Teichert, Katrin; Seidel, Tobias; Süss, Philipp

Article — Published Version

Combining discrete and continuous information for multi-criteria optimization problems

Mathematical Methods of Operations Research

Suggested Citation: Teichert, Katrin; Seidel, Tobias; Süss, Philipp (2024) : Combining discrete and continuous information for multi-criteria optimization problems, Mathematical Methods of Operations Research, ISSN 1432-5217, Springer Berlin Heidelberg, Berlin/Heidelberg, Vol. 100, Iss. 1, pp. 153-173, https://doi.org/10.1007/s00186-024-00849-0

This Version is available at:
https://hdl.handle.net/10419/314961

Mathematical Methods of Operations Research (2024) 100:153–173
https://doi.org/10.1007/s00186-024-00849-0

ORIGINAL ARTICLE

Combining discrete and continuous information for multi-criteria optimization problems

Katrin Teichert[1] · **Tobias Seidel**[1] · **Philipp Süss**[1]

Received: 11 April 2023 / Revised: 4 January 2024 / Accepted: 4 January 2024 /
Published online: 23 February 2024

Abstract
In multi-criteria optimization problems that originate from real-world decision making tasks, we often find the following structure: There is an underlying continuous, possibly even convex model for the multiple outcome measures depending on the design variables, but these outcomes are additionally assigned to discrete categories according to their desirability for the decision maker. Multi-criteria deliberations may then take place at the level of these discrete labels, while the calculation of a specific design remains a continuous problem. In this work, we analyze this type of problem and provide theoretical results about its solution set. We prove that the discrete decision problem can be tackled by solving scalarizations of the underlying continuous model. Based on our analysis we propose multiple algorithmic approaches that are specifically suited to handle these problems. We compare the algorithms based on a set of test problems. Furthermore, we apply our methods to a real-world radiotherapy planning example.

Keywords Multi-criteria optimization · Decision making · Non-linear optimization · Pareto front approximation

1 Introduction

In many practical applications, the decision maker's utility function is of a step-wise nature even though the underlying measure and its dependence on the real-valued optimization variables is continuous. For example, in radiotherapy one tries to have a sufficiently high dose in the target and then, provided this is the case, as little dose in surrounding organs at risk as possible. While the dose values in those nearby organs are continuously dependent on administered radiation, from a physician's point of

✉ Katrin Teichert
katrin.teichert@itwm.fraunhofer.de

[1] Fraunhofer Institute for Industrial Mathematics (ITWM), Fraunhoferplatz 1, 67663 Kaiserslautern, Germany

view they are judged on whether they exceed certain threshold values associated with particular side effects. Thereby, a dose in an organ might be labeled as either "dangerously high", "acceptable", or "ideal". The multi-criteria decision making would then deliberate on the trade-offs between different organs based on these broad categorizations.

We formalize this observation by defining the following type of multi-criteria optimization problem:

$$\mathcal{P}: \quad \min_{\boldsymbol{x} \in M} d_1(f_1(\boldsymbol{x})), \ldots, d_k(f_k(\boldsymbol{x})) \tag{1}$$

where we assume $M \subseteq \mathbb{R}^n$ to be compact and non-empty, $f_1, \ldots, f_k : M \to \mathbb{R}$ to be continuous functions, and $d_1, \ldots, d_k : \mathbb{R} \to \mathbb{R}$ to be monotone increasing. We let $\boldsymbol{f} := (f_1, \ldots, f_k)$ and $\boldsymbol{d} := (d_1 \circ \pi_1, \ldots, d_k \circ \pi_k)$, where π_i denotes the projection from a vector in $\mathbb{R}^k$ to the i-th component. We assume that there is no a priori preference between the objectives.

To establish the discrete nature of our problem, we require the following assumption for the monotone increasing utility functions d_i:

Assumption 1.1 (Discrete utility) For $i \in \{1, \ldots, k\}$ there is a finite set $\Lambda_i \subset \mathbb{R}$ such that $d_i : \mathbb{R} \to \Lambda_i$.

In the literature, there is a vast amount of general-purpose approaches that can be applied to multi-criteria problems with discrete and continuous decision variables and objectives alike (Steuer and Choo 1983; Benson and Sayin 1997; Das and Dennis 1998; Schandl et al. 2002). While these are applicable to $\mathcal{P}$, they do not take into account its special structure. An efficient algorithm for $\mathcal{P}$ should ideally combine certain characteristics of algorithms intended specifically for continuous problems with those specific for discrete problems.

When facing a problem with the structure of $\mathcal{P}$, the aim is to find the non-dominated solutions with respect to the discrete utility function evaluations d_i. However, we also have an underlying continuous problem that we can potentially capitalize on:

$$\mathcal{P}^c: \quad \min_{\boldsymbol{x} \in M} f_1(\boldsymbol{x}), \ldots, f_k(\boldsymbol{x}). \tag{2}$$

Similar to how non-dominated solutions are obtained in the purely continuous setting, efficient solutions to (2) can be calculated with potentially very efficient general-purpose continuous solvers, such as IPOPT (Wächter and Biegler 2006) or knitro (Byrd et al. 2006). Suitable scalarization methods are weighted sum if the problem is convex, and ϵ-constraint (see e.g. Ehrgott 2005), or Pascoletti–Serafini scalarization (Pascoletti and Serafini 1984) in the case of non-convex problems. Particularly suited approaches for approximating the Pareto front of a continuous problem are sandwiching (Serna 2012; Bokrantz and Forsgren 2012) in case the problem is convex, and hypervolume or hyperboxing algorithms (Bringmann and Friedrich 2010; Teichert 2014) if the problem is non-convex.

However, algorithms for continuous problems are approximation algorithms: they aim at calculating a discrete set of non-dominated solutions that represent the Pareto

front up to some approximation error measure, often by establishing lower and/or upper bounds on the front (Sayın 2000; Klamroth et al. 2002; Eichfelder 2009). In contrast, for our setting we do not aim at an approximation of the Pareto front, but a full representation. That is, we want to find at least one efficient solution for each non-dominated point in the image space of the discrete utility functions. In this regard, our approach is in line with many algorithmic approaches for calculating the Pareto front of a purely discrete problem (Ulungu and Teghem 1994; Holzmann and Smith 2018).

While it is in fact possible to solve the multi-criteria problem $\mathcal{P}$ analogously to a purely discrete problem with those techniques, such an approach is unnecessarily computationally expensive, as the more difficult discrete problem is solved repeatedly and not the potentially simpler continuous problem. On the other hand, one could solve the continuous problem $\mathcal{P}^c$ in the approximative sense and, in a second step, infer the discrete front. But then many points on the continuous Pareto front may be calculated that are redundant for the discrete Pareto front of $\mathcal{P}$. This is why in this paper we aim to combine both worlds. We will present algorithms that avoid solving the discrete problem and rather solve the simpler continuous problem $\mathcal{P}^c$. Simultaneously, the algorithms will incorporate the discrete aspects of $\mathcal{P}$ to avoid solving redundant continuous problems.

The outline for this paper is as follows. We first investigate the relationship between the solutions of $\mathcal{P}$ and $\mathcal{P}^c$ (Sect. 2). Based on this theoretical insight, we then propose different algorithms to find all non-dominated points for $\mathcal{P}$ (Sect. 3). Finally, we demonstrate and evaluate these algorithms using a set of test problems and an example from radiotherapy (Sect. 4). We conclude with a discussion of our results (Sect. 5).

2 Theoretical results

In this section, we investigate the theoretical properties of $\mathcal{P}$ (1). In particular, we describe the relationship between peculiar solutions of $\mathcal{P}$ and their corresponding counterparts with respect to the underlying continuous problem $\mathcal{P}^c$ (2). This is crucial for the development and the analysis of the algorithms described later in this paper. Note that for all observations within this section, Assumption 1.1 is not required; they hold for arbitrary monotone functions.

