Pareto navigator with an example with three objective functions. However, our approach is not limited to such problems because bar charts can easily be used as the main means of visualization also when solving problems with more objectives. Particularly in cases when a visualization of the whole Pareto optimal set is not available, Pareto navigator is a useful tool to explore the Pareto optimal set in an interactive and efficient way.

References

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

[2] K. Deb. *Multi-Objective Optimization using Evolutionary Algorithms.* Wiley & Sons, Chichester, 2001.

[3] K. Deb, S. Chaudhuri, and K. Miettinen. Towards estimating nadir objective vector using evolutionary approaches. In M. Keijzer, editor, *Proceedings of the 8th Annual Genetic and Evolutionary Computation Conference (GECCO-2006), Seattle*, volume 1, pages 643–650, New York, 2006. The Association of Computing Machinery (ACM).

[4] C.-L. Hwang and A.S.M. Masud. *Multiple Objective Decision Making – Methods and Applications: A State-of-the-Art Survey.* Lecture Notes in Economics and Mathematical Systems 164. Springer–Verlag, Berlin, Heidelberg, 1979.

[5] A. Jaszkiewicz and R. Slowiński. The 'Light Beam Search' approach – An overview of methodology and applications. *European Journal of Operational Research*, 113:300–314, 1999.

[6] I. Kaliszewski. Out of the mist – Towards decision-maker-friendly multiple criteria decision making support. *European Journal of Operational Research*, 158:293–307, 2004.

[7] K. Klamroth and K. Miettinen. Integrating approximation and interactive decision making in multicriteria optimization. *Operations Research.* To appear.

[8] K. Klamroth, J. Tind, and M.M. Wiecek. Unbiased approximation in multicriteria optimization. *Mathematical Methods of Operations Research*, 56:413–437, 2002.

[9] M. Köksalan and R.D. Plante. Interactive multicriteria optimization for multiple-response product and process design. *Manufacturing & Service Operations Management*, 5:334–347, 2003.

[10] P. Korhonen and J. Wallenius. A Pareto race. *Naval Research Logistics*, 35:615–623, 1988.

[11] A. V. Lotov, V.A. Bushenkov, and G. K. Kamenev. *Interactive Decision Maps. Approximation and Visualization of Pareto Frontier.* Kluwer, 2004.

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

[13] K. Miettinen and M.M. Mäkelä. Interactive bundle-based method for nondifferentiable multiobjective optimization: NIMBUS. *Optimization*, 34:231–246, 1995.

[14] K. Miettinen and M.M. Mäkelä. Interactive multiobjective optimization system WWW-NIMBUS on the internet. *Computers & Operations Research*, 27:709–723, 2000.

[15] K. Miettinen and M.M. Mäkelä. On scalarizing functions in multiobjective optimization. *OR Spectrum*, 24:193–213, 2002.

[16] K. Miettinen and M.M. Mäkelä. Synchronous approach in interactive multiobjective optimization. *European Journal of Operational Research*, 170:909–922, 2006.

[17] H. Nakayama and Y. Sawaragi. Satisficing trade-off method for multiobjective programming. In M. Grauer and A.P. Wierzbicki, editors, *Interactive Decision Analysis*, pages 113–122. Springer–Verlag, 1984.

[18] S. Ruzika and M. M. Wiecek. Approximation methods in multiobjective programming. *Journal of Optimization Theory and Applications*, 126:473–501, 2005.

[19] Y. Sawaragi, H. Nakayama, and T. Tanino. *Theory of Multiobjective Optimization.* Academic Press, Inc., 1985.

[20] R.E. Steuer. *Multiple Criteria Optimization: Theory, Computation and Applications.* John Wiley & Sons, Inc., 1986.

[21] D. Vanderpooten. The interactive approach in MCDA: A technical framework and some basic concceptions. *Mathematical and Computer Modelling*, 12(10–11):1213–1220, 1989.

[22] A.P. Wierzbicki. A mathematical basis for satisficing decision making. *Mathematical Modelling*, 3(25):391–405, 1982.

[23] A.P. Wierzbicki. On the completeness and constructiveness of parametric characterizations to vector optimization problems. *OR Spectrum*, 8(2):73–87, 1986.