Before we present the theoretical results, we summarize some notation used throughout this paper. For two vectors $\boldsymbol{y}^1, \boldsymbol{y}^2 \in \mathbb{R}^k$ we write $\boldsymbol{y}^1 \leqq \boldsymbol{y}^2$ if the inequality $y_i^1 \leq y_i^2$ holds for every component $i = 1, \dots k$ (analogously for $\geqq, <, >$). We say that a point $\boldsymbol{y} \in \mathbb{R}^k$ dominates a point $\boldsymbol{y}' \in \mathbb{R}^k$ if:

$$\begin{aligned} y_i &\leq y_i' \text{ for all } i \in \{1, \dots, k\} \\ y_i &< y_i' \text{ for at least one } i \in \{1, \dots, k\}. \end{aligned}$$

We say that a point $\boldsymbol{y} \in \mathbb{R}^k$ strictly dominates a point $\boldsymbol{y}' \in \mathbb{R}^k$ if for all $i \in \{1, \dots, k\}$ the inequality $y_i < y_i'$ holds. The image of the combined MCO problem $\mathcal{P}$ is denoted by

$$\mathcal{Y} := (\boldsymbol{d} \circ \boldsymbol{f})(M). \tag{3}$$

For the the underlying continuous problem $\mathcal{P}^c$ the image is denoted by

$$\mathcal{Y}^c := \boldsymbol{f}(M). \tag{4}$$

A point $\boldsymbol{x} \in M$ is called efficient solution for $\mathcal{P}$ (for $\mathcal{P}^c$) if there is no $\boldsymbol{y} \in \mathcal{Y} \quad (\boldsymbol{y} \in \mathcal{Y}^c)$ such that $\boldsymbol{y}$ dominates $(\boldsymbol{d} \circ \boldsymbol{f})(\boldsymbol{x})$ (dominates $\boldsymbol{f}(\boldsymbol{x})$). The corresponding image point is called non-dominated. A point $\boldsymbol{x} \in M$ is called weakly efficient solution for $\mathcal{P}$ (for $\mathcal{P}^c$) if there is no $\boldsymbol{y} \in \mathcal{Y} \quad (\boldsymbol{y} \in \mathcal{Y}^c)$ such that $\boldsymbol{y}$ strictly dominates $(\boldsymbol{d} \circ \boldsymbol{f})(\boldsymbol{x})$ (strictly dominates $\boldsymbol{f}(\boldsymbol{x})$). The corresponding image point is called weakly non-dominated.

We can now begin to introduce statements about the relationship between particular solutions of $\mathcal{P}$ and $\mathcal{P}^c$. We first observe the connection between weakly non-dominated points of the two problems.

Lemma 2.1 *If $\boldsymbol{y}^{c,*} \in \mathcal{Y}^c$ is a weakly non-dominated point of $\mathcal{P}^c$ then $\boldsymbol{y}^* := \boldsymbol{d}(\boldsymbol{y}^{c,*}) \in \mathcal{Y}$ is weakly non-dominated for $\mathcal{P}$.*

Proof For an arbitrary point $\boldsymbol{y}$ in the image $\mathcal{Y}$ of $\mathcal{P}$ there is at least one point $\boldsymbol{y}^c \in \mathcal{Y}^c$ such that $\boldsymbol{y} = \boldsymbol{d}(\boldsymbol{y}^c)$. Now consider the weakly non-dominated point $\boldsymbol{y}^{c,*} \in \mathcal{Y}^c$. There must be at least one index $i \in \{1, \ldots, k\}$ such that

$$y_i^{c,*} \leq y_i^c,$$

as otherwise $\boldsymbol{y}^c$ would strictly dominate $\boldsymbol{y}^{c,*}$ and $\boldsymbol{y}^{c,*}$ could not be a weakly non-dominated point. For the same index i it follows by monotonicity of d_i that

$$y_i^* = d_i(y_i^{c,*}) \leq d_i(y_i^c) = y_i.$$

This shows that $\boldsymbol{y}$ does not strictly dominate $\boldsymbol{y}^*$. As $\boldsymbol{y}$ was chosen arbitrary in $\mathcal{Y}$, this shows that also $\boldsymbol{y}^*$ is a weakly non-dominated point. □

Lemma 2.1 also implies that a non-dominated point is mapped to a weakly non-dominated point. Unfortunately this point is not necessarily non-dominated anymore, as the following example shows:

Example 2.2 Consider the following example (illustrated in Fig. 1):

- $M = \{\boldsymbol{x} \in \mathbb{R}^2 \mid \|\boldsymbol{x}\|_2 \leq 1\}$
- $k = 2$, $f_1(\boldsymbol{x}) = x_1$, $f_2(\boldsymbol{x}) = x_2$
- for $t \in \mathbb{R}$: $d_1(t) = d_2(t)$ and

$$d_1(t) = \begin{cases} 1 & \text{if} \quad t \leq -0.5, \\ 2 & \text{if} \quad t > -0.5. \end{cases}$$

The point $(-1, 0)$ is a non-dominated point for problem $\mathcal{P}^c$. However, by $\boldsymbol{d}$ it is mapped to the point $(1, 2)$. This point is dominated by $(1, 1)$, which is the image of the feasible solution $(-0.5, -0.5)$ under $\boldsymbol{d} \circ \boldsymbol{f}$.

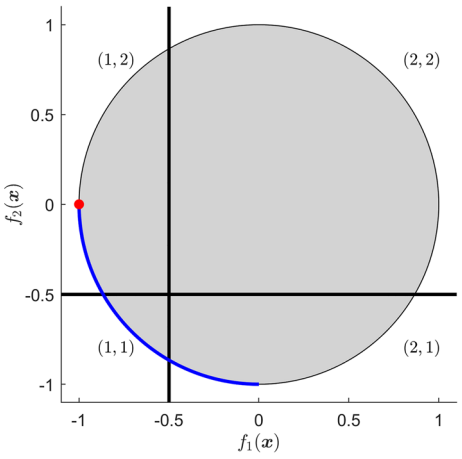


Fig. 1 Illustration of Example 2.2. The image set $\mathcal{Y}^c$ of the inner problem $\mathcal{P}^c$ is depicted in gray and its Pareto front in blue. The non-dominated point $(-1, 0)$, depicted in red, is mapped by $\boldsymbol{d}$ to the weakly non-dominated point $(1, 2)$. This point is dominated by $(1, 1)$

The example also shows that the reverse relationship does not hold. If $\boldsymbol{d}(\boldsymbol{y}^c)$ is a weakly non-dominated point, then the preimage can be strictly dominated. The point $(-0.5, -0.5)$ is dominated in $\mathcal{P}^c$ by the point $(-\frac{\sqrt{2}}{2}, -\frac{\sqrt{2}}{2})$. However, $\boldsymbol{d}((-0.5, -0.5))$ is a non-dominated point. This means that not all preimages of a non-dominated point are again (weakly) non-dominated. The following can be shown about the existence of non-dominated points in the preimage:

Lemma 2.3 *Let $\boldsymbol{y}^* \in \mathcal{Y}$ be a non-dominated point of $\mathcal{P}$. Then there exists a non-dominated point $\boldsymbol{y}^{c,*} \in \mathcal{Y}^c$ of $\mathcal{P}^c$ such that*

$$\boldsymbol{y}^* = \boldsymbol{d}(\boldsymbol{y}^{c,*}).$$

Proof Let

$$Z = \boldsymbol{d}^{-1}(\boldsymbol{y}^*) \cap \mathcal{Y}^c$$

denote the preimage of the non-dominated point $\boldsymbol{y}^* \in \mathcal{Y}$ given in the statement. By assumption this set is nonempty. We want to show that we can find a point in Z that is non-dominated for $\mathcal{P}^c$. First consider any point $\boldsymbol{z} \in Z$ and a point $\boldsymbol{z}' \in \mathcal{Y}^c$ such that:

$$z'_i \leq z_i \text{ for all } i \in \{1, \ldots, k\}. \tag{5}$$

Then by monotonicity of d_i we have:

$$d_i(z'_i) \leq d_i(z_i) = y^*_i \text{ for all } i \in \{1, \ldots, k\}$$

As $\boldsymbol{y}^*$ is non-dominated, we must have:

$$\boldsymbol{d}(\boldsymbol{z}') = \boldsymbol{y}^*.$$

This means that every point $z \in \mathcal{Y}^c$ satisfying the inequalities in Eq. (5) is already contained in Z.

As we did not assume any continuity properties for d, the set Z is not necessarily closed. This is why we consider the closure of this set denoted by $\overline{Z}$. As we assumed that f is continuous and M is compact, the set $\overline{Z}$ is as a subset of $\mathcal{Y}^c = \boldsymbol{f}(M)$ compact.

Now fix an arbitrary point $\boldsymbol{z}^* \in Z$ and consider the following optimization problem:

$$\begin{aligned} \min_{z \in \overline{Z}} \quad & \sum_{i=1}^{k} z_i \\ \text{s.t.} \quad & z_i \leq z_i^* \text{ for all } i \in \{1, \ldots, k\} \end{aligned}$$

The feasible set is compact and non-empty by construction. Moreover every feasible point of this problem is by the arguments given above in Z. This means that also an optimal solution $\boldsymbol{y}^{c,*} \in \mathcal{Y}^c$ to this problem is again in Z, which then means that

$$d(\boldsymbol{y}^{c,*}) = d(\boldsymbol{y}^*).$$

It remains to argue that $\boldsymbol{y}^{c,*}$ is instead a non-dominated point for $\mathcal{P}^c$. Because of optimality, $\boldsymbol{y}^{c,*}$ cannot be dominated by any point in $\overline{Z}$. Moreover, every point in $\mathcal{Y}^c$ that would dominate $\boldsymbol{z}$ also satisfies (5) and is again in Z. Together this means that the constructed point cannot be dominated by any point in $\mathcal{Y}^c$ and is indeed a non-dominated point for problem $\mathcal{P}^c$. □

This Lemma is crucial for an algorithmic approach. It shows that we can find the Pareto front of the perhaps more complex problem $\mathcal{P}$ by calculating points of the Pareto front of the continuous problem $\mathcal{P}^c$. For completeness we show in the next example that the statement of Lemma 2.3 does not hold true, if one replaces non-dominated by weakly non-dominated.

Example 2.4 Consider the same feasible set and the same objectives as in Example 2.2 but consider the following utility function:

- for $t \in \mathbb{R}$: $d_1(t) = d_2(t)$ and

$$d_1(t) = \begin{cases} 1 & \text{if} \quad t \leq 0.5, \\ 2 & \text{if} \quad t > 0.5. \end{cases}$$

For an illustration, see Fig. 2. In this example, the point $(2, 1)$ is a weakly non-dominated point for $\mathcal{P}$. However, all points in $\mathcal{Y}^c$ that are mapped to this point are strictly dominated in $\mathcal{P}^c$, e.g. by $(-1, 0)$.

3 Algorithms

In this section, we discuss different algorithms tailored at calculating the Pareto front for a problem $\mathcal{P}$ with discrete utility mappings. That is, for the algorithms presented

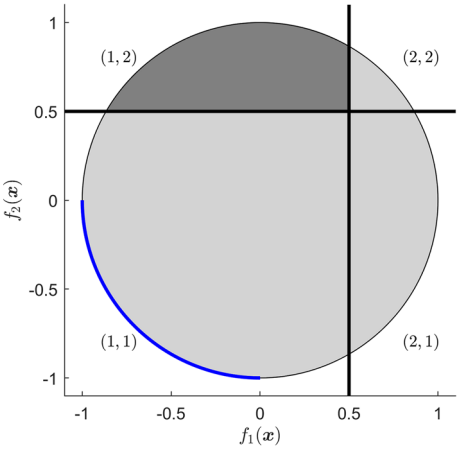


Fig. 2 Illustration of Example 2.4. The image set $\mathcal{Y}^c$ of the inner problem $\mathcal{P}^c$ is depicted in gray and its Pareto front in blue. Every point in the dark area is strictly dominated in the inner problem $\mathcal{P}^c$, but it is the preimage of a weakly non-dominated point (1,2) of $\mathcal{P}$

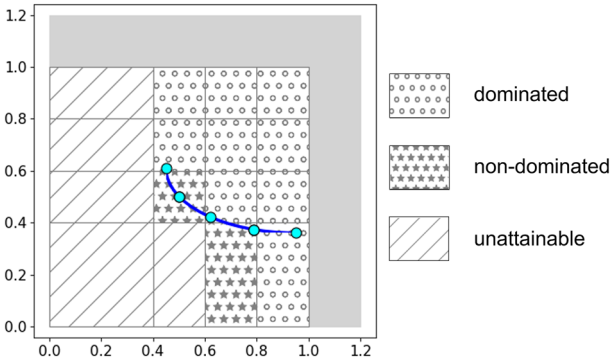


Fig. 3 Illustration of exhaustively solving the multi-criteria problem $\mathcal{P}$. The preimages of the points $\boldsymbol{y} \in \prod_{i=1}^{k} \Lambda_i$ divide the image space of the continuous problem $\mathcal{P}^c$ into boxes. By finding specific solutions to the inner problem—which map to points on the continuous Pareto front depicted in blue—we can determine whether each box, and thus each $\boldsymbol{y} \in \prod_{i=1}^{k} \Lambda_i$, is non-dominated, dominated, or unattainable

in this section to be applicable, we from now on explicitly require that Assumption 1.1 is satisfied. This means that for the image of $\mathcal{P}$ we have:

$$\mathcal{Y} \subseteq \prod_{i=1}^{k} \Lambda_i .$$

The proposed algorithms utilize the special structure of $\mathcal{P}$. On the one hand, all proposed methods employ a gradient-based solver to find either feasible or weakly efficient solutions $\boldsymbol{x}$ of $\mathcal{P}$ by iteratively solving particular single-criteria optimization problems—so-called scalarizations—derived from the continuous inner problem $\mathcal{P}^c$. On the other hand, the algorithms also take the discrete utility mappings into account, namely when choosing which scalarization to solve next and when evaluating their progress.

The proposed algorithms aim to exhaustively solve the multi-criteria problem $\mathcal{P}$ (see Fig. 3). This means that:

- For all $\boldsymbol{y} \in \prod_{i=1}^{k} \Lambda_i$, the algorithm determines whether $\boldsymbol{y}$ is non-dominated, dominated or unattainable.
- For any $\boldsymbol{y} \in \prod_{i=1}^{k} \Lambda_i$ that is non-dominated, the algorithm calculates a solution $\boldsymbol{x} \in M$ such that $\boldsymbol{y} = (\boldsymbol{d} \circ \boldsymbol{f})(\boldsymbol{x})$.

In the applications we will introduce in Sect. 4 the cardinality of each individual Λ_i will not only be finite but also small (3–5). Additionally we will not have arbitrary many objectives in mind (in the examples 2–4). This means that it is possible to iterate over all elements in $\prod_{i=1}^{k} \Lambda_i$ and to save information for each element. However, our goal is to keep the computational effort – in the sense of optimization problems to solve over all elements – small.

During application of the following algorithms a point $\boldsymbol{y} \in \prod_{i=1}^{k} \Lambda_i$ can have four different current states encoded by $\sigma(\boldsymbol{y})$:

- $\sigma(\boldsymbol{y}) = \textit{attainable}$: The algorithm found an $\boldsymbol{x}$ with

$$(\boldsymbol{d} \circ \boldsymbol{f})(\boldsymbol{x}) \leqq \boldsymbol{y}.$$

  In this case we save $\boldsymbol{x}$ in $X^*(\boldsymbol{y})$.
- $\sigma(\boldsymbol{y}) = \textit{dominated}$: The algorithm determined a point $\tilde{\boldsymbol{y}} \in \mathcal{Y}$ that is attainable and that dominates $\boldsymbol{y}$.
- $\sigma(\boldsymbol{y}) = \textit{unattainable}$: The algorithm checked that there is no point $\boldsymbol{x}$ that is mapped to $\boldsymbol{y}$ or any point that dominates $\boldsymbol{y}$.
- $\sigma(\boldsymbol{y}) = \textit{unknown}$: The algorithm did not make any statement about this point, yet.

Note that we do not need to explicitly encode all points that are attainable but non-dominated. If we found correctly all points that are attainable and of these points all that are dominated, then the complement of the dominated points are the non-dominated ones. We can also be sure to have calculated a preimage $\boldsymbol{x}$ for each non-dominated $\boldsymbol{y}$. As for each attainable point $\boldsymbol{y}$ we found an $\boldsymbol{x}$ with:

$$(\boldsymbol{d} \circ \boldsymbol{f})(\boldsymbol{x}) \leqq \boldsymbol{y}.$$

For a non-dominated point none of these inequalities can be strict and we must have:

$$(\boldsymbol{d} \circ \boldsymbol{f})(\boldsymbol{x}) = \boldsymbol{y}.$$

In the following, we call a point $\boldsymbol{y}$ a supported point of $\mathcal{P}^c$ with weights $\boldsymbol{w}$ (see e.g. Ehrgott 2005) if it solves the following optimization problem:

$$\min_{\boldsymbol{y} \in \mathcal{Y}^c} \sum_{i=1}^{k} w_i \cdot y_i .$$

3.1 Propagation rules

An immediate approach to exhaustively solve the problem $\mathcal{P}$ consists of solving, for each $\boldsymbol{y} \in \prod_{i=1}^{k} \Lambda_i$, the individual feasibility problem

$$\begin{aligned} &\text{Find } \boldsymbol{x} \in \boldsymbol{M} \\ &\text{such that } (\boldsymbol{d} \circ \boldsymbol{f})(\boldsymbol{x}) \leqq \boldsymbol{y}. \end{aligned} \tag{6}$$

However, solving all these problems without additional considerations would be very inefficient. Efficiency can be improved by propagating information on a found solution $\boldsymbol{x}$ of (6) and its image points $\boldsymbol{y}^c$ and $\boldsymbol{y}$, and thereby deducing the attainability or unattainability of other points $\tilde{\boldsymbol{y}} \in \prod_{i=1}^{k} \Lambda_i$. These propagation rules are the following:

I. If $\boldsymbol{y}$ is unattainable, and $\tilde{\boldsymbol{y}} \leqq \boldsymbol{y}$, then $\tilde{\boldsymbol{y}}$ is unattainable.
II. If $\boldsymbol{y}$ is attainable, and $\tilde{\boldsymbol{y}} \geqq \boldsymbol{y}$, $\tilde{\boldsymbol{y}} \neq \boldsymbol{y}$, then $\tilde{\boldsymbol{y}}$ is dominated.
III. Let $\boldsymbol{y}^c$ be a weakly non-dominated point of $\mathcal{P}^c$.
If $\tilde{\boldsymbol{y}} < \boldsymbol{d}(\boldsymbol{y}^c)$, then $\tilde{\boldsymbol{y}}$ is unattainable.
IV. Let $\boldsymbol{y}^c$ be a supported point of $\mathcal{P}^c$ with weights $\boldsymbol{w}$.
If $(\sup(\boldsymbol{d}^{-1}(\tilde{\boldsymbol{y}})) - \boldsymbol{y}^c)^T \boldsymbol{w} < 0$, then $\tilde{\boldsymbol{y}}$ is unattainable.

Note that the supremum over the set in the last rule is understood componentwise.

Lemma 3.1 *The propagation rules I to IV, introduced above, are valid.*

Proof The first two rules are self evident. For the third rule, note that as $\boldsymbol{y}^c$ is weakly non-dominated for $\mathcal{P}^c$, $\boldsymbol{d}(\boldsymbol{y}^c)$ is weakly non-dominated for $\mathcal{P}$ according to Lemma 2.1, and therefore there cannot be a point $\tilde{\boldsymbol{y}}$ that strictly dominates $\boldsymbol{d}(\boldsymbol{y}^c)$. For the last rule, as $\boldsymbol{y}^c$ is supported, by definition $\boldsymbol{w}^T \boldsymbol{y}^c \leq \boldsymbol{w}^T \tilde{\boldsymbol{y}}^{\boldsymbol{c}}$ for all $\tilde{\boldsymbol{y}}^{\boldsymbol{c}}$ from the image $\mathcal{Y}^c$ of the feasible set. This can be rewritten as

$$(\tilde{\boldsymbol{y}}^{\boldsymbol{c}} - \boldsymbol{y}^c)^T \boldsymbol{w} \geq 0$$

for all $\tilde{\boldsymbol{y}}^{\boldsymbol{c}} \in \mathcal{Y}^c$. Now assume for a point $\tilde{\boldsymbol{y}} \in \prod_{i=1}^{k} \Lambda_i$ we have

$$\begin{aligned} 0 &> (\sup(\boldsymbol{d}^{-1}(\tilde{\boldsymbol{y}})) - \boldsymbol{y}^c)^T \boldsymbol{w} \\ &= \sup(\boldsymbol{d}^{-1}(\tilde{\boldsymbol{y}}))^T \boldsymbol{w} - \boldsymbol{w}^T \boldsymbol{y}^c. \end{aligned}$$

Now observe that $\sup(\boldsymbol{d}^{-1}(\tilde{\boldsymbol{y}}))^T \boldsymbol{w} \geq \boldsymbol{w}^T \tilde{\boldsymbol{y}}^{\boldsymbol{c}}$ for all $\tilde{\boldsymbol{y}}^{\boldsymbol{c}} \in \boldsymbol{d}^{-1}(\tilde{\boldsymbol{y}})$ as $\boldsymbol{w} \geqq 0$ and therefore the dot product with $\boldsymbol{w}$ is a positive map. Hence

$$\begin{aligned} 0 &> \boldsymbol{w}^T \tilde{\boldsymbol{y}}^{\boldsymbol{c}} - \boldsymbol{w}^T \boldsymbol{y}^c \\ &= (\tilde{\boldsymbol{y}}^{\boldsymbol{c}} - \boldsymbol{y}^c)^T \boldsymbol{w} \end{aligned}$$

for all $\tilde{y}^c \in \boldsymbol{d}^{-1}(\tilde{\boldsymbol{y}})$, which proves that the preimage $\boldsymbol{d}^{-1}(\tilde{\boldsymbol{y}})$ and the image of the feasible set are disjoint, and hence $\tilde{\boldsymbol{y}}$ is unattainable. □

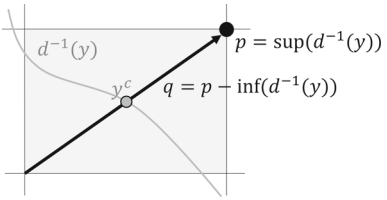


Fig. 4 Illustration of the Pascoletti-Serafini scalarization, which can be used in the box checking algorithm. The scalarization finds the intersection point of the box diagonal and the Pareto front

3.2 Box checking algorithm

The first algorithm we discuss is called the box checking algorithm. It systematically checks the attainability of points $\boldsymbol{y} \in \prod_{i=1}^{k} \Lambda_i$, making effective use of the propagation rules I.- III. Attainability is checked by solving specific scalarizations of the underlying continuous problem $\mathcal{P}^c$.

Definition 3.2 We call a single-criteria optimization problem a $\boldsymbol{y}$-constraint scalarization of $\mathcal{P}^c$, and denote it with $\mathcal{S}(\mathcal{P}^c, \boldsymbol{y})$, if the following hold:

i.) The optimization problem has an optimal solution if and only if $\boldsymbol{y}$ is attainable.
ii.) For any optimal solution $\boldsymbol{x}$ of the problem, the relation $(\boldsymbol{d} \circ \boldsymbol{f})(\boldsymbol{x}) \leqq \boldsymbol{y}$ holds.

Such a $\boldsymbol{y}$-constraint scalarization solves the feasibility problem (6). Under the assumption that all utility functions d_i are left-continuous, a version of a $\boldsymbol{y}$-constraint scalarization is the following Pascoletti-Serafini scalarization (Pascoletti and Serafini 1984), where $\boldsymbol{p} := \sup(\boldsymbol{d}^{-1}(\boldsymbol{y}))$ and $\boldsymbol{q} := \boldsymbol{p} - \inf(\boldsymbol{d}^{-1}(\boldsymbol{y}))$:

$$\begin{aligned} &\min_{\boldsymbol{x} \in M, \alpha \leq 0} \alpha \\ &\boldsymbol{p} + \alpha \boldsymbol{q} \geqq \boldsymbol{f}(\boldsymbol{x}). \end{aligned} \tag{7}$$

The applicable propagation rules depend on the chosen scalarization. If the feasibility problem (6) is solved, only the basic propagation rules I. and II. can be applied. If the Pascoletti–Serafini problem (7) is used as scalarization, propagation rule III. can additionally be applied, as any optimal solution to (7) is weakly Pareto efficient. The observations result in Algorithm 1.

At the beginning of Algorithm 1, the state map σ and the solution map X^* are initialized (ll 1–2). Then, in each iteration a point $\boldsymbol{y}$ is picked whose state is *unknown*, and a $\boldsymbol{y}$-constraint scalarization is set up (ll 4–5). If the scalarization can be solved to feasibility, $\sigma(\boldsymbol{y})$ is set to *attainable* and the solution is stored in the solution map (ll 6–9); otherwise, $\sigma(\boldsymbol{y})$ is set to *unattainable* (l 11). The state information is additionally updated according to the applicable propagation rules (l 13). When no point with state *unknown* remains, the algorithm returns the status map σ and the solution map X^* (l 16).

Note that the algorithm is guaranteed to terminate, as in each iteration at least one point $\boldsymbol{y}$ is removed from the set of points with $\sigma(\boldsymbol{y})$ *unknown*. Also, for all points $\boldsymbol{y} \in \prod_{i=1}^{k} \Lambda_i$ that have been found to be non-dominated, the algorithm will have found a solution $\boldsymbol{x} \in M$ with $\boldsymbol{y} = (\boldsymbol{d} \circ \boldsymbol{f})(\boldsymbol{x})$. That is, on termination of the algorithm, the problem $\mathcal{P}$ is solved exhaustively.

Algorithm 1 Box checking algorithm

Set $\sigma(\boldsymbol{y})$ to *unknown* for all $\boldsymbol{y} \in \prod_{i=1}^{k} \Lambda_i$
Set $X^*(\boldsymbol{y})$ to $\emptyset$ for all $\boldsymbol{y} \in \prod_{i=1}^{k} \Lambda_i$
while $\exists\ \boldsymbol{y}$ with $\sigma(\boldsymbol{y})$ *unknown* **do**
  Pick $\boldsymbol{y}$ with $\sigma(\boldsymbol{y})$ *unknown*
  Solve $\boldsymbol{y}$-constraint scalarization $\mathcal{S}(\mathcal{P}^c, \boldsymbol{y})$
  if solution $\boldsymbol{x}^*$ found **then**
    Let $\boldsymbol{y}^* := \boldsymbol{d}(\boldsymbol{f}(\boldsymbol{x}^*))$
    Set $\sigma(\boldsymbol{y}^*)$ to *attainable*
    Set $X^*(\boldsymbol{y}^*)$ to $\boldsymbol{x}^*$
  else
    Set $\sigma(\boldsymbol{y})$ to *unattainable*
  end if
  Apply propagation rules I., II. and possibly III.
end while
return status map σ and solution map X^*

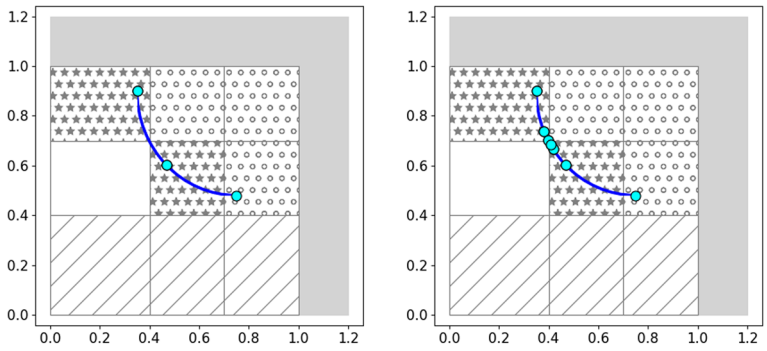
One crucial aspect of the box checking algorithm is the strategy employed to pick the next $\boldsymbol{y}$ with $\sigma(\boldsymbol{y})$ *unknown* at the beginning of each iteration. In our implementation, the strategy is as follows. For a given *unknown* $\boldsymbol{y}$, denote $N^1(\boldsymbol{y})$ the number of *unknown* $\tilde{\boldsymbol{y}}$ that would be found *unattainable* if $\boldsymbol{y}$ was *unattainable* (according to propagation rule I). Conversely, denote $N^2(\boldsymbol{y})$ the number of *unknown* $\tilde{\boldsymbol{y}}$ that would be found *attainable* if $\boldsymbol{y}$ was *attainable* (according to propagation rule II). Pick the *unknown* $\boldsymbol{y}$ for which $min\left\{N^1(\boldsymbol{y}), N^2(\boldsymbol{y})\right\}$ is largest. If this leaves more than one option to choose, pick one with largest $N^1(\boldsymbol{y}) + N^2(\boldsymbol{y})$ among those.

3.3 Algorithm using supported solutions

This algorithm (given in Algorithm 2) utilizes weighted sum scalarizations (see Eq. 6) of the inner continuous problem $\mathcal{P}^c$. As the points obtained through a weighted sum scalarization are, by definition, not only non-dominated but even supported, all of the propagation rules I.-IV. can be applied.

Algorithm 2 Supported solutions algorithm

Set $\sigma(\boldsymbol{y})$ to *unknown* for all $\boldsymbol{y} \in \prod_{i=1}^{k} \Lambda_i$
Set $X^*(\boldsymbol{y})$ to $\emptyset$ for all $\boldsymbol{y} \in \prod_{i=1}^{k} \Lambda_i$
while $\exists\ \boldsymbol{y}$ with $\sigma(\boldsymbol{y})$ *unknown* **do**
  Pick $\boldsymbol{w}$
  Solve weighted sum scalarization with weights $\boldsymbol{w}$ and obtain $\boldsymbol{x}^*$
  Let $\boldsymbol{y}^* := \boldsymbol{d}(\boldsymbol{f}(\boldsymbol{x}^*))$
  Set $X^*(\boldsymbol{y}^*)$ to $\boldsymbol{x}^*$
  Set $\sigma(\boldsymbol{y}^*)$ to *attainable*
  Apply propagation rules I.-IV.
end while
return status map σ and solution map X^*



(a) After three iterations. (b) After seven iterations.

Fig. 5 An example of bad performance of the supported solutions algorithm. While the first three iterations reduced the number of $\boldsymbol{y} \in \prod_{i=1}^{k} \Lambda_i$ with state *unknown* (white area) to one, in the next four iterations no further progress was made

After initialization (ll 1–2), Algorithm 2 iteratively calculates a new point $\boldsymbol{y}^*$ as the image of a solution $\boldsymbol{x}^*$ of a weighted sum scalarization (ll 5–6). In each iteration, the solution map is updated, and the state information is refreshed according to the propagation rules (ll 7–9). The algorithm returns the status map σ and the solution map X^* when no point $\boldsymbol{y}$ with state *unknown* remains (l 16).

The algorithm can be used effectively as a first phase algorithm for both convex and non-convex problems. As all four propagation rules I.-IV. can be applied, a lot of progress can be achieved early on. However, a switch to the box checking Algorithm 1 should be triggered the moment no meaningful progress is achieved any more.

Even for convex problems, the algorithm on its own is in later iterations not the most effective choice. Unlike the box checking algorithm, its termination is not guaranteed. The reason for this is that the calculated points are determined indirectly through the weights. If only a few particular $\boldsymbol{y} \in \prod_{i=1}^{k} \Lambda_i$ remain unknown, they cannot be targeted directly. Figure 5 shows a worst-case scenario, where the algorithm does not make progress over many iterations.

The strategy for picking the next weight vector $\boldsymbol{w}$ at the beginning of each iteration (line 4) is as follows. For the first k iterations, we choose the extreme compromises, where one of the entries of $\boldsymbol{w}$ is set to one and the rest are set to zero. For later iterations, we calculate the convex hull of the previously found solutions. We determine the face that intersects with the greatest number of preimage sets $\boldsymbol{d}^{-1}(\boldsymbol{y})$ and pick its normal as the new $\boldsymbol{w}$.

3.4 Algorithm for convex problems

If $\mathcal{P}^c$ is a convex problem, Algorithm 2 can be modified to, in each iteration, calculate and employ the convex hull of the images $\boldsymbol{f}(X^*)$, where X^* denotes the set of all solutions found so far. The following Lemma will be needed.

Lemma 3.3 *Let* $\{\boldsymbol{x}^1, ..., \boldsymbol{x}^n\} \in M$ *be a set of efficient solutions to the convex problem* $\mathcal{P}^c$*, and let*

$$\boldsymbol{d}^{-1}(\boldsymbol{y}) \cap conv(\boldsymbol{f}(\boldsymbol{x}^1), .., \boldsymbol{f}(\boldsymbol{x}^n)) \neq \emptyset$$

for a point $\boldsymbol{y} \in \prod_{i=1}^{k} \Lambda_i$*. Then there is* $\boldsymbol{x}^{int} \in M$ *such that* $(\boldsymbol{d} \circ \boldsymbol{f})(\boldsymbol{x}^{int}) \leqq \boldsymbol{y}$.

Proof Let $\boldsymbol{y}^c \in \boldsymbol{d}^{-1}(\boldsymbol{y}) \cap conv(\boldsymbol{f}(\boldsymbol{x}_1), .., \boldsymbol{f}(\boldsymbol{x}_n))$.Then by definition

$$\boldsymbol{d}(\boldsymbol{y}^c) = \boldsymbol{y},$$

and there are coefficients $0 \leq c_i \leq 1$ $(i = 1, .., n)$ such that

$$\boldsymbol{y}^c = \sum_{i=1}^{n} c_i \boldsymbol{f}(\boldsymbol{x}_i).$$

Now let $\boldsymbol{x}^{\text{int}} := \sum_{i=1}^{n} c_i \boldsymbol{x}_i$. Because of convexity of $\boldsymbol{f}$, we have

$$\boldsymbol{f}(\boldsymbol{x}^{\text{int}}) \leqq \boldsymbol{y}^c.$$

By the componentwise monotonicity of $\boldsymbol{d}$, we obtain $(\boldsymbol{d} \circ \boldsymbol{f})(\boldsymbol{x}^{\text{int}}) \leqq \boldsymbol{y}$, which shows the claim. □

Algorithm 3 Algorithm for convex problems

1: Set $\sigma(\boldsymbol{y})$ to *unknown* for all $\boldsymbol{y} \in \prod_{i=1}^{k} \Lambda_i$
2: Set $X^*(\boldsymbol{y})$ to $\emptyset$ for all $\boldsymbol{y} \in \prod_{i=1}^{k} \Lambda_i$
3: **while** $\exists\ \boldsymbol{y}$ with $\sigma(\boldsymbol{y})$ *unknown* **do**
4: Pick $\boldsymbol{w}$
5: Solve weighted sum scalarization with weights $\boldsymbol{w}$ and obtain $\boldsymbol{x}^*$
6: Let $\boldsymbol{y}^* := \boldsymbol{d}(\boldsymbol{f}(\boldsymbol{x}^*))$
7: Set $X^*(\boldsymbol{y}^*)$ to $\boldsymbol{x}^*$
8: Set $\sigma(\boldsymbol{y}^*)$ to *attainable*
9: Apply propagation rules I.-IV.
10: **for** $\boldsymbol{y}$ with $\sigma(\boldsymbol{y})$ *unknown* and $\boldsymbol{d}^{-1}(\boldsymbol{y}) \cap conv(\boldsymbol{f}(X^*)) \neq \emptyset$ **do**
11: Set $\sigma(\boldsymbol{y})$ to *attainable*
12: Set $X^*(\boldsymbol{y})$ to the interpolated solution $\boldsymbol{x}^{\text{int}}$
13: Apply propagation rule II.
14: **end for**
15: **end while**
16: **return** status map σ and solution map X^*

Along the lines of Algorithm 2, Algorithm 3 iteratively calculates a new point $\boldsymbol{y}^*$ as the image of the solution $\boldsymbol{x}^*$ of a weighted sum scalarization (ll 5–6). After updating the solution set and the state information according to the propagation rules (ll 7–9), Lemma 3.3 is applied to potentially identify additional points $\boldsymbol{y}$ as attainable. If that is the case, the interpolated solution $\boldsymbol{x}^{\text{int}}$ is stored in the solution map (ll 10–14).

In accordance with Lemma 3.3, the interpolated solution $\boldsymbol{x}^{\text{int}}$ in Algorithm 3 is defined as follows: if $\boldsymbol{d}^{-1}(\boldsymbol{y}) \cap conv(\boldsymbol{f}(X^*)) \neq \emptyset$ is established by finding coefficients $0 \leq c_i \leq 1$ $(i = 1, .., n)$ such that for $\boldsymbol{y}^c := \sum_{i=1}^{n} c_i \boldsymbol{f}(\boldsymbol{x}_i)$ we have $\boldsymbol{d}(\boldsymbol{y}^c) = \boldsymbol{y}$, then $\boldsymbol{x}^{\text{int}} := \sum_{i=1}^{n} c_i \boldsymbol{x}^i$ (see proof of Lemma 3.3 for the rationale behind this.)

For picking the weight vector $\boldsymbol{w}$ at the beginning of each iteration (line 4), we choose the same strategy as the one described above for the algorithm using supported solutions.

4 Examples

In this section, we will investigate the performance of the described algorithms over both a set of test problems and a real world example from radiotherapy plan optimization. The three algorithms we will compare are the following:

(A) Box checking Algorithm 1 using the Pascoletti–Serafini scalarization (7).
(B) Combined algorithm where, in a first phase, the supported solutions algorithm (Algorithm 2) is employed until no further progress is made (i.e. an iteration occurs where the number of boxes with status *unknown* is not reduced.) From then on, the box checking algorithm with Pascoletti-Serafini scalarization is used until the problem is solved exhaustively.
(C) Combined algorithm where, in the first phase, the algorithm for convex problems (Algorithm 3) is used instead.

4.1 Randomized problems

We consider the following problem

$$\begin{aligned} (d_1^K(x_1), ..., d_k^K(x_k)) \rightarrow \min \\ s.t. \ \sum_{i=1}^{k} \left(\frac{x_i - c_i}{r_i} \right)^2 \leq 0 \\ 0 \leq x_i \leq 1 \ \ \forall i = 1, .., k \end{aligned} \tag{8}$$

where the coordinates c_i are randomly chosen from $[0.6, 1.0]$ and the half lengths r_i are randomly chosen from $[0.2, 0.5]$. We investigate different utility functions $\boldsymbol{d}^K$ that differ in the cardinality K of their image sets Λ_i. Namely,

$$d_i^3(x_i) = \begin{cases} 0.4 & \text{if} \quad 0 \leq x_i \leq 0.4, \\ 0.7 & \text{if} \quad 0.4 < x_i \leq 0.7, \\ 1.0 & \text{if} \quad x_i > 0.7 \end{cases} \quad \forall i = 1, .., k$$

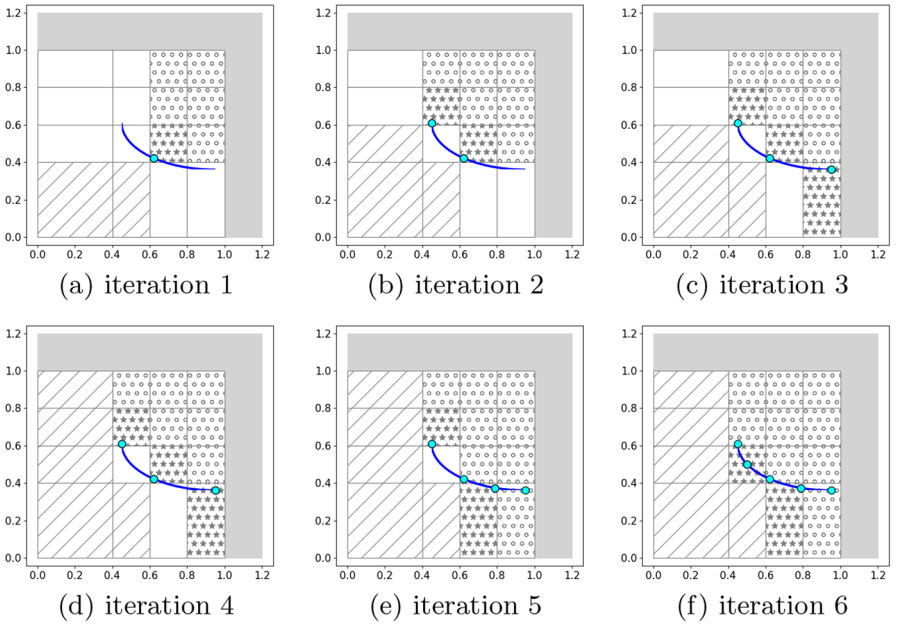


Fig. 6 The box checking algorithm, using the Pascoletti-Serafini scalarization, for a problem instance of 8 with $k = 2$ and $K = 4$. In total, six iterations are necessary to exhaustively solve the problem

$$d_i^4(x_i) = \begin{cases} 0.4 & \text{if} \quad 0 \leq x_i \leq 0.4, \\ 0.6 & \text{if} \quad 0.4 < x_i \leq 0.6, \\ 0.8 & \text{if} \quad 0.6 < x_i \leq 0.8, \\ 1.0 & \text{if} \quad x_i > 0.8 \end{cases} \quad \forall i = 1, .., k$$

$$d_i^5(x_i) = \begin{cases} 0.4 & \text{if} \quad 0 \leq x_i \leq 0.4, \\ 0.5 & \text{if} \quad 0.4 < x_i \leq 0.5, \\ 0.6 & \text{if} \quad 0.5 < x_i \leq 0.6, \\ 0.8 & \text{if} \quad 0.6 < x_i \leq 0.8, \\ 1.0 & \text{if} \quad x_i > 0.8 \end{cases} \quad \forall i = 1, .., k.$$

The underlying continuous problem $\mathcal{P}^c$ of the randomized problems (8) is convex, allowing the application of all algorithms (A), (B) and (C). For a specific problem instance with $k = 2$ and $K = 4$, Fig. 6 illustrates the course of the box checking algorithm (A), and Fig. 7 depicts the course when using the combined algorithm for convex problems (C) instead. We see that for this instance, the combined algorithm for convex problems outperforms the box checking algorithm, finishing in 3 rather than 6 iterations.

We can compare the performance of algorithms (A)–(C) by measuring the progress p over the iterations, with p being defined as:

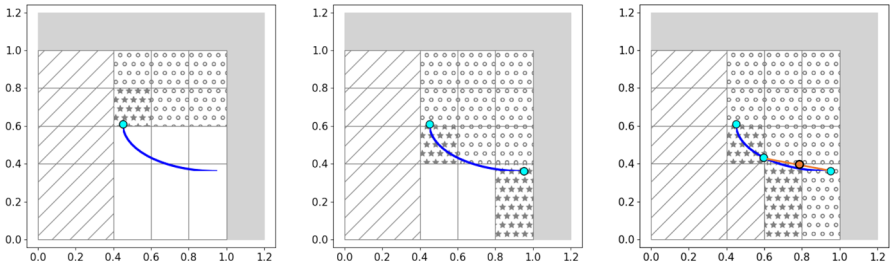


Fig. 7 The convex algorithm applied to a problem instance of Eq. 8 with $k = 2$ and $K = 4$. Only three iterations are necessary to determine all non-dominated points. Note that the solution for one of the non-dominated outcomes is obtained by convex combination of previously calculated points according to Lemma 3.3 (orange)

$$p = \frac{1 - |\{\boldsymbol{y} \in \prod_{i=1}^{k} \Lambda_i \,|\, \sigma(\boldsymbol{y}) unknown\}|}{|\prod_{i=1}^{k} \Lambda_i|}. \tag{9}$$

Figure 8 shows the averaged progress for the artificial problem 8 over varying number of objectives $k \in \{2, 3, 4\}$ and image set cardinalities $K \in \{3, 4, 5\}$. To create each of the averaged plots, 10 randomly created problem instances were solved and the mean of obtained progress values p after each iteration was calculated.

The combined convex algorithm (C) performs best by exploiting the convexity of the problem. The next best choice is the combined algorithm that uses the supported solutions algorithm as a first phase. The box checking algorithm on its own does not perform as well. With higher dimension k and higher image set cardinality K, the differences in algorithm performance become more pronounced.

4.2 Radiotherapy planning example

In radiotherapy planning, the aim is to deliver the prescribed dose to the target volumes while sparing nearby organs at risk. In intensity-modulated radiation therapy (IMRT), the irradiation—the so-called *fluence*—is delivered from different angles around the patient and can additionally be modulated over the cross section of each beam by moving collimator leaves in and out of the beam field.

The fluence is represented by a vector $\boldsymbol{x} \geqq 0$ and constitutes the optimization variable of the multi-criteria radiotherapy planning problem (Küfer et al. 2002; Küfer et al. 2003; Craft et al. 2012). The *dose influence matrix* D depends on the patient anatomy as established by a CT scan and promotes the mapping from the fluence to the dose, such that the entries of $D\boldsymbol{x}$ represent the dose for each voxel in the voxelized patient anatomy.

The optimization objectives then evaluate the dose vector entries over the target volumes and organs at risk. These evaluation functions are continuous and often convex. Possible choices are minimum, maximum, mean, p-norms or one-sided p-norms.

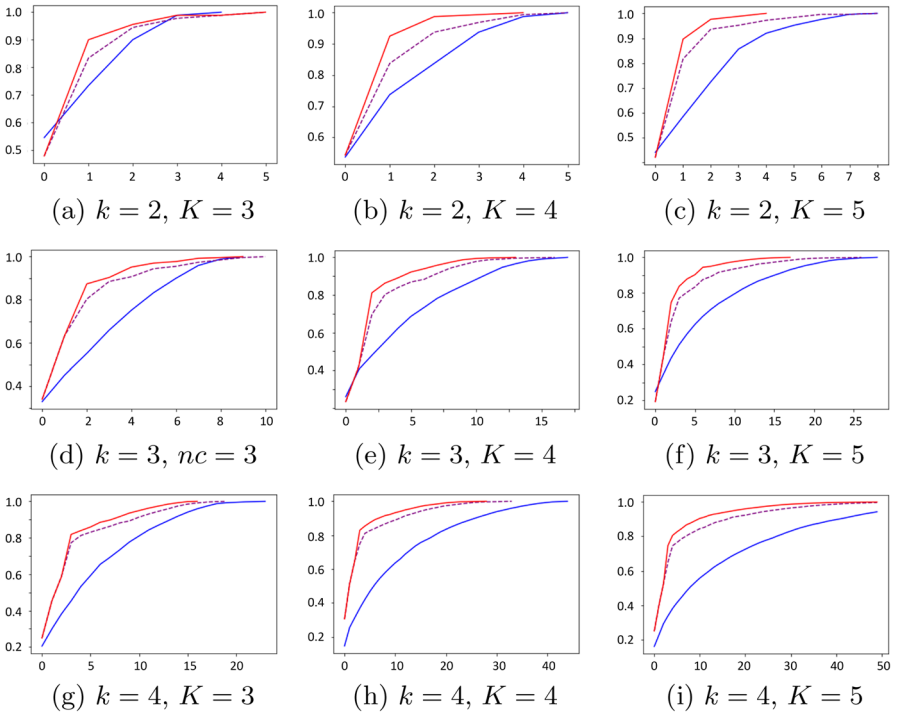


Fig. 8 Averaged progress for algorithms (**A**)–(**C**): box checking algorithm (dark blue), combined algorithm with supported solutions algorithm as first phase (purple dotted) and combined algorithm with convex algorithm as first phase (red). The averages were taken over 10 randomized problem instances of 8. The number of objectives k was varied from 2 to 4 and the image set cardinality K was varied from 3 to 5

Thus, the (continuous) multi-criteria radiotherapy planning problem has the properties required for the inner problem $\mathcal{P}^c$ in our setting.

In clinical practice, certain threshold values for the objectives play a crucial role. In the case of organs at risk, violating a threshold is linked to specific side effects. For the targets, failing to meet a specific value may increase the probability of the tumor recurring. Often, these clinical goals are further categorized into those which represent the absolute minimum requirement, and others that correspond to an average or an ideal dose distribution in the volume. Taking these discretizations ("minimum", "average", "ideal") into account, we obtain the structure of the continuous problem with discretized utility $\mathcal{P}$.

As an example, we consider the TG119 case. The anatomy of the case has a U-shaped target volume, an organ at risk (OAR) in the center of the target, and the surrounding normal tissue region. We define the beam geometry as 7 equidistantly spaces beams, see Fig. 9.

To set up the optimization problem, we formulate one objective for each structure as detailed in Table 1. Additionally, for each objective we define the discretization mappings into the categories 3 ("minimum"), 2 ("average"), and 1 ("ideal") by certain upper bounds, also given in Table 1, such that the discretized utility functions d_i

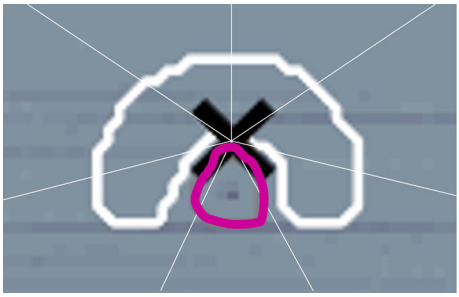


Fig. 9 Geometry of the TG119 case. It has a U-shaped target volume (white), an organ at risk (magenta) in the center of the target, and surrounding normal tissue. Seven beams are spaced equidistantly (white lines)

Table 1 Optimization model for the TG119 case: optimization objectives for target, OAR and tissue, and threshold values for the discretization into categories 1–3

Volume	Objective	Categories: 3	2	1
OAR	$\frac{1}{\lvert OAR\rvert}\sum_{v\in OAR}(Dx)_v$	25.0	23.0	20.0
Tissue	$\frac{1}{\lvert tissue\rvert}\sum_{v\in OAR}(Dx)_v$	8.0	6.0	5.0
Target	$\sum_{v\in target}\lVert\max\{0, 60-(Dx)_v\}\rVert^2$	2.0	1.5	1.0

$(i = 1, 2, 3)$ map to the smallest category for which $f_i(D\boldsymbol{x})$ falls below the category's upper bound.

Again, we compared the algorithms (A)–(C). Figure 10 shows the result obtained with all three algorithms. There are two Pareto-optimal solutions to the discretized problem: one where the target and the tissue dose quality are ideal while the OAR dose quality is average, and one where the OAR and tissue dose quality are ideal and the target dose quality is average. This is reflected in the dose volume histograms for the two solutions, where the first solution shows a better slope at the prescription dose level of 60 Gy, while the other solution exhibits a lower OAR curve.

Figure 11 shows the progress—as defined in (9)—for the different algorithms over the iterations. For this particular example, the algorithms do not differ much in their efficiency. The combined algorithm that uses the convex algorithm as a first phase performs the best, followed by the box checking algorithm and the other combined algorithm.

5 Conclusion

In this paper we considered multi-criteria optimization problems which are based on a continuous problem. The outputs of the objectives are mapped using a discrete utility function making the overall multi-criteria optimization a discrete one. For example, such categories can be used to quickly find candidate solutions for a decision maker, especially when the Pareto front is no longer easily visualized. In a first step a decision maker might want to get a rough overview of the different alternatives instead of navigating locally.

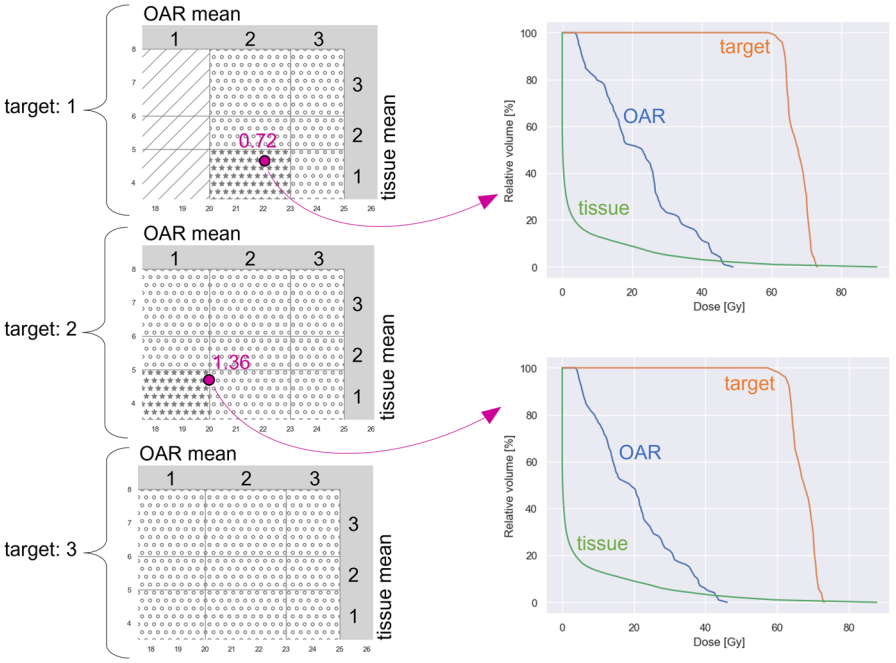


Fig. 10 The result for the TG119 example. On the left side, the OAR mean objective and its discretization into categories is plotted against the tissue mean objective and its discretization for each of the 3 categories of the target underdose objective. There are two Pareto optimal outcomes in the discretized space. These are achieved by two efficient solutions $\boldsymbol{x}$ mapping to the marked points under the continuous objectives defined in Table 1 (the target objective value being shown in purple.) The dose distributions for the two solutions—displayed as dose-volume histograms at the right—represent two distinct compromises, with the upper solution achieving a better target coverage and the lower solution a better sparing of the OAR

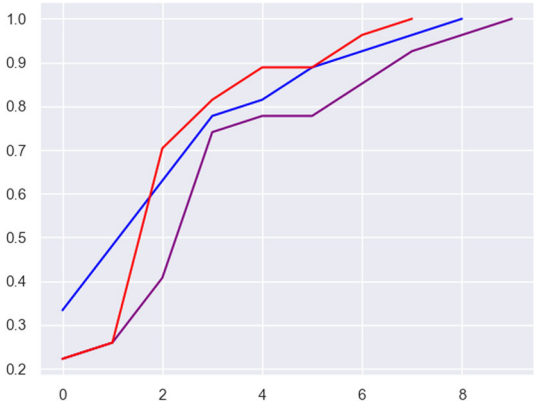


Fig. 11 Progress for algorithms (**A**)–(**C**) for the TG119 radiotherapy planning example: box checking algorithm (dark blue), combined algorithm with supported solutions algorithm as first phase (purple), and combined algorithm with convex algorithm as first phase (red)

In this paper, we first studied the connection between solutions of the underlying continuous problem and the combined multi-criteria problem. The results were used to introduce several algorithms that combine the discrete and continuous structure. In numerical examples, we were able to show that our approaches can save a lot of computation compared to a naive approach. For larger problems with more objectives and more categories, the experiments showed that it is beneficial to use as much information as possible and also to exploit convexity.

Several aspects were beyond the scope of this paper, but may be of interest for future investigations. We focused on fully discrete utilities. In Sect. 2 we already pointed out that this is not necessarily required from a theoretical point of view. All results hold for monotone functions in general. It would be interesting to look at combined continuous and discrete utilities. We also focused on problems with a moderate number of categories and objectives. This way it was not a problem to iterate over all possible category combinations. It would be interesting to study how the developed strategies transfer to this larger setting.

Funding Open Access funding enabled and organized by Projekt DEAL.

Declarations

Conflict of interest The authors have no competing interests to declare that are relevant to the content of this article. No funding was received to assist with the preparation of this manuscript.

References

Benson HP, Sayin S (1997) Towards finding global representations of the efficient set in multiple objective mathematical programming. Nav Res Logist 44:47–67

Bokrantz R, Forsgren A (2012) An algorithm for approximating convex pareto surfaces based on dual techniques. Informs J Comput 25(2)

Bringmann K, Friedrich T (2010) Tight bounds for the approximation ratio of the hypervolume indicator. In: Schaefer R, Cotta C, Kołodziej J, Rudolph G (eds) Parallel problem solving from nature. PPSN XI, volume 6238 of Lecture Notes in Computer Science. Springer, Berlin, pp 607–616

Byrd RH, Nocedal J, Waltz RA (2006) Knitro: an integrated package for nonlinear optimization. Springer, Berlin, pp 35–59

Craft DL, Hong TS, Shih HA, Bortfeld TR (2012) Improved planning time and plan quality through multicriteria optimization for intensity-modulated radiotherapy. Int J Rad Onc Biol Phys 82(1):e83–e90

Das I, Dennis J (1998) Normal-boundary intersection: a new method for generating the pareto surface in nonlinear multicriteria optimization problems. SIAM J Optim 8:631–657

Ehrgott M (2005) Multicriteria optimization: with 88 figures and 12 tables, 2nd edn. Springer, Berlin

Eichfelder Gabriele (2009) Scalarizations for adaptively solving multi-objective optimization problems. Comput Optim Appl 44:249–273

Holzmann T, Smith JC (2018) Solving discrete multi-objective optimization problems using modified augmented weighted tchebychev scalarizations. Eur J Oper Res 271:436–449
Klamroth K, Tind J, Wiecek MM (2002) Unbiased approximation in multicriteria optimization. Math Methods Oper Res 56(3):413–437
Küfer KH, Hamacher H, Hans W (2002) Inverse radiation therapy planning—a multiple objective optimization approach. Discrete Appl Math 118
Küfer K-H, Scherrer A, Monz M, Alonso F, Trinkaus H, Bortfeld T, Thieke C (2003) Intensity-modulated radiotherapy—a large scale multi-criteria programming problem. OR Spectrum 25:223–249
Pascoletti A, Serafini P (1984) Scalarizing vector optimization problems. J Optim Theory Appl 42(4):499–524
Sayın Serpil (2000) Measuring the quality of discrete representations of efficient sets in multiple objective mathematical programming. Math Program 87:543–560
Schandl B, Klamroth K, Wiecek MM (2002) Norm-based approximation in multicriteria programming. Comput Math Appl 44:925–942
Serna JI (2012) Multi-objective optimization in mixed integer problems with application to the beam angle optimization problem in IMRT. Ph.D. thesis, Technical University of Kaiserslautern, Department of Mathematics
Steuer RE, Choo EU (1983) An interactive weighted tchebycheff procedure for multiple objective programming. Math Program 26:326–344
Teichert IK (2014) A hyperboxing Pareto approximation method applied to radiofrequency ablation treatment planning. Ph.D. thesis, Technical University of Kaiserslautern, Department of Mathematics
Ulungu EL, Teghem J (1994) Multi-objective combinatorial optimization problems: a survey. Multi-criteria Decis Anal 3:83–104
Wächter A, Biegler LT (2006) On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming. Math Program 106:25–